Solving chemical master equation for large biological networks: novel state-space expansion algorithm based on Artificial Intelligence and Bayesian standards

by

Rahul Kosarwal

A thesis submitted in partial fulfilment

of the requirements for the Degree of

Doctor of Philosophy

in Computational Systems Biology

at

Lincoln University

New Zealand

2019

This thesis is dedicated to both of my parents, who have raised me to the person I am today. You have been with me every step of the way, through good times and bad. Thank you for all the unconditional love, guidance, and support that you have given me, helping me to succeed and instilling in me the confidence that I am capable of doing anything I put my mind to.

Abstract

Solving Chemical Master Equation for Large Biological Networks: Novel State-Space Expansion Algorithm based on Artificial Intelligence and Bayesian Standards

by

Rahul Kosarwal

Numerical solutions of the chemical master equation (CME) are gaining increasing attention with the need to understand the stochasticity of biochemical systems. In this thesis, we aim to develop the intelligent state projection (ISP) method to use in the stochastic analysis of these systems. For any biochemical reaction network, it is important to capture more than one moment to describe the dynamic behaviour of the system. ISP is based on a state-space search and the data structure standards of artificial intelligence (AI) to explore and update the states of a biochemical system. To support the expansion in ISP, we also develop a Bayesian likelihood node projection ($BLNP$) function to predict the likelihood of the states. The proposed algorithm systematically expands the projection space based on predefined inputs, which are useful in providing accuracy in the approximation (Æ) and an exact analytical solution at the time of interest. We also discuss the applications of the ISP algorithm for realistic models and compare this with other existing, and latest, domain expansion methods based on the *r-step reachability* of the finite state projection (FSP) and the stochastic simulation algorithm (SSA). In all examples, our proposed methods perform better in terms of speed and accuracy of the expansion, accuracy of the solution, and provide a better understanding of the state-space of the system.

***Keywords*:** Biochemical reaction network, chemical master equation, stochastic, intelligent state projection, Bayesian likelihood node projection, mathematical modelling, ordinary differential equations.

Acknowledgements

Writing a significant scientific thesis is hard work and it would be impossible without support from various people, to only some of whom it is possible to give particular mention here.

Above all, I would like to express my chasmic gratitude to my principal supervisor, *Professor Don Kulasiri*, for his constant encouragement, guidance and trust on me. Without his supervision, friendship and patience this thesis would not have been possible. I would also express my appreciation to my associate supervisor, *Professor Sandhya Samarasinghe*, for her insightful discussions, guidance and feedback. Their persistent supervison and guidance has been invaluable on both an academic and an personal level, for which I am extremely greatful to both of them.

I would like to thank my friends of C-fACS (Centre for Advanced Computational Solutions) for their support and productive discussions. In particular, I would like to thank Dr. Yao He, Dr. Jingyi Liang for their support during initial year of my doctoral study. I would also like to thank Ms. Junwen Zhang, Dr. Hamish Bullmore for their support, friendship and being my sport companion.

My special appreciation to Dr. Parul Tiwari, Mr. Vikas Tiwari and Dorothy Crosbie for all the affection, support and making my life in New Zealand very delightful.

I would like to express my appreciation to Lincoln University, Health Centre and IT Services team for providing the support and facilities during my study. Also, I would like to acknowledge Lorraine Holmes and Robyn Wilson for their warm care and generous support.

I would like to express my deepest appreciation to my brother, Mr. Anshul Kosarwal and sister-in-law, Ms. Meeta Kosarwal, for being my great source of inspiration and support throughout this study.

Last but not least, I am most grateful to my parents, Mr. O. P. Kosarwal and Mrs. Anita Kosarwal, and my wife Dr. Shamayitri Ghosh for their unconditional love, support, trust and encouragement. Also, I would like to thank my in-laws Dr.(Prof) Samarendranath Ghosh and Mrs. Sagorika Ghosh for their invaluable support and patience.

Table of Contents

Abstract ..... **iii**
Acknowledgements ..... **iv**
Table of Contents ..... **v**
List of Tables ..... **viii**
List of Figures ..... **x**
Abbreviations ..... **xviii**
Terminology ..... **xviii**
Notations ..... **xx**
Key Factors and Outputs ..... **xxii**

Chapter 1 Introduction **24**
1.1 Stochastic Chemical Kinetics 26
1.2 Motivation and Main Research Questions 28
1.3 Objectives for this study 30
1.4 Thesis structure 32

Chapter 2 Background and Literature Review **35**
2.1 Markov Processes – An Introduction for the Computational Biologist 36
2.2 The Chemical Master Equation 39
2.2.1 Initial Value Problem 41
2.2.2 Time Homogeneous Markov Processes 43
2.3 Existing State-space Expansion Methods 44
2.3.1 r-step Reachability Method 45
2.3.2 Stochastic Simulation Methods 49
2.4 Existing Numerical Solution Methods 53
2.4.1 Uniformisation Method 54
2.4.2 Krylov Subspace 56
2.5 Biological Examples 58
2.6 Discussion and Conclusions 66

Chapter 3 Markov Chain Tree and Graphs for Markov Process **68**
3.1 Introduction 69
3.1.1 Definitions and Preliminaries 70
3.2 Finite state Markov Chains as sample space 72
3.2.1 Sample Space for Biochemical Systems 74
3.2.2 States Classification of Markov Chain for Biochemical System 76
3.3 Markov Chain as a Markov Chain Tree 79
3.4 Problem State-Space Model of Biochemical networks 86
3.5 Intelligent Search and Tracking 89
3.5.1 Artificial Intelligence for CME 89
3.5.1.1 Infrastructure for Searching *91*
3.5.2 Bayesian Likelihood Node Projection Function 97
3.6 Complexity of Optimal Solutions 104
3.7 Discussion and Conclusions 106

Chapter 4 Intelligent State Projection 108
4.1 Introduction 108
4.2 Derivation of the Method Conditions 114
4.2.1 Expansion Criterion for States Space! 114
4.2.2 Cease of Criterion after Updating 119
4.3 Latitudinal Search Strategy 120
4.3.1 Expansion and Update 125
4.3.2 Biological Example 137
4.4 Longitudinal Latitudinal Search 146
4.4.1 Expansion and Update 151
4.4.2 Biological Example 162
4.5 Data Structure Complexity of Operations 170
4.6 Discussion and Conclusion 171

Chapter 5 Comparative Study And Analysis 173
5.1 Study Overview 173
5.2 Comparison Based on Catalytic Reaction System 174
5.3 Comparsion Based on the Dual Enzymatic Reaction Network 178
5.4 Discussion and Conclusion 183

Chapter 6 Case study 1 - G1/S checkpoint involving the DNA-damage signal transduction pathway 185
6.1 Introduction 185
6.1.1 What Happens in Normal Conditions? 187
6.1.2 What Happens in the Presence of a DNA-damage signal? 188
6.2 Model Integration 189
6.3 Computational Experiments 198
6.4 Discussion and Summary 212

Chapter 7 Case study 2 – Oxidative stress adaptation in the Fungal Pathogen *Candida albicans* 214
7.1 Introduction 215
7.1.1 Integrated Model Overview 216
7.2 Model Integration 220
7.2.1 Transporter Module 229
7.2.2 Antioxidant Module 230
7.2.3 Protein-thiol Module 234
7.2.4 Signalling and Gene Expression Module 235
7.3 Computational Experiments 243
7.4 Discussion and Summary 282

Chapter 8 Conclusion and Future Directions 284
8.1 Overview of the study 284
8.2 Contributions 287
8.3 Future Directions 288
8.4 Conclusions 289

Appendix A . Probability Laws 291

Appendix B . Derivation of the CME 293

Appendix C . Expression for Rate Laws 296
C.1 Mass Action Based Models 296
C.2 Michaelis-Menten Model 298
C.2.1 Equilibrium approximation 298

Appendix D . Dilemma in Selection 300

Appendix E . Complexity Based on Operations 304

Appendix F . Implementation Details - Environment, Dependencies 308
F.1 Environment 308
F.2 Dependencies and Libraries 309

Appendix G . Replicate Experiments 312
G.1 Biological Experiments in Literature 312
G.2 Biological Experiments using *ISP* 313
G.3 Biological Experiments in Comparative Study 315
G.4 Output Data of Algorithms 316

Appendix H . Case Study 2 Supporting Information 317

References 321

List of Tables

Table 2.1. The *r-step reachability* steps. 46

Table 2.2. Step in the Stochastic Simulation Algorithm. 50

Table 3.1. Events with the likelihood of the future reactions. Where, *True* events define the expansion of nodes. 103

Table 4.1. Modules/sub-modules, components of the comprehensive ISP method..... 111

Table 4.2. Steps of ISP Latitudinal Search (LAS) Algorithm. 123

Table 4.3. LAS nodes expansion strategy for the toy model. 128

Table 4.4. LAS state update strategy for the toy model. 130

Table 4.5. LAS expansion of the state-space and solution of the toy model at different time points. 136

Table 4.6. LAS expansion response and solution at t_f for the catalytic system. 142

Table 4.7. Steps of ISP Longitudinal Latitudinal Search ($LOLAS$) Algorithm. 150

Table 4.8. $LOLAS$ nodes expansion strategy for the toy model..... 154

Table 4.9. $LOLAS$ states update strategy for the toy model..... 156

Table 4.10. LAS expansion of the state-space and solution of the toy model at different time points. 160

Table 4.11. $LOLAS$ expansion response and solution at t_f for the dual enzymatic reaction network. 166

Table 5.1. Comparison of the solution of the catalytic reaction system based on ISP, $OFSP$ and SSA. 174

Table 5.2. Comparison of the solutions of the dual enzymatic reaction system based on ISP, $OFSP$ and SSA..... 178

Table 6.1. Description of model variables of the G1/S checkpoint involving proteins to create the state-space for $ISP\ LOLAS$. 191

Table 6.2. Reactions in the G1/S model involving the DNA-damage signal transduction pathway..... 192

Table 6.3. Kinetic parameter values for reactions involved in the G1/S model..... 193

Table 6.4. Lower and upper Bounds of the domain for G1/S model given by $ISP\ LOLAS$ trend based on bound limit $б_{limit}$ 203

Table 6.5. $ISP\ LOLAS$ expansion response and solution at t_f for the G1/S model. 205

Table 7.1. Modules and components of comprehensive network of the integrated model. 217

Table 7.2. The description of model variables of the oxidative stress response in *C. albicans* pathway involving chemical and biochemical species to create the state-space for $ISP\ LAS$ 221

Table 7.3. Auxiliary variables involved in the oxidative stress response in the *C. albicans* pathway..... 222

Table 7.4. Biochemical reactions involved in oxidative stress response in the *C. albicans* pathway..... 225

Table 7.5. Kinetic parameter values for the biochemical reactions involved in the oxidative stress response in the *C. albicans* pathway..... 227

Table 7.6. Constants involved in the oxidative stress response in the *C. albicans* pathway..... 228

Table 7.7. Lower and upper Bounds of the *transporter module* given by $ISP\ LAS$ trend. 257

Table 7.8. Lower and upper Bounds of the *cat1 pathway* given by $ISP\ LAS$ trend..... 259

Table 7.9. Lower and upper Bounds of the *pentose pathway* given by $ISP\ LAS$ trend 261

Table 7.10. Lower and upper Bounds of the *thioredoxin* given by $ISP\ LAS$ trend..... 263

Table 7.11. Lower and upper Bounds of the *protein-thiol* given by $ISP\ LAS$ trend..... 265

Table 7.12. Lower and upper bounds of the *cap1 pathway* given by $ISP\ LAS$ trend..... 267

Table 7.13. Lower and upper bounds of the *hog1 pathway* given by $ISP\ LAS$ trend. 269

Table 7.14. $ISP\ LAS$ expansion output and solution at t_f for the Integrated model based on its different modules. 270

List of Figures

Figure 1.1. Solution methods to CME are a combination of state-space expansion methods and approximation (Æ) methods. 30

Figure 1.2. Flow chart showing organisation of thesis and major research points in the form of chapters. 34

Figure 2.1. Stochastic processes are categorised into different process. This study relates to the processes that are marked with black in the flowchart. 36

Figure 2.2. Visualisation of generation-recombination process for the Master equation. .. 43

Figure 2.3. Some existing state-space expansion methods used to create domain for the solution of CME. 44

Figure 2.4. Types of numerical solution methods – Uniformisation and Krylov subspace methods. Each is categorised based on global and local solution. 53

Figure 2.5. CME solution of T-cell homoeostasis at different *times (0, 1, 3, 5, 6, 10 secs)* with A (y-axis) and B (x-axis). The significant part of the probability mass is initially located at *t=0 sec* as given in Fig (A), then shifted further, as given in Fig (F) at t *=10 secs.* 59

Figure 2.6. SSA trajectories of mRNA and protein of a gene expression model upto $t_f =$ 2000 *secs*. 61

Figure 2.7. Probability distribution of species of gene expression model at $t_f = 2000$ *secs*. Fig (A) is the joint probability distribution of the species, Fig (B) is the stochastic versus deterministic response, Fig (C) and Fig (D) are the conditional probabilities of mRNA and protein respectively. 63

Figure 2.8. Conditional probability of the Michaelis Menten reaction system species evaluated at $t_f = 10.0$ *sec*, using $OFSP$ 65

Figure 2.9. Change in molecular counts of the species of Michaelis Menten reaction system over t_f. 65

Figure 3.1. Representation of Markov chain with six states and a transition matrix $[P]$. ... 74

Figure 3.2. Representation of the Markov chain of biochemical network with reactions and relevant propensities and transition matrix $[A_{i,j}]$. 75

Figure 3.3. Classification of transient and recurrent states 77

Figure 3.4. Markov chain graph showing forward and reversible reactions through four different states. 80

Figure 3.5. Equivalent tree of Markov chain graph, as shown in Figure 3.4. It is a special form of graph that has no cycle, no self-loops and depicts the state-space of the system in the form of a tree (DAG). 83

Figure 3.6. Markov chain graph (G_{mc}) with $\mathbf{n}_J$ nodes carrying $\approx\mathbf{X}_J$ states and the arcs showing transitions between them while, together, they form a Markov process. 88

Figure 3.7. Equivalent tree $\mathbb{HX}$ of G_{mc} (see Figure 3.6) as DAG representing the state-space of the system. 92

Figure 3.8. Limits of our visibility in the state-space before expansion, visualised by a Markov chain graph, where 🟢 is the initial node and 🟠 nodes are directly reachable nodes from the initial node when exactly one R_M occurs. When a further R_M occur, the system jumps to other ⚪ nodes. 95

Figure 3.9. A and B with copy counts 50 and 20, respectively, creating a system of two species, A and B with 70, total copy counts. 98

Figure 3.10. Two states X_1 of A and X_2 of B with almost equal propensities. This creates a dilemma when choosing the direction for expansion. 98

Figure 3.11. Current $state(N_2) = X_2$, and future $states(N_{3,4,5}) = (X_{3,4,5}, d_{l=3})$ with corresponding reactions R_1, R_2, R_5 and assumed propensities $a_{2,3} = 38, a_{2,4} = 39, a_{2,5} = 40$, respectively, at any time t, given $b_{0,2} = 0.4871, b_{6,2} = 0.5128$. 100

Figure 3.12. Node N as a junction of forward and backward reactions R_M, where $a'_{1,N}$,....,$a'_{N,N}$, are propensities of the prior reactions' and $b'_{1,N}$,....,$b'_{N',N}$ are the likelihood of the prior reactions. 100

Figure 4.1. Comprehensive ISP method flow chart. A description of the modules (steps), sub-modules and the list of derived components considered in ISP are provided briefly in Table 4.1. 110

Figure 4.2. General framework of *2D pyramid* domain showing the increase in the size of the domain with the increase in state with an increase in the bounds. $Bound_{lower}(1)$ represents the initial condition, whereas $Bound_{upper}(Z)$ represents the final domain carrying explored set of states of the system..... 118

Figure 4.3. Infrastructure of the *Latitudinal Search* strategy, showing G_{mc}, the $queue$ and the domain. 121

Figure 4.4. Toy model network. Showing three $\tilde{\mathrm{N}} = 3$ species, C, A, T in a network defining reactions, as given in Eq. (85). 124

Figure 4.5. Six stages of state exploration based on a LAS search for the toy model, given an average transitioning factor of $\mathrm{T} = 2$. Fig (A) is the first stage that denote the initial node carrying initial state of the system, Fig (B) is the second stage that denote all the nodes with new node at $\mathrm{d}_l = 2$, Fig (C) is the third stage that denote all the nodes with new node at $\mathrm{d}_l = 3$, Fig (D) is the fourth stage that

denote all the nodes with new node at $\mathfrak{d}_l = 3$, Fig (E) is the fifth stage that denote all the nodes with new node at $\mathfrak{d}_l = 4$, Fig (F) is the sixth stage that denote all the nodes with new node at $\mathfrak{d}_l = 4$. 126

Figure 4.6. Based on the *LAS* strategy, the response shows how the size of the domain increases with addition of new states with time in the toy model. 133

Figure 4.7. Based on the *LAS* strategy, the response shows the total bunked probability during the approximation of the toy model. 133

Figure 4.8. Time-Space complexity of the *LAS* response shows the linear behaviour at different average transition factors and the depth of the end state. For cases like $(\mathfrak{T} = 2, \mathfrak{d}_l = 4)$, and $(\mathfrak{T} = 5, \mathfrak{d}_l = 10)$, *LAS* has the same response. 135

Figure 4.9. Shows the set of states explored for the toy model after every *LAS* iteration. Based on the reactions, *LAS* unfolds the state-space pattern to update states in the domain and expands 66 probable states in 1.0 *sec*. ★ shows the time point where new set of states is explored and updated in the domain. 135

Figure 4.10. Catalytic reaction network having five $\tilde{\mathrm{N}} = 5$ species S, B, C, P, and E in a network defining reactions, as given in Eq. (88). 137

Figure 4.11. Expansion and updation of the states for the catalytic reaction system based on the *LAS* method. The state-space expansion increases the number of addition of new states in the domain. The size and colour of ▲ shows the increase in the size of the domain with the states population. 139

Figure 4.12. Shows the set of states explored for the catalytic system after every *LAS* iteration. Based on network of reactions, *LAS* unfolds the state-space pattern to update the states in the domain and expands 14666 probable states in 0.5 *sec*. ★ shows the time point where new sets of states are explored and updated in the domain. 141

Figure 4.13. Conditional probability of the catalytic system evaluated at $t_f = 0.5$ *sec*, $t_{step} = 0.01$ using *LAS*. Fig (A) is the probability of the species S over t_f, Fig (B) is the probability of the species B over t_f, Fig (C) is the probability of the species C over t_f, Fig (D) is the probability of the species P over t_f, Fig (E) is the probability of the species E over t_f. 143

Figure 4.14. Total probability of states bunked at t' from the domain of catalytic system produced by *ISP LAS* iteration while expansion and solving the CME. 145

Figure 4.15. Infrastructure of the *Longitudinal Latitudinal Search* strategy, showing the G_{mc}, the $stack$ and the domain. 147

Figure 4.16. Six stages of state exploration based on *LOLAS* search for the toy model, given average transitioning factor $\mathfrak{T} = 2$, assumed $\mathfrak{d}_{step} = 1$, $\mathfrak{b}_{limit} = 3$. Fig (A) is the first stage that denote the initial node carrying initial state of the system, Fig (B) is the second stage that denote all the nodes with new node at $\mathfrak{d}_l = 2$, Fig (C) is the third stage that denote all the nodes with new node at $\mathfrak{d}_l = 3$, Fig (D) is the fourth stage that denote all the nodes with new node at

$\mathfrak{d}_l = 4$, Fig (E) is the fifth stage that denote all the nodes with new node at $\mathfrak{d}_l = 4$, Fig (F) is the sixth stage that denote all the nodes with new node at $\mathfrak{d}_l = 4$. 152

Figure 4.17. Based on $LOLAS$ strategy, the response shows how the size of the domain increases with the addition of new states with time in the toy model. 159

Figure 4.18. Based on $LOLAS$ strategy, the response shows the total bunked probability during the approximation of the toy model. 159

Figure 4.19. Shows the set of states explored for the toy model after every $LOLAS$ iteration. Based on the reactions, $LOLAS$ unfolds the state-space pattern to update states in the domain and expands 66 probable states in 1.0 *sec*. ★ shows the time point where new set of states is explored and updated in the domain. 161

Figure 4.20. Coupled enzymatic reactions network. Showing six $\tilde{\mathrm{N}} = 6$ species, S, E_1, C_1, P, E_2, C_2, in a network defining reactions, as given in Eqs. (95) and (96)..... 162

Figure 4.21. Expansion and updation of states for the dual enzymatic reaction network based on the $LOLAS$ method. The state-space expansion increases the number of additions of new states in the domain. The size and colour of ▲ shows the increase in size of the domain with the states' population. 165

Figure 4.22. Shows the set of states explored for the dual enzymatic reaction network after every $LOLAS$ iteration. Based on the network of reactions, $LOLAS$ unfolds the state-space pattern to update states in the domain and expands 8296 probable states in 2.0 *sec*. ★ shows the time point where new set of states is explored and updated in the domain. 165

Figure 4.23. Conditional probability of the dual enzymatic reactions system evaluated at $t_f = 2.0$ *sec*, $t_{step} = 0.01$ using $LOLAS$. Fig (A) is the probability of the species S over t_f, Fig (B) is the probability of the species B over t_f, Fig (C) is the probability of the species C over t_f, Fig (D) is the probability of the species P over t_f, Fig (E) is the probability of the species E over t_f. 167

Figure 4.24. Total probability of states bunked at t' from the domain produced by dual enzymatic reactions system in $ISP\ LOLAS$ iteration while expansion and solving the CME 169

Figure 4.25. Routines of operation of ISP modules. The worst-case complexity is considered for LAS and $LOLAS$ algorithm for the upper limit of the execution. 170

Figure 5.1. The comparison of ISP (LAS and $LOLAS$) with $OFSP$ based on the solution of the catalytic reaction system. 176

Figure 5.2. The comparison of ISP (LAS and $LOLAS$) with $OFSP$ and SSA by computational time. All methods were applied to the catalytic reaction system that was previously integrated in section 4.3.2. 176

Figure 5.3. AWS® CPU utilisation percentage, when the catalytic reaction system is solved up to $t_f = 0.5$ *sec* using $OFSP$ and $LOLAS$. The performance analysis was carried out using CloudWatch® (Statistic: Average, Time Range: Hour, Period: 5 Minutes). 177

Figure 5.4. Comparison of the ISP (LAS and $LOLAS$) with $OFSP$ based on the solution of the dual enzymatic reaction system. 179

Figure 5.5. The comparison of ISP (LAS and $LOLAS$) with $OFSP$ and SSA by computational time. All the methods were applied to the dual enzymatic reaction system previously integrated in section 4.4.2. 181

Figure 5.6. AWS® CPU utilisation percentage, when dual enzymatic reaction system is solved up to $t_f = 2.0$ *sec* using $OFSP$ and $LOLAS$. The performance analysis is done using CloudWatch® (Statistic: Average, Time Range: Hour, Period: 5 Minutes). 182

Figure 6.1. 2D structure of the G1/S checkpoint model involving the DNA damage signal transduction pathway using model variables for $ISP\ LOLAS$. 190

Figure 6.2. Markov chain graph given ● as the initial node of the G1/S checkpoint involving the DNA damage signal transduction pathway model. The ● nodes are directly reachable nodes from the initial state when exactly one R_M occurs. The ● nodes are reachable from the initial state when exactly two R_M occurs. When R_M occurs {3,4…….} times the system jumps to the ● nodes, respectively. 196

Figure 6.3. Showing the expansion and updating of the states for the G1/S model based on the $ISP\ LOLAS$ method. The state-space expansion increases the number of additions of new states in the domain. $ISP\ LOLAS$ quickly expands the state-space up to ≈3.5 million states in 1.5 *sec*. 200

Figure 6.4. Response of the $ISP\ LOLAS$ checkpoint for examining the G1/S checkpoint involving the DNA-damage signal transduction pathway's initial state probability over time. 202

Figure 6.5. The set of states explored for G1/S model using the $ISP\ LOLAS$ method. Based on network of reactions, $ISP\ LOLAS$ unfolds the state-space pattern to update states in the domain and expands 3409899 states up to t_f. ★ shows the time point where new set of states is explored and updated in the domain. 204

Figure 6.6. Total probability bunked at t' from the domain in $ISP\ LOLAS$ iteration while expansion and solving the CME for G1/S model. 206

Figure 6.7. Conditional probability of the G1/S model evaluated at $t_f = 1.5$ *sec*, $t_{step} = 0.1$ using $ISP\ LOLAS$. 211

Figure 7.1. Network of reactions involving 45 species, 6 auxiliary variables of the model of the adaptation to oxidative stress response in *C. albicans* (fungal pathogen) pathway using model variables for $ISP\ LAS$ 224

Figure 7.2. Expansion and updating trend of the states for *transporter module* in the integrative model of the fungal pathogen, *C. albicans*, based on the *ISP LAS* method. *ISP LAS* quickly expands the state-space up to 78054 states for 3.01 *sec* for *transporter module*. The state-space expansion of *transporter module* contributes in increasing the number of additions of new states to the domain. 246

Figure 7.3. Expansion and updating trend of the states for *catalase* (antioxidant) in the integrative model of the fungal pathogen, *C. albicans*, based on the *ISP LAS* method. *ISP LAS* quickly expands the state-space up to 38235 states for 3.01 *sec* for *catalase*. The state-space expansion of *catalase* contributes in increasing the number of additions of new states to the domain. 246

Figure 7.4. Expansion and updating trend of the states for *pentose phosphate pathway* in the integrative model of the fungal pathogen, *C. albicans*, based on the *ISP LAS* method. *ISP LAS* quickly expands the state-space up to 20097 states for 5.1 *sec* for *pentose phosphate pathway*. The state-space expansion of *pentose phosphate pathway* contributes in increasing the number of additions of new states to the domain. 247

Figure 7.5. Expansion and updating trend of the states for *thioredoxin* (antioxidant) in the integrative model of the fungal pathogen, *C. albicans*, based on the *ISP LAS* method. *ISP LAS* quickly expands the state-space up to 6641483 states for 0.00062 *sec* for *thioredoxin* (antioxidant). The state-space expansion of *thioredoxin* (antioxidant) contributes in increasing the number of additions of new states to the domain. 249

Figure 7.6. Expansion and updating trend of the states for *protein mono and di-thiols* in the integrative model of the fungal pathogen, *C. albicans*, based on the *ISP LAS* method. *ISP LAS* quickly expands the state-space up to 245701 states for 0.040 *sec* for *protein mono and di-thiols*. The state-space expansion of *protein mono and di-thiols* contributes in increasing the number of additions of new states to the domain. 249

Figure 7.7. Expansion and updating trend of the states for *cap1 pathway* in the integrative model of the fungal pathogen, *C. albicans*, based on the *ISP LAS* method. *ISP LAS* quickly expands the state-space up to 2367590 states for 8.46 *sec* for *cap1 pathway*. The state-space expansion of *cap1 pathway* contributes in increasing the number of additions of new states to the domain. 251

Figure 7.8. Expansion and updating trend of the states for *hog1 pathway* in the integrative model of the fungal pathogen, *C. albicans*, based on the *ISP LAS* method. *ISP LAS* quickly expands the state-space up to 15980 states for 200.0 *sec* for *hog1 pathway*. The state-space expansion of *hog1 pathway* contributes in increasing the number of additions of new states to the domain. 251

Figure 7.9. Response of the *ISP LAS* checkpoint for examining the integrative model of fungal pathogen *C. albicans* module's (Figs (1) for *transporter module's*, Figs (2) for *catalase's*, Figs (3) for *thioredoxin*, Figs (4) for *protein mono and di-thiol*, Figs (5) for *cap1 pathway's*, Figs (6) for *hog1 pathway's*), initial state and any random state probability over time t_f. 254

Figure 7.10. The set of states explored for the *transporter module* using the *ISP LAS* algorithm. Based on network of reactions, *ISP LAS* unfolds the state-space pattern to update states in the domain and expands 78054 states up to t_f. ★ shows the time point where the new set of states are explored and updated in the domain. 256

Figure 7.11. The set of states explored for the *cat1 pathway* using *ISP LAS* algorithm. Based on network of reactions, *ISP LAS* unfolds the state-space pattern to update the states in the domain and expands 38235 states up to t_f. ★ shows the time point where the new set of states is explored and updated in the domain. 258

Figure 7.12. The set of states explored for the *pentose pathway* using the *ISP LAS* algorithm. Based on network of reactions, *ISP LAS* unfolds the state-space pattern to update the states in the domain and expands 20097 states up to t_f. ★ shows the time point where the new set of states are explored and updated in the domain. 260

Figure 7.13. The set of states explored for *thioredoxin* using the *ISP LAS* algorithm. Based on network of reactions, *ISP LAS* unfolds the state-space pattern to update states in the domain and expands 6641483 states up to t_f. ★ shows the time point where the new set of states are explored and updated in the domain.... 262

Figure 7.14. The set of states explored for *protein-thiol* using the *ISP LAS* algorithm. Based on network of reactions, *ISP LAS* unfolds the state-space pattern to update states in the domain and expands 245701 states up to t_f. ★ shows the time point where the new set of states are explored and updated in the domain. 264

Figure 7.15. The set of states explored for *cap1 pathway* using the *ISP LAS* algorithm. Based on network of reactions, *ISP LAS* unfolds the state-space pattern to update states in the domain and expands 2367590 states up to t_f. ★ shows the time point where new set of states are explored and updated in the domain.. 266

Figure 7.16. The set of states explored for the *hog1 pathway* using the *ISP LAS* method. Based on network of reactions, *ISP LAS* unfolds the state-space pattern to update states in the domain and expands 15980 states up to t_f. ★ shows the time point where the new set of states are explored and updated in the domain. 268

Figure 7.17. Conditional probability of the species x_0 (extra-cellular H_2O_2), x_1 (intra-cellular H_2O_2) evaluated at $t = 0.2$ *sec*, 1.2 *sec* and 3.0 *sec* with $t_{step} = 0.01$ using *ISP LAS*..... 272

Figure 7.18. Conditional probability of the species x_2 ($Cat1$), x_{33} ($CAT1$ or $CAT1\ mRNA$) evaluated at $t = 0.2$ *sec*, 1.2 *sec* and 3.0 *sec* with $t_{step} = 0.01$ using *ISP LAS*. 272

Figure 7.19. Conditional probability of the species x_{20}, x_{41} evaluated at $t = 1.5$ *sec*, 3.2 *sec* and 5.1 *sec* with $t_{step} = 0.01$ using *ISP LAS*. 274

Figure 7.20. Conditional probabilities of species x_{12}, x_{14}, x_{15}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}, x_{37}, x_{38}, x_{39} evaluated at $t = 0.00028$ *sec*, 0.00042 *sec* and 0.00062 *sec* with $t_{step} = 0.0001$ using *ISP LAS*. 275

Figure 7.21. Conditional probability of the species x_9, x_{12} evaluated at $t = 0.014$ *sec*, 0.028 *sec* and 0.040 *sec* with $t_{step} = 0.01$ using *ISP LAS*. 277

Figure 7.22. Conditional probability of the species x_{21}, x_{22}, x_{32}, x_{34}, x_{35}, x_{36}, x_{37}, x_{38}, x_{39}, x_{40}, x_{41}, evaluated at $t = 3.01$ *sec*, 7.01 *sec* and 8.46 *sec* with $t_{step} = 0.01$ using *ISP LAS*. 278

Figure 7.23. Conditional probability of the species x_{24}, x_{26}, x_{25}, x_{27}, x_{28}, x_{29}, x_{30}, x_{31}, x_{42}, x_{43}, x_{44}, evaluated at $t = 25.0$ *sec*, 50.0 *sec* and 200.0 *sec* with $t_{step} = 1.0$ using *ISP LAS*. 281

Abbreviations

Terminology

	A
AD	Alzheimer's Disease
AI	Artificial Intelligence
Algo	Algorithm
AWS	Amazon Web Services
	B
BLNP	Bayesian Likelihood Node Projection
	C
CK	Chapman-Kolmogorov Equation
CME	Chemical Master Equation
CPU	Central Processing Unit
CSR	Compressed Row Format
CTMC	Continuous Time Markov Chain
CTHMC	Continuous Time Homogeneous Markov Chain
	D
DDR3	Double Data Rate Version 3
DAGs	Directed Acyclic Graphs
	E
EBS	Elastic Block Storage
EC2	Elastic Computing Version 2
ECUs	Electronic Controlling Units
EFS	Elastic File System
	F
FIFO	First In, First Out
FSP	Finite State Projection
	G
GP2	General Purpose Version 2 (SSD)
GORDE	Gated One Reaction Domain Expansion
	H
HVM	Hardware Virtual Environment
	I

IOPS	Input Output per second
ISP	Intelligent State Projection Method
	L
LAS	Latitudinal Search
LIFO	Last In, First Out
LOLAS	Longitudinal Latitudinal Search
	M
ML	Machine Learning
	O
ODE or ODEs	Ordinary Differential Equation(s)
OFSP	Optimal Finite State Projection
	Q
Queue	Defines the temporary line-up of nodes for any iteration in LAS
	S
SSA	Stochastic Simulation Algorithm
SSD	Solid State Drive
Stack	Defines the temporary storage of nodes for any iteration in LOLAS
SW	Sliding Windows Method
	U
URN	Uniform Random Number Generator

Notations

$a_{i,j}$ or a_{μ}	Propensity of chemical reaction
$\Delta a_{i,j}$	Change in propensity
$a'_{1,N'},\ldots,a'_{N,N'}$	Propensities of the prior reactions
a_{fr}	Probability of a jump process from state X_{i-1} to X_i per unit time
a_{rv}	Probability of a jump process from state X_i to X_{i-1} per unit time
A or $A_{i,j}$	Defines the transition between i, j and its weightage
$\mathfrak{b}_{limit}$	Exploration bound limit in $LOLAS$
$b'_{N',N}(X - v_{\mu})'$	Prior Bayesian likelihood values $\{b'_{1,N}, \ldots \ldots b'_{N',N}\}$
$b(N_{N1,..NM}\|b'_{1,N....N',N})$	Represents Bayesian likelihood value given prior $b'_{N',N}$
$Bound_{lower}$ or $Bound_L$	Define the set of states $\{X_{1,2.....S}, b_{1,2,3,....limit}\}$ at $\mathfrak{b}_{limit}$ already present in the domain for current iteration
$Bound_{upper}$ or $Bound_U$	Define the set of states $\{X_{1,2.....S}, b_{1,2,3,....limit}\}$ at $\mathfrak{b}_{limit}$ added in the domain at the end of current iteration
$\boldsymbol{c}, c_1, c_2$	Constants
$\mathfrak{c}_{N_i,N'_i}$	Total transition/walk cost from node N_i to N'_i
$Dict$	Dictionary of the model having transition records
$\mathfrak{d}_l$	Exploration depth limit in LAS
$\mathfrak{d}_{step}$	Exploration depth step in ISP
dim	Dimension of sub-matrix in Sliding Windows Method
$domain$	Defines the set of states of domain in current iteration that forms $Bound_{upper}$
$domain_{previous}$	Defines the set of states of domain in previous iteration that forms $Bound_{lower}$
D_j	Diagonal matrix whose diagonal entries are one
e	Markov chain tree edge representing walk from N_i to N'_i
e_1	First unit basis vector in Krylov Subspace Method
e_{rror}	Represents error value in calculation
$exp()$	Exponential function
eps	Epsilon
$\acute{\mathrm{E}}_y$	Denote the sequence of events $\acute{\mathrm{E}}_1$, $\acute{\mathrm{E}}_2$….,
$f(y)$	Represent the positive real value function of y
G_{mc}	Represents graph associated with Markov chain tree
$\bar{H}_{dim}$	Upper matrix (Hessenberg Matrix)
I^T	Identity matrix $I = diag(1,1, \ldots.1)^T$
I_{tr}	Denote the iterations in ISP
k_M	Kinetic parameter of the chemical reaction where $M = \{1,2 \ldots. \infty\}$

l	Used as subscript for Length of depth, for example $\mathfrak{d}_l$
$\mathbf{n}_J$	Set of node as $\{N_1, N_2, \dots \dots . N_{S^{\tilde{\mathrm{N}}}}\}$
$\mathbf{n}_K$	Set of nodes carrying set of $\mathbf{X}_K$ at any iteration
$\mathbf{n}'_K$	Set of nodes carrying set of $\mathbf{X}'_K$ at any iteration
N_0	Root node carrying initial state X_0
N_i or N_i	Any node
$\tilde{\mathrm{N}}$ or $\{S_1, \dots \dots . S_{\tilde{\mathrm{N}}}\}$	Number of different species
num_1, num_2	Random number generated by uniform random number generator (URN)
$P^{(t_0)}(X_0)$	Initial probability at $t = 0$
$P^{(t)}(\mathbf{X}_K)$	Probability of set of states at time t
$P_{N,N'}(\omega)$	Weighted probability of transition from N_i to N'_i
R_M	M elementary chemical reaction channels $\{R_1, R_2 \dots \dots . R_M\}$
R'_M	Prior M elementary chemical reaction channels $\{R'_1, R'_2 \dots \dots . R'_M\}$
$R_{M(fs)}$	M elementary chemical reaction channels of fast reactions
$R_{M(sr)}$	M elementary chemical reaction channels of fast reactions
$\mathbb{R}_{tract}$	Number of retraction in $LOLAS$
$S^{\tilde{\mathrm{N}}}$	Approximate number of states
$\hat{\mathrm{S}}$	Number of stages in expansion $\{1,2, \dots \dots .\}$
S_{uc}	Implicit successor or operator
$sqrt$	Square root
t_0	Time at which initial conditions of system are defined
t'	Time at which $\mathbf{X}'_K$ is dropped from the domain
t	Any random time in seconds
t_d	Time at which $\mathbf{X}_K$ is updated in the iteration
t_f	Final time at which solution is required
$\mathbb{T}$	Transitioning factor
$U_{X_i,X_{i'}}$	Set of all arborescences
$\lvert U \rvert$	Define the cardinality of any set
v	Krylov Sub-space method - A column vector of a length equal to the number of different species present in the system
v_μ or v_M	Stoichiometric vector represents the change in the molecular population of the chemical species by the occurrence of one R_M reaction. It also defines the transition from state X_i to $X_{i'}$ in Markov chain tree.
$v_\mu(X(t))$ or $v_M(X(t))$	Stoichiometric vector function, where X is any random state
V_μ or V_M	Matrix of all the Stoichiometric vectors $[v_1; v_2; \dots \dots v_\mu]$

W^{y}	Probability that is computed inductively by $W^{(0)} = P^{(0)}$ in uniformisation method
$x_1, \ldots \ldots x_{\tilde{\mathrm{N}}}$	Number of counts of different species
X or X_i or $X_{i'}$	Any random state
X_0	Initial state or Initial condition
$\mathbf{X}_J$	ordered set of possible states $\{X_1, \ldots \ldots X_{S^{\tilde{\mathrm{N}}}}\}$ of the system
$\mathbf{X}_K$	Set of new states or domain at any iteration
$\mathbf{X}'_K$	Set of states dropped from domain at t' at any iteration
y, y_0	Positive integers
Y_y	Poisson process given that $0 < y \leq M$
Z	Number of bounds in ISP
τ_m or tol_m	Tolerance value
$\tau_m(leak)$	$P^{(t)}(\mathbf{X}'_K)$ leakage point
Æ	Approximate solution of the CME
ω	Weight or cost of single transition from X_i to $X_{i'}$. It is equivalent to $a_{i,j}$
$\mathbb{X}_c$	Markov chain representing biochemical process
$\mathbb{HX}$	Markov chain tree with $\mathbf{n}_J$
λt	Uniformisation rate
$\mathfrak{v}_j$	Number of nonzero elements in P_j
φ	Sample space
Ω	Asymptotic lower bound
O	Asymptotic upper bound
Θ	Asymptotic tight bound
$\{1,2 \ldots \ldots K\}$	Indexing of set of states and set of nodes
$\mu = \{1,2, \ldots . M\}$	Channels of chemical reaction propensity

Key Factors and Outputs

Dimension	Indicating factor for knowing the difficulty level for solving the CME
Run-time	Time taken by the algorithm to expand the state-space and approximate the solution of CME
Number of states	Total number of states at any time
Approximation Error	Total error when states were simultaneously expanded and bunked
Probabilities	Conditional probabilities of the species

Page intentionally left blank

Chapter 1
Introduction

In systems biology, it is of great interest to understand the dynamics of large and complicated biochemical reaction networks based on recent advances in computing and mathematical techniques. These advances in techniques have made it easier for biologist to deal with enormous amounts of experimental data down to the level of a single molecule of a species. Such information reveals the presence of a high level of stochasticity in the networks of biochemical reactions. The establishment of stochasticity in biochemical reaction networks has made a significant contribution towards the field of cells (Kholodenko et al., 2000; Roberts et al., 2004), neuroscience (Ozer et al., 2009), and drug modelling (Murray et al., 2009), for example. For certain problems, the minimal domain grows polynomial in time; however, in an application, if the domain is not constructed adaptively by default, hyper-rectangular support is established, which grows exponentially in high dimension biochemical systems. An example of diversity in biochemical systems is the discrepancy from species to species and from reaction to reaction in the biochemical networks of Alzheimer's disease (AD) (Hogervorst et al., 2003) where the behaviour of different pathogens in pathways is still largely unknown; and the biochemical networks of pathways in the fungal pathogen *Candida albicans* (Schulze et al., 2009).

Biochemical reaction networks are widely used to describe typical biological processes within systems biology using deterministic models; however, deterministic models do not provide probabilistic descriptions of such systems. These biochemical processes can be captured by incoporating stochasticity into the biological networks via stochastic mathematical models. To do this, one approach to model a biochemical reaction network is to deduce a set of integro differential equations known as chemical master equations (CMEs) (Gillespie, 1977, 1992a). CMEs describe the evolution of the probability distribution over the entire state-space of a biochemical system that jumps from one set of states to other in continuous time; they are a continuous time version of Markov chains (CTMCs) (Gillespie, 1992a; Weber, 2012) with discrete states. By defining the Markov chain (Goutsias et al., 2013; Weber, 2012), we can consider the probability density associated with a system that changes over time.

The distribution obtained from the solution of CMEs is very useful for scientists. For example, they can study the experimental data to amend the parameters underlying large

biochemical models or achieve a detailed perception about their molecular composition before proceeding to an expensive trial and error investigation programme. Accordingly, they can use the solution of CMEs as a reference guide for new investigations by solving them in a future time point where no investigations have been undertaken (Munsky et al., 2009; Shepherd et al., 2013).

Despite these applications, solving CMEs is a formidable task due the nonlinear nature of the reactions and size of the networks that result in different realisations and, most importantly, the exponential growth of the size of the state-space with respect to the number of different species in the system. When the size of the biochemical system is very large in terms of the number of variables, the solution to the CME quickly becomes intractable. This is due to an explosion of the state-space, which is considered to be the curse of systems' dimensions. Although, CMEs have been employed and solved explicitly by existing algorithms for relatively small biological systems (Burrage et al., 2006; Dinh et al., 2017; MacNamara et al., 2008; Sidje et al., 2015; Sunkara et al., 2010; Wolf et al., 2010) computationally complaisant but accurate and efficient solutions are still unknown for most significant systems, particularly large biochemical systems. These are untouched when it comes to the expansion of state-space and solutions of CME. Also, the nature of information generated by the exisiting algorithms and computational platforms are also not sufficient to deeply understand and visualise the state-space and the process of domain formation.

The aforementioned highlights the challenges and importance of exploring the state-space and having an efficient domain size for solving CMEs and the timeliness of our study. This study aims to solve the high demand for state-space expansion problems efficiently, particularly in large and integrated biochemical systems, by incorporating fresh ideas in applied computing and computational mathematics.

1.1 Stochastic Chemical Kinetics

The great significance and application of reaction rate equations ($RREs$), which are deterministic ordinary differential equations ($ODEs$) in chemical kinetics, cannot be challenged. This approach presumes that the time for the evolution of biochemical reaction networks is both deterministic and continuous because the species' molecules are large in numbers and justify continuous concentrations but some species' molecules may be present in either very small or very large numbers and their nonlinearity is not smoothed by statistical averaging. This contrasts, with the fact that molecule count levels apparently change only in a discrete fashion; moreover, when the time of their evolution is considered, it is not a determinstic process either.

In such cases, the inability to define the nonlinearity at the molecule count level in $RREs$ can become very important, so mean values are measured by taking a set of possible values for probability distribution. Under the same configuration as for time evolution, the biochemical reaction network can then be defined in terms of the discrete state $X \equiv (x_1, \dots \dots . x_{\tilde{\text{N}}})^T$ vector of non-negative integers $x_{\tilde{\text{N}}}$ given the initial conditions.

Definition 1.1. The stochastic process is the time evolution of a collection of random variables, such that $\{X(t) : t \in K, \varphi\}$ for a single variable and where K is some indexing scheme and φ is a sample space. For each fixed t, $X(t)$ indicates one random variable defined in φ and correlated with a function defined for K and this is what we define as process *stochastic realisation*. When we define a collection of more than one random variable then $\{X_i(t),\ X_{i'}(t) : t \in K, \varphi\}$, where X_i and $X_{i'}$ are two variables in sample space φ.

Definition 1.2. The stochastic process $\{X(t) : t \in K, \varphi\}$ is said to be a *jump process* that defines the jumps between different states in φ or the state-space with time.

The focus of our study is only on stochastic processes of the *continuous* type; therefore, from here on the stochastic processes are considered to be *continuous time* stochastic processes fixed to $[0, \infty)$, based on our indexing scheme. The biochemical reactions are triggered based on the process of counting the species, then the change in molecular counts of the species is given by a *jump process* in the Markov process (discussed further in Chapter 3), which is basically a stochastic model based on Markov properties. For our problem domain, a *continuous time jump process* is a jump in the Markov process that is defined as a continuous time Markov chain (CTMC) with discrete states. The biochemical systems we are interested in are the systems whose changes in molecular population of species are defined by CTMC

discrete states (discussed further in Chapter 3). The process to describe the time evolution of the randomness of variable X is the chemical master equation (CME) (as discussed in Chapter 2).

Based on the derivation of Gillespie (1977), for every reaction, R, there exists a reaction channel such that R_M determines the unique reaction in the system with constant k_M. The specific combinations of the reactant species, R_M, will react during an infinitesimal $[t, t + dt)$ time interval. Therefore, the average probability $a_{\mu}(X(t))dt$ at which a particular R_M will fire in $[t, t + dt)$ is the combination of the total number of reactant species times k_M, called the propensity function. For example,

$$\begin{aligned} &\boldsymbol{R_1}\colon\ C \overset{k_1}{\rightarrow} A, && a_1\colon\ k_1([C]) \\ &\boldsymbol{R_2}\colon\ A + E \overset{k_2}{\rightarrow} T, && a_2\colon\ k_2([A].[E]) \\ &\boldsymbol{R_3}\colon\ T \overset{k_3}{\rightarrow} C, && a_3\colon\ k_3([T]) \end{aligned}$$

In the case when the reactants are of the same type; for example $A + A \overset{k_2}{\rightarrow} T$, then $a_2\colon\ k_2(\frac{A(A-1)}{2})$, where $\frac{A(A-1)}{2}$ denotes the individual pair of species A molecules.

Definition 1.3. The total number of reactions, $R_{M,}$ present in the biochemical reaction network is the union of sets of *fast* reactions and *slow* reactions, and they are categorised into sets of $R_{M(fs)}$ and $R_{M(sr)}$ reactions, respectively, based on their propensity values. Therefore,

$$R_M = R_{M(fs)} \bigcup R_{M(sr)}. \tag{1}$$

The reaction is considered to be faster than others if its propensity is of several orders of magnitude larger than the other reactions' propensity values.

1.2 Motivation and Main Research Questions

The difficulty of solving the CMEs (see Chapter 2) is exacerbated when the dimension of the model grows with the number of distinct species, and the size of a model with the number of states. The size of the model shows the complexity of the state-space and the 'curse' of system's dimension. Under computational experimental conditions using existing algorithms, the results from small biochemical reaction networks are being studied. However, they expand the state-space quickly so solving the CME with accuracy is still a challenging task for a significant number of biochemical reaction networks, especially large ones (Komalapriya et al., 2015; Ling et al., 2010). It is computationally expensive and challenging to study the complex behaviour of large biochemical reaction networks because of the limitations and behaviour of the existing algorithms. Controversial results of small biochemical sytems from different research groups may be due to the different experimental materials or the methods used. This undeveloped area provides great opportunities to study this kind of problem in systems biology, using mathematical modelling and recent computing advancements. Therefore, deep insights can be obtained by properly interpreting the state-space of a biochemical system from thoughtfully designed *de novo* numerical computation algorithms. Through the experiments in this thesis, we aim to address the following questions about state-space, accuracy and computational time using the numerical algorithm we will develop in our study:

(1) How would the state-space be visualised efficiently to understand its complexity? How would states be searched and state-space expanded for small and large biochemical systems? What would be the size of the domain created at any one time? What would be the state-space blueprint of the model?

To answer these questions, we need to select key changes that have previously been suggested in the data structure of computer science to link them to the state-space. Based on that, a mathematical model based on the data's structure then needs to be developed to represent the state-space, according to published experimental observations. The state-space of biochemical systems should also be able to adopt selected changes after including the relevant sophistication of the state-space structure.

(2) How can potential new states in the state-space be identified? What would be the direction of the state-space search? What would be the accuracy of the solution based on the size of domain created by efficient searching for states in the state-space at different time points? How would domain produed affect the overall solution of the CME?

To answer these questions, we first need to develop an innovative method that treats state-space as a sophisticated data structure. This can be achieved by putting new modules into suitable step locations to consider the space, apply the expansion technique and capture the exact size of the domain at particular time points. Certain parameters should then be calibrated or estimated according to published experimental observations. This setting will then be used to test the domain solution at different time points and the overall solution of the CME. The simulation results should provide information in terms of the state-space patterns, errors at different time points and predicitions based on conditional probabilities.

(3) How long does it take to expand the state-space of a large biochemical reaction network up to the desired time. How is the new algorithm better than other existing algorithms?

To answer these questions we need to choose and model large biochemical reaction networks based on the given parameters and variables. We then import the model into a new algorithm to investigate the effects of different reactions on the molecular population of the species, resulting in new states that lead to the expansion of the state-space. We will note the performance of the new algorithm in terms of computational time and relate the results to the key biological events leading to these complex formations wherever needed. Lastly, we will compare the new algorithm with the existing algorithms for accuracy and computational time.

1.3 Objectives for this study

The solution methods to CMEs can broadly be classified as combination of: (a) state-space expansion methods; and (b) approximation methods, as shown in Figure 1.1.

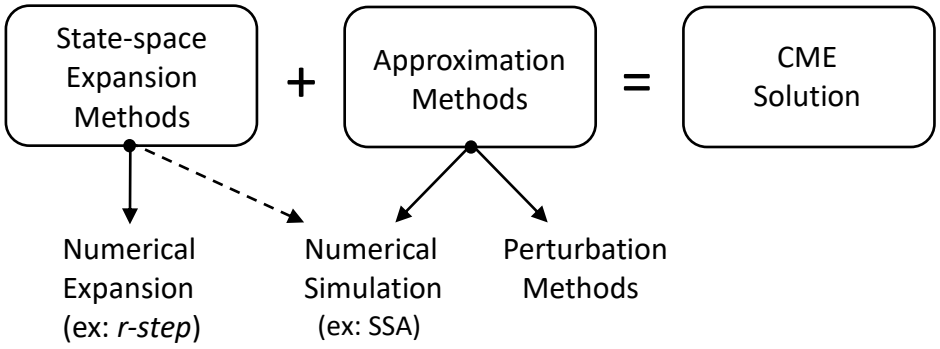


Figure 1.1. Solution methods to CME are a combination of state-space expansion methods and approximation (Æ) methods.

Significant improvements have already been made towards approximation methods (discussed in Chapter 2) to improve the solution of CMEs but far fewer for expansion methods. To achieve the best solution of CMEs, it is also important to use an efficient expansion strategy simultaneously before any efficient approximation method becomes feasible. Therefore, this study focuses on developing the expansion strategy to analyse the state-space of large biochemical reaction networks to create efficient domains for improved solutions of the CME in terms of accuracy and computational time. There are four major objectives in this study.

(1) To develop a data structure to represent and investigate the class of state-space growth in terms of its objects, i.e. number of states for biochemical reaction networks.

It is important to have a compatible data structure for the new algorithm to treat the model functions in the correct way. Therefore, we will use the Markov chain to extend our understanding of stochastic processes and interpret the formation of state-space by including the new components in the space.

(2) To develop a platform-independent numerical algorithm based on recent trends in computing compatible with the structure of the data (1^{st} objective) for the expansion of state-space suitable for significant types of large biochemical reaction networks.

We will develop a framework that accepts the new data structure for the characteristics of the vector form of the CME. This framework will integrate the parameters in the form of functions and track the major events of biochemical reaction networks from initialisation, to

the expansion of the state-space, to the formation of domains. Using mathematical modelling for the biochemical system, the new framework will mimic the functions of the model reactions to expand the state-space.

(3) To study the formation of domains at different time points during expansion to improve the accuracy of the solution while maintaining minimal computational expenses.

We will closely analyse the addition of new states for the formation of domains at different time points to understand their contributions towards the solution of the CME. We will use state patterns called blueprints to investigate the time points for the explosion of states that should define the firing of reactions in the network. We will also investigate the total time taken by the new algorithm to expand the states and check the accuracy of the domain produced at different time points through the CME.

(4) To study the application of the new algorithm on large biochemical reaction networks by mathematical modelling.

We will investigate the application of the new algorithm on large biochemical reaction networks for its performance and the accuracy of the solution. To do this, we will first discuss the model in brief and integrate it using mathematical modelling. The results from the new algorithm should reveal the number of states, size of the domain produced, the explosion of states in space, conditional probabilities of the species involved, and the solution of the CME at the desired time.

1.4 Thesis structure

This thesis consists of eight chapters. Figure 1.2 illustrates the organisation of the chapters and the main research questions to be answered in Chapters 3 and 5, respectively, followed by case studies using large biochemical systems, in Chapters 6 and 7.

In the current chapter, Chapter 1, we introduce stochastic chemical kinetics and its challenges for solving the CME for large biochemical reaction networks. We also discuss the motivations, main research questions that need to be addressed and, based on these, we discuss the objectives of this thesis in brief.

Chapter 2 reviews the background of the CME problem, its structure and experimental findings, including the latest publications. We also discuss, in detail, the existing numerical approaches that particularly focus on state-space expansion and its potential for understanding the underlying complexity in the state-space and its expansion (which is the primary focus of this thesis). Lastly, we will briefly discuss the approximation methods of the CME.

In Chapter 3, we discuss the definitions and preliminaries to create a data structure for representing the state-space. We then define the sample space and classify the type of states for Markov chains arising in biochemical reaction networks. We will then use this classification to layout the data structure to define the Markov chain as Markov chain graphs or trees to create a problem state-space model. Lastly, we will discuss the benefits of following the standards of artificial intelligence for the CME and develop an infrastructure for searching for problems in the state-space model. We will then develop a simple function to project the expansion strategy based on the new data structure.

In Chapter 4, extends the idea discussed in previous chapters by adding the conditions for the expansion of the state-space, and how the states are treated based on its probability mass. Based on these conditions, we will layout the infrastructure of our ISP algorithm (includes two variants – LAS and $LOLAS$) and discuss the step–by–step expansion stages on a toy model and how it can be applied to real biochemical reaction networks.

In Chapter 5, we compare our ISP algorithm with the results from existing algorithms based on – domain size, accuracy and computational time. We will also compare the depth of our understanding of the state-space governed by these algorithms.

In Chapter 6, we discuss the case study of the application of the $ISP\ LOLAS$ algorithm on the large biochemical reaction network at the G1/S checkpoint involving the DNA-damage signal

transduction pathway.

In Chapter 7, we discuss the second case study for the application of the $ISP\ LAS$ algorithm on a large integrated biochemical reaction network of pathways in the oxidative stress adaptation in the fungal pathogen, *Candida albicans*.

In final Chapter 8, we summarise the contribution and work of this thesis and discuss future directions that could be considered to extend the application of ISP, and also about solving the CME to understand the behaviour of the system itself.

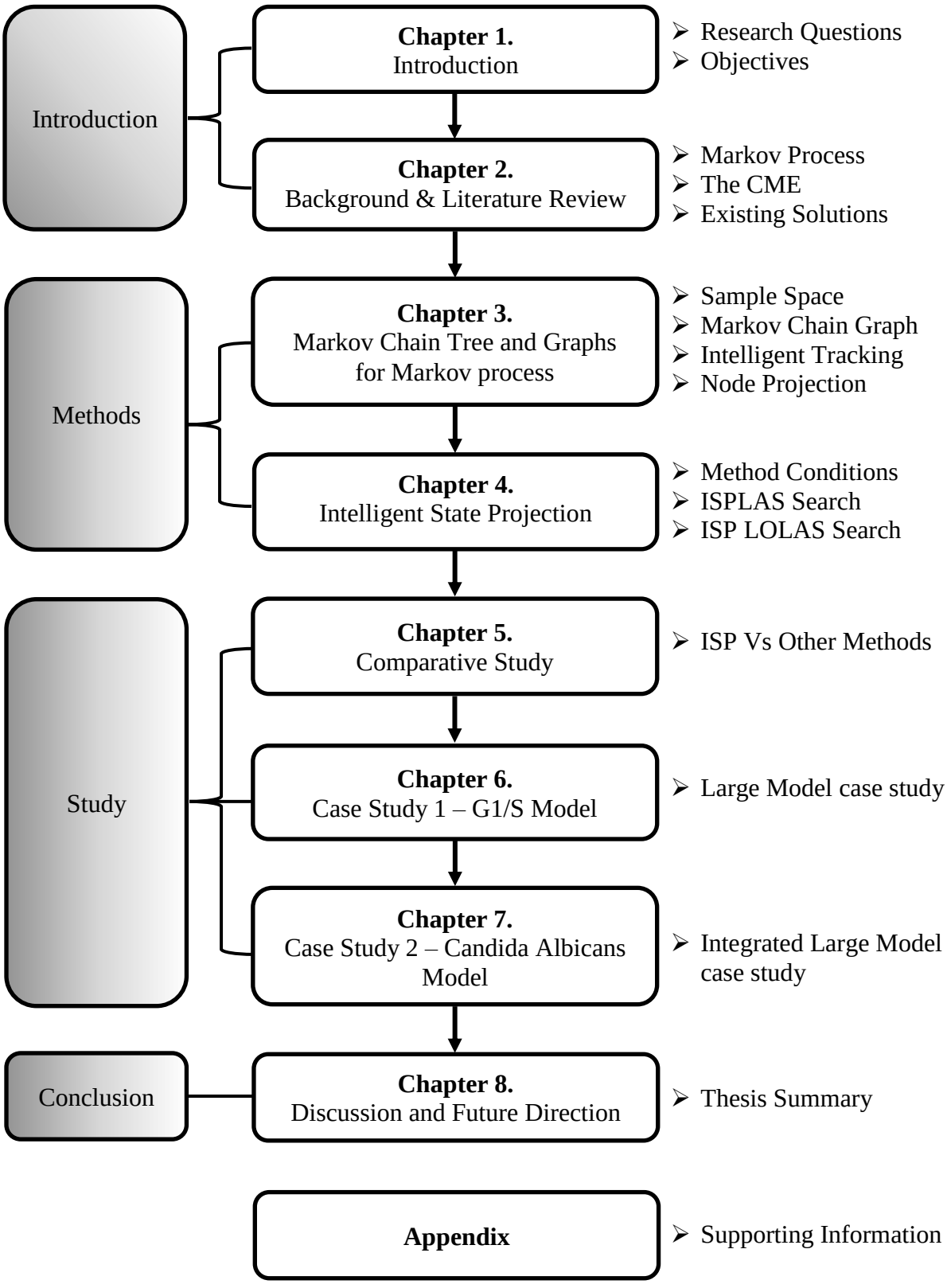


Figure 1.2. Flow chart showing organisation of thesis and major research points in the form of chapters.

Chapter 2
Background and Literature Review

The Master equation is a first-order integro-differential equation, which relates any function with the derivatives (function value or dependent variable). The function usually represents the physical or chemical quantities, while the derivative represents the changes with respect to time. Collectively, the Master equation defines the relation between these two. Master equation play a vital role in many disciplines, including chemistry, physics, biology, engineering because such relationships are very common in today's systems. The stochastic behaviour in biochemical reaction networks was coined by McQuarrie (McQuarrie, 1967) and, later, by Gillespie (Gillespie, 1992b), which came up with rigorous derivation of Chemical Master Equation (CME) to accommodate the intrinsic stochastic behaviour of a chemical system. It is important to note that according to the Master equation, the dimension of a problem can be classified into 'D' dimensions and network systems. In our study we only deal with the CME, which outlines the time evolution of a biochemical reaction network that can be modelled exactly for a given discrete set of states. The most favoured form of CME is written in the matrix form, which depicts the connections between the different states, and the way connections are made between the states decides the complexity of the state-space.

In section 2.1, we first introduce the Markov process and the probability definition. In secton 2.2, we introduce the Chemical Master Equation (CME), its initial value problem and structure. The derivation of the CME and the probability laws on which it depends is also discussed. Then, in section 2.3, we review the existing state-space expansion methods, discuss their limitations and give biological examples, including the latest publications, with an emphasis on finding scope for improvements in state-space expansion. In section 2.4, we review the existing approximation methods in brief. In section 2.5, we present some biological examples based on the existing methods to show the time evolution of the system. In section 2.6, we summarise this literature review that will be taken into consideration to broaden our understanding when developing a new method.

2.1 Markov Processes – An Introduction for the Computational Biologist

The time evolution of a biochemical reaction system may be represented by the Markov process only if the system depends on the present state and, based on that, it changes state using transition rules. The Markov process describes all possible states of a biochemical reaction system and illustrates all possible walk from one state to another in time. Fundamentally, the Markov process is a type of stochastic process based on Markov property (Gillespie, 1992b). A tree of stochastic process in Figure 2.1 highlights the processes, considered in this study.

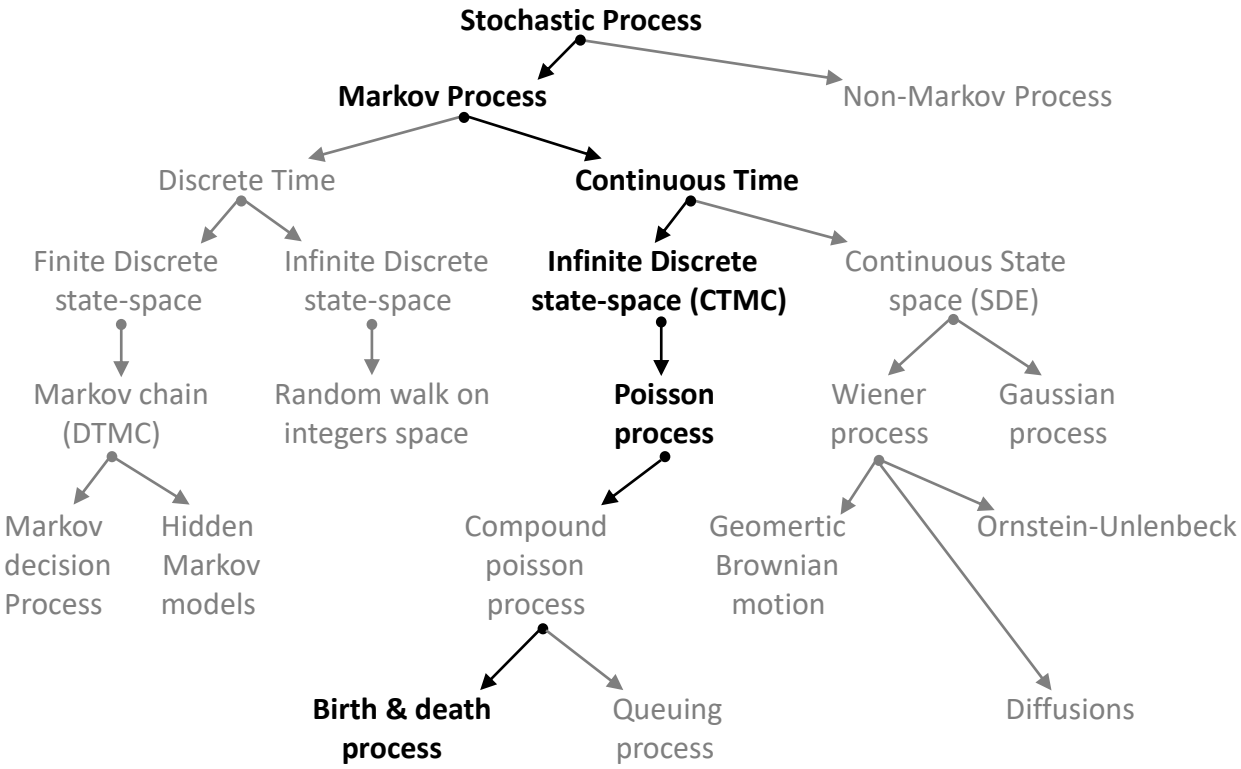


Figure 2.1. Stochastic processes are categorised into different process. This study relates to the processes that are marked with black in the flowchart.

Usually, the state-space of large biochemical reaction systems are considered infinite; however, to understand the behaviour of the system at any point it is crucial to approximate the CME. Therefore, the theory, conditions and methods that we develop in this thesis will be based on the Continuous Time Markov Chain (CTMC) to limit the infinite state-space to the finite state-space. To interpret the Markov process in detail, we first discuss the probability laws (see Appendix A) and define the conditional probability. If X_1 and X_2 are two state variables, then:

$$P_2(X_1, t_1; X_2, t_2) = P_{1|1}(X_2, t_2 \mid X_1, t_1)\, P_1(X_1, t_1). \quad (2)$$

The joint probability of calculating X_1 at t_1 and X_2 at t_2 is equal to the probability of calculating X_1 at t_1 times the probability of calculating X_2 at t_2 given X_1 at t_1. The conditional probability must fulfill the following criteria:

Criteria 1: $P_{1|1} \geq 0$

Criteria 2: $\int P_{1|1}(X_2, t_2 \mid X_1, t_1)\, dX_2 = 1$

Criteria 3: $P_1(X_2, t_2) = \int P_{1|1}(X_2, t_2 \mid X_1, t_1)\, P_1(X_1, t_1)\, dX_1$

For example, if we assume that y is a positive integer and $t_1 < t_1 < \dots\dots\dots\dots\ t_y$, then Markov property justify the Markov process by the relation:

$$P_{1|y-1}\big(X_y, t_y \,\big|\, X_{y-1}, t_{y-1}; \dots\dots\dots\dots; X_1, t_1\big) = P_{1|1}\big(X_y, t_y \,\big|\, X_{y-1}, t_{y-1}\big). \quad (3)$$

The transition probability at time t_{y-1} at value X_{y-1} to value X_y at time t_y will be conditional only on X at time t_{y-1} but not on the system's history, so $P_{1|1}$ is called the transition probability. For $y = 3$, the Markov property relation can be formed as:

$$\begin{aligned} P_3(X_1, t_1; X_2, t_2; X_3, t_3) &= P_2(X_1, t_1; X_2, t_2)\, P_{1|2}(X_3, t_3 \mid X_1, t_1; X_2, t_2) \\ &= P_1(X_1, t_1) P_{1|1}(X_2, t_2 \mid X_1, t_1) P_{1|1}(X_3, t_3 \mid X_2, t_2) \end{aligned} \quad (4)$$

If Eq (4) is integrated over X_y then divide both sides (left and right) by P_1.

$$P_3(X_1, t_1; X_2, t_2; X_3, t_3) = P_1(X_1, t_1) P_{1|1}(X_2, t_2 \mid X_1, t_1) P_{1|1}(X_3, t_3 \mid X_2, t_2)$$

$$\int P_3(X_1, t_1; X_2, t_2; X_3, t_3)\, dX_2 = \int P_1(X_1, t_1)\, P_{1|1}(X_2, t_2 \mid X_1, t_1)\, P_{1|1}(X_3, t_3 \mid X_2, t_2)\, dX_2$$

$$P_2(X_1, t_1; X_3, t_3) = P_1(X_1, t_1) \int P_{1|1}(X_2, t_2 \mid X_1, t_1)\, P_{1|1}(X_3, t_3 \mid X_2, t_2)\, dX_2$$

$$P_{1|1}(X_3, t_3 \mid X_1, t_1)\, P_1(X_1, t_1) = P_1(X_1, t_1) \int P_{1|1}(X_2, t_2 \mid X_1, t_1)\, P_{1|1}(X_3, t_3 \mid X_2, t_2)\, dX_2$$

$$P_{1|1}(X_3, t_3 \mid X_1, t_1) = \int P_{1|1}(X_2, t_2 \mid X_1, t_1)\, P_{1|1}(X_3, t_3 \mid X_2, t_2)\, dX_2 \quad (5)$$

We will then have a C-K equation (Chapman Kolmogorov equation) (Gillespie, 1992b). This determines that a process starting at t_1 with X_1 reaches X_3 at time t_3 through any attainable

value of X_2 in the transitional time t_2. As we are dealing with large biochemical reaction networks, our focus will be on the Markov process, which is nonstationary in nature. In section 2.2, we will discuss the CME problem and the homogeneous Markov processes.

2.2 The Chemical Master Equation

In this thesis, we will consider a network of biochemical reactions at a constant volume that consists of $\tilde{N} \geq 1$ different species $\{S_1, \ldots \ldots S_{\tilde{N}}\}$ that are spatially homogeneous and interact through $M \geq 1$ reaction channels in thermal equilibrium. The number of counts of each different species defines the state of the system. If all the species were bounded by S, then the approximate number of states present in the system would be $S^{\tilde{N}}$ (Burrage et al., 2006). If we describe a different species, $S_{\tilde{N}}$, with model variables of $x_{\tilde{N}}$, then each state $X \equiv (x_1, \ldots \ldots x_{\tilde{N}})^T$ is a vector of a non-negative integer, $x_{\tilde{N}}$, that denotes the number of counts of each species (negative populations of different species are not meaningful; hence, they are not considered). The dimension of the system is given by the length of the state vector which is also equal to the number of different species that react in the system. For every state, X, the probability satisfies the following CME (Gillespie, 1992a),

$$\frac{\partial P^{(t)}(X)}{\partial t} = \sum_{\mu=1}^{M} a_\mu\left(X - v_\mu\right) P^{(t)}\left(X - v_\mu\right) - \sum_{\mu=1}^{M} a_\mu(X)\, P^{(t)}(X) \qquad (6)$$

where $P^{(t)}(X)$ = the probability function representing the time-evolution of the system, given that $t \geq t_0$ and the initial probability is, $P^{(t_0)}(X_0)$,

M = elementary chemical reaction channels R_1,….. R_M,

a_μ = chemical reaction propensity of channel $\mu = \{1,2, \ldots . M\}$, and

v_μ = stoichiometric vector – this is of the same dimension as the state vector but represents a change in the molecular population of the chemical species by the occurrence of one R_M reaction. The system transitions to a new state by $X + v_\mu$ recording the changes in the number of counts of different species when the reactions occur.

We note that $a_\mu\left(X - v_\mu\right)dt$ is the probability for state $(X - v_\mu)$ to transition to state X through chemical reaction, R_M, during $[t, t + dt)$, and $\sum_{\mu=1}^{M} a_\mu(X)dt$ is the probability for the system to shift from state X through any reaction during dt. (refer to the Notations used in this thesis). Let $\mathbf{X}_J = \{X_1, \ldots \ldots X_{S^{\tilde{N}}}\}$ be the ordered set of possible states of the system indexed by $\{1,2 \ldots \ldots K\}$ having $S^{\tilde{N}}$ elements, then Eq. (6) represents the set of ordinary differential equations (ODEs) that determines the changes in probability density $P^{(t)}$= $(P^{(t)}(X_1),\ldots\ldots\ldots P^{(t)}(X_{S^{\tilde{N}}}))^T$. Once $\mathbf{X}_J$ is selected, the matrix-vector form of Eq. (6) is

described by an ODE:

$$\frac{\partial P^{(t)}}{\partial t} = A.P^{(t)}, \tag{7}$$

where the transition rate matrix is $A = [a_{i,j}]$. If each reaction leads to a different state, $X_{i'}$, then the elements in submatrix $A_{i,j}$ are given as:

$$A_{i,j} = \begin{cases} -\sum_{\mu=1}^{M} a_{\mu}(X_i), & \text{if } i = j \\ a_{\mu}(X_i), & \text{if } X_{i'} = X_i + v_{\mu} \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

This represents the infinitesimal generator of the Markov process (Gillespie, 1992b; Jones, 2005; Weber, 2012), such that rows and columns are ordered according to lowercase, i and j respectively. The entry of $a_{i,j}$ into the matrix gives the propensity for the chemical system to transition from one state to another state, given that $i \neq j$, are non-negative. The diagonal terms of the matrix are defined by a_{jj}, when $i = j$ and has a zero column sum, so its probability is conserved. From Eq. (7) we can derive the $P^{(t_f)}$ probability vector at the final time, t_f, of interest given an initial density of $P^{(t_0)}$:

$$P^{(t_f)} = exp(t_f A).P^{(t_0)}, \tag{9}$$

where the matrix exponential function is defined by the convergent Taylor series as (Eslahchi et al., 2011; Mouroutsos et al., 1985)

$$exp(t_f A) = I + \sum_{n=1}^{\infty} \frac{(t_f A)^n}{n!}. \tag{10}$$

However, algorithms, such as in (Burrage et al., 2006; Sidje et al., 2015; Sunkara et al., 2010; Wolf et al., 2010) truncate the Eq. (10) infinite summation to approximate Eq. (7) at the cost of an acute truncation error. For derivation of the CME, refer to the Appendix B.

2.2.1 Initial Value Problem

If v_μ or v_M, for μ or $M = \{1,2, \ldots . M\}$ be the stoichiometric vectors for R_M reaction channels, then we will define the stoichiometric matrix for the system by V_μ or $V_M = [v_1; v_2; \ldots\ldots v_\mu]^T$. Let φ be the sample space and $X_0 \in \varphi$ be the initial state of the system if $\mathbf{X}_J$ denotes the only set of states in φ. To solve $P^{(t)}(X)$ in Eq. (6) for $X \in \varphi$, we define the $P^{(t)}$ vector to be $(P^{(t)}(X))_{X \in \varphi}$ or $(P^{(t)}(X))_{X \in \mathbf{X}_J}$ for a finite set of states, then $\frac{\partial P^{(t)}}{\partial t}$ be defined as a vector $(\frac{\partial P^{(t)}}{\partial t})_{X \in \varphi}$. Therefore, solving the CME is to find the solution of the initial value problem over a time period by the differential equation Eq. (7) when $t > 0$, whereas, $P^{(t_0)}$ is the initial distribution at $t = 0$. Here, the sample space φ can be infinite for large biochemical systems, so finding the solution of Eq. (7) for the given parameters with finite set of states $\mathbf{X}_J$ is cited as the CME problem of large biochemical systems because the size of A will be extremely large.

For example, consider an enzymatic reaction network (Burrage et al., 2006) described by reactions $R_1: S + E \overset{k_1}{\rightarrow} C$, $R_2: C \overset{k_2}{\rightarrow} S + E$, $R_3: C \overset{k_3}{\rightarrow} P + E$. This network of reactions involves four species: namely, $S -$ substrate, $E -$ enzyme, $C -$ complex and $P -$ product molecules. The $X \equiv (x_1, x_2, x_3, x_4)^T \equiv (S, E, C, P)^T$ represents any state of the system, given $X_0 \equiv (S_0, E_0, C_0, P_0)$ as the initial state. The stoichiometric vectors are given by $v_1 = (-1, -1,1,0)$, $v_2 = (1,1, -1,0)$, $v_3 = (0,1, -1,1)$. Therefore, for (x_1, x_2, x_3, x_4) $x_{\tilde{N}=4}$, the propensity functions are:

$$R_1: \quad a_1([x_1], [x_2], [x_3], [x_4]) = k_1 \times x_1(t) \times x_2(t)$$

$$R_2: \quad a_1([x_1], [x_2], [x_3], [x_4]) = k_2 \times x_3(t)$$

$$R_3: \quad a_1([x_1], [x_2], [x_3], [x_4]) = k_3 \times x_3(t)$$

The set of states reachable from X_0 is finite in number. With the multiple explosions of the number of states in a large model, the size of A increases multiple folds.

As seen in Eq. (9), the difficulty of solving Eq. (6) is a problem when the dimension of the model grows with the number of species present in the system – especially for large biochemical models. The approximate estimate of $S^{\tilde{N}}$ shows how the size of the problem increases and this explosion in size is known as the *curse of dimensionality* (Burrage et al., 2006; Gillespie, 1977). In addition, the complete $P^{(t)}$ distribution cannot be solved precisely

in the presence of nonlinear transition probabilities for CME. We presented the justification of the CME for large biochemical reaction network or biological system, where changes in molecular counts of the species are defined by the *continuous time jump process* in the Markov process (see section 1.1) results in the new states. We will now discuss the non-stationary Markov process in section 2.2.2 with an example to show the forward and reverse reaction (i.e. one step birth-death process) and write its Master equation.

We use these notations in further chapters and refer to these equations, above, and parameters frequently to define the finite sets of states and the CME problem in terms of the initial value problem.

2.2.2 Time Homogeneous Markov Processes

In general, a homogeneous process is a nonstationary Markov process defined by probability values. For example, if any function is nonstationary and affected by the time shift, then it is defined as homogeneous process in the Markov process, which is basically a stochastic model based on Markov property (Gillespie, 1992b). There is one key category of a Markov process, called a generation-recombination process that is broadly known as a birth-death process (Novozhilov et al., 2006).

The theory of birth-death processes was coined early in the twentieth century in an effort to understand the growth of populations, by taking its stochasticity elements into consideration. Later, this study became more sophisticated when biologists understood the importance of stochastic process analysis (Jagers, 1991). The birth-death process can address most of the problems articulated by way of transitions or probabilities of the states of the process. The birth-death of species in biochemical reaction networks are continuous in time and their range is only defined by the non-negative integers $X \equiv (x_1, \dots \dots . x_{\tilde{N}})^T$, which enables the system to jump only to adjacent states when the molecular population changes. For example, if we assume that a forward reaction and its reverse reaction represent the birth and death of the species, respectively, and X_i be some state, then Figure 2.2 visualise this process as:

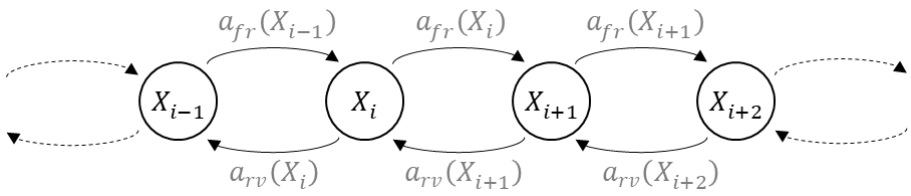


Figure 2.2. Visualisation of generation-recombination process for the Master equation.

and the Master equation of the process is given by:

$$\dot{P}(X_i) = a_{fr}(X_{i-1})P(X_{i-1}) + a_{rv}(X_{i+1})P(X_{i+1}) - (a_{rv}(X_i) + a_{fr}(X_i))P(X_i) \qquad (11)$$

where, $\dot{P}(X_i)$ represents the derivative taken with respect to time, i.e. $(\frac{\partial P^{(t)}(X)}{\partial t})$, $a_{rv}(X_i)$ is the probability of a jump process from state X_i to X_{i-1} per unit time and $P_{fr}(X_i)$ is the probability of a jump process from state X_i to X_{i+1}. Based on coefficients a_{fr} and a_{rv}, the generation-recombination processes can be categorised as – linear (if the coefficients are linear), nonlinear (if the coefficients are nonlinear) and random walks (if the coefficients are constant). As we are dealing with the stochasticity of biochemical models, in further sections, we will keep our discussion to the non-linear generation-recombination processes.

2.3 Existing State-space Expansion Methods

For most biochemical reaction networks the size of the state-space is usually in millions, when the number of different biochemical species interact in the system. On other hand, if the biochemical reaction network is large and has a large state-space, usually in billions, then the number of the most probable states which contribute to most of the probability mass, are far fewer. This makes it very challenging to expand the state-space and effectively select the states and approximate Eq. (7). These challenges are brought by the ubiquitousness of the events, which possess a stochastic process nature but, despite that, biologists have a keen interest in solving Eq. (7). Some existing state-space expansion methods such as in Figure 2.3 are widely used to create the domain for the solution of CME.

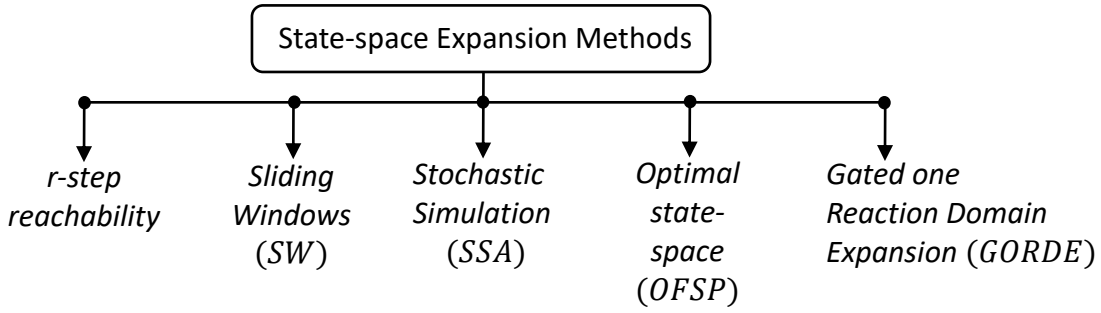


Figure 2.3. Some existing state-space expansion methods used to create domain for the solution of CME.

One can use any state-space expansion method with a approximation method based on the requirements, their limitations and compatibility. For the expansion of the state-space, the Finite State Projection (FSP) (Munsky et al., 2006) and Sliding Windows (SW) (Wolf et al., 2010) are widely used to find the domain, which is based on *r-step reachability* and the *Stochastic Simulation Algorithm* (SSA), respectively. Both methods are based on time stepping which adaptively finds the domain. The process of guessing the most probable set of states to be considered for the domain is known as the *selection of domain*. Whereas, methods like uniformization, krylov subspace (Burrage et al., 2006) and runge kutta (Mikeev et al., 2013) are popularly used for approximation (of the selected domain) of the CME Eq. (7) and broadly fall in approximation methods. As most of the improved expansion methods are based on *r-step reachability*; therefore, we will first discuss the steps of *r-step reachability* followed by a detailed discussion of the algorithms based on *r-step reachability*.

2.3.1 r-step Reachability Method

The *r-step reachability* is a popular algorithm, which finds several subsets of the finite state-space to create the domain. *r-step reachability* was originally suggested by (Munsky et al., 2006) to approximate the solution using the FSP algorithm. According to *r-step reachability*, a state should be considered for the selection of the domain if it is reachable from its present state. A system will enter into the end state, $X_{i'} \in \mathbf{X}_J$, through a finite number of intermediate states from the initial state, $X_0 \in \mathbf{X}_J$. Consequently, the *r-step reachability*, considers the initial domain and find reachable states through only one reaction at a time, then through another single reaction and so on. The final domain of an FSP algorithm will be the union of all the subset of states obtained at different times. For example, if we call $R_1(J)$ be the index of these subsets then:

$$\mathbf{X}_{R_1(J)} \leftarrow \mathbf{X}_J \bigcup_{\mu=1}^{M} \{X + v_\mu \in \varphi;\ X \in \mathbf{X}_J\} \tag{12}$$

For all reactions, $\mathbf{X}_J$ can be defined as $\mathbf{X}_{R_M(J)} = \mathbf{X}_{R_1(R_{M-1}(J))}$ and the steps in Table 2.1. The *r-step reachability* takes $\mathbf{X}_J$ (state-space), R_M (M number of reactions) and stoichiometric v_μ vectors as inputs.

Table 2.1. The r-step reachability steps.

Step 0:	**Initialise Inputs:** $\mathbf{X}_J$, M, $\{R_1, R_2, \ldots \ldots .. R_M\}$, v_μ
Step 1:	Begin with given $\mathbf{X}_J$
Step 2:	For $\boldsymbol{R_M}$ repeat: $\mathbf{X}_{J'} \leftarrow \bigcup_{\mu=1}^{M} \{X + v_\mu \in \varphi;\ X \in \mathbf{X}_J\}$
Step 3:	update $\mathbf{X}_J \leftarrow \mathbf{X}_{J'}$, and go back to step 1
	Output: $\mathbf{X}_J$

The number of new states that are reachable from the present state-space may have low probabilities, so the addition of these states does not show a significant breakthrough in the approximation error. To address this problem (Munsky et al., 2007) suggested that instead of collecting all the states only into $\mathbf{X}_J$, it is better to divide them among multiple sink states $\hat{\mathrm{G}}_1$, $\hat{\mathrm{G}}_2$,…….$\hat{\mathrm{G}}_J$ and then calculate the probabilities $g^{(t)}$= $(g^{(t)}(X_1)$,……. $g^{(t)}(X_{S^{\tilde{\mathrm{N}}}}))^T$ of the states that escape to a particular sink state. The system then follows the $ODEs$ such that

$$\begin{bmatrix} \dot{P}^{(t)} \\ \dot{g}^{(t)} \end{bmatrix} = \begin{bmatrix} A_{i,j} & 0 \\ Q & 0 \end{bmatrix} \begin{bmatrix} P^{(t)} \\ g^{(t)} \end{bmatrix} \tag{13}$$

where, $Q = \begin{cases} a_{\mu}(X), & \text{if } (X + v_{\mu}) \notin \mathbf{X}_J \\ 0, & \text{otherwise} \end{cases}$.

Although this provides knowledge about which $\hat{\mathrm{G}}_J$ leaks the high probability and direction of the high probability mass drops but, from Eq. (13), we can see that these $ODEs$ do not improve the expansion and interpretation of state-space and approximation of $\dot{P}^{(t)}$, instead they simply decompose the A matrix into a sub-matrix.

The original FSP based on the *r-step reachability* state-space expansion put into practice in (Munsky et al., 2006) and (Dinh et al., 2016) provides a logical solution for biochemical systems that have finite numbers of explicit population vectors of the species. However, if a biochemical system has an infinite or a very large number of population variations, then FSP uses a truncation point to limit the *r-step reachability* state-space to approximate the solution of the CME, but, on other side, this has some drawbacks. When any system has a certain finite state, FSP is computationally expensive for the finite numbers of steps and the truncation of states affects the solution. The FSP and its variants Krylov-FSP (Burrage et al., 2006), Krylov-FSP-SSA (Dinh et al., 2017), $OFSP$ (Sunkara et al., 2010), $FSP\ GORDE$ (Sunkara, 2013) are also based on *r-step reachability*; however, $OFSP$ and $FSP\ GORDE$ come with a variation of usage of *r-step reachability* for state-space expansion. On other hand, Krylov-FSP and Krylov-FSP-SSA are based on the Krylov subspace (Fallis et al., 2012) focused approximation of the domain created by FSP through *r-step reachability*. Krylov-FSP-SSA also differs in terms of generating the trajectories of the simulation similar to a *Stochastic Simulation Algorithm* (SSA) (discussed in section 2.3.2). The main focus of improvement in both of these algorithms was on the approximation part but not on the expansion part; therefore, we will not discuss them in detail, instead, we keep our focus on *r-step reachability* expansion part improvements.

To support a *priori*, functions like *Gated One Reaction Domain Expansion* ($GORDE$) and likelihood methods (Dinh et al., 2017) were introduced to improve the *r-step reachability* expansion strategy; however, instead of improving the state-space after the calculation of the probability vector, $P^{(t)}$, it evaluates the likelihood of reachable states by a function. The result is that *r-step reachability* explores the states in the direction of states having a high probability mass but does not improve the interpretation of the state-space. It has been reported that $FSP\ GORDE$ removes the states with small probabilities before calculation of Eq. (9) which saves omputational time and performs faster than conventional FSP *r-step reachability.* However, removing the probabilities before the calculation of Eq. (9) increases the step error and affects the accuracy of the final solution at t_f. If one is interested in solving stiff and/or large systems, this will greatly affect the solution. It is also noted that a variant called the *Optimal Finite State Projection* ($OFSP$) (Sunkara et al., 2010) based on *r-step reachability* performs better in terms of producing an optimal order of a domain and is computationally faster than FSP as well as $FSP\ GORDE$, as seen in section 5.3 of (Sunkara, 2013) but all – $OFSP$, FSP and $FSP\ GORDE$ have error of order of 10^{-4}. The $OFSP$ emphasies problems that *Sliding Windows* (SW) (discussed in following section 2.3.2) and *r-step reachability* attempt to solve – which is state-space expansion without removing any states or by truncating those states, which have low probabilities and makes A matrix excessively large. In contrast, $OFSP$ uses a compression technique (Sunkara et al., 2010) to truncate the domain by removing some of the states with low probabilities to achieve computational performance.

We will demonstrate the biological numerical experiments in section 2.5 and later in Chapter 5, we compare the most efficient version of *r-step reachability* expansion method, i.e. $OFSP$ with our methods on the basis of computational efficiency and accuracy of the domain formed.

2.3.2 Stochastic Simulation Methods

To approximate the solution of the CME, stochastic simulation methods are widely used. The approximation can be achieved by formulating and computing the number of realisations using stochastic simulation methods. The deprivation of closed-form solution has driven systems biology research towards *Monte-carlo Algorithms* (MC) (Harrison et al., 2010) to capture the dynamics. One such algorithm is the *Stochastic Simulation Algorithm* (SSA), which was originally introduced by Gillespie (Gillespie, 1977) to be used for processes in the CME to construct empirical distributions. There are state-space expansion methods that build upon stochastic simulation methods for creating a domain or a projection and one of them is *Sliding Windows* (SW). The SW introduced by Wolf (Wolf et al., 2010), is also a FSP based method but employs SSA for finding the domain, which proves to be better than FSP and suitable for stiff problems. Although, SSA is a solution method due to its employability in SW for state-space expansion, we are discussing SSA under this section but not under solution methods in section 2.4.

The improvements suggested by (Gillespie, 2001) such as the τ–leap method, has enabled computation of hundreds of stochastic simulation particles. Due to its popularity, SSA has been further improved by many others (Cao et al., 2005; Haseltine et al., 2002; Tian et al., 2004; Weinan et al., 2007) for different applications. The efficiency of the SSA has been greatly enchanced by researchers through various schemes such as the adaptive τ–leap (Cao et al., 2006; Padgett et al., 2016). The empirical distribution type approximation can be obtained by the number of realisations that are not associated with each other. When the approximation converges in the direction of the expectation of distribution (explicit) then it is said to be converging weakly; however, if the differences in the expectation of the approximation and distribution (explicit) tends towards zero, then it said to be strongly converging. Most of the methods are formulated to converge weakly because this is facile to attend and computational time can be improved (Cao et al., 2005; Haseltine et al., 2002; Weinan et al., 2007).

The τ–leap method is based on the assumption that all reactions have constant propensities and fire according to a poisson distribution in the largest chosen time step. To hold this assumption, chosing the right time step is a difficult part in τ–leap. The realisations are computed in this time step and, based on poisson distribution sampling, the reactions fired in this time step are identified. Table 2.2 shows the steps of SSA that have been employed in SW for state-space expansion.

The SSA takes $x_{\tilde{N}}$ (number of different species), X_0 (vector of initial population of species), R_M (M number of reactions), t_0 (start time), t_f (final time), v_μ (stoichiometric vectors), a_μ (propensity functions) and X (final time realised state) as inputs.

Table 2.2. Step in the Stochastic Simulation Algorithm.

Step 0:	**Initialise Inputs:** X_0, t_0, $x_{\tilde{N}}$, $\mathbf{X}_J$, $\{R_1, R_2, \dots\dots R_M\}$, $\{v_1, v_2, \dots\dots v_\mu\}$, $\{a_1, a_2, \dots\dots a_\mu\}$, t_f, X
Step 1:	Begin with given $X \leftarrow X_0$ and $t \leftarrow t_0$
Step 2:	**When** $t < t_f$, calculate $a_0 \leftarrow \sum_{\mu=1}^{M} a_\mu(X)$
Step 2a:	**if** $a_0 == 0$ then $t \leftarrow t_f$; break and **end**
Step 2b:	Generate two random numbers using uniform random number (URN) generator say num_1, num_2 and calculate $\tau \leftarrow \frac{1}{a_0}\log\left(\frac{1}{num_1}\right)$
Step 2c:	**if** $t + \tau > t_f$ then $t \leftarrow t_f$, break and **end**
Step 2d:	$t \leftarrow t + \tau$
Step 2e:	select M' such that $\sum_{\mu=1}^{M'-1} a_y(X) < a_0.num_2 \leq \sum_{\mu=M'}^{M} a_\mu(X)$
Step 2f:	$X \leftarrow X + v_{M'}$
Step 2g:	**end**
Step 3:	**Output:** X
Step 4:	Go back to step 1

In *Step 2e*, we denote the number of poisson process by the positive integer, y. The τ–leaping is also popularly known as the Euler approximation of the *jump process*, which is given by

$$X_{x_{\tilde{\text{N}}}}(t) = X_{x_{\tilde{\text{N}}}}(0) + \sum_{y=1}^{M} Y_y \left(\int_0^t a_y \left(X_{x_{\tilde{\text{N}}}}(\tilde{\text{N}}) \right) d\tilde{\text{N}} \right) v_y \,, \tag{14}$$

where, Y_y is a poisson process given that $0 < y \leq M$. Let $\tau_0 = 0$,……, $\tau_{lp} = t_f$ be $(lp + 1)$ values between the interval of $[0, t_f]$ given $X_{x_{\tilde{\text{N}}}}(\tau_0) = X_0$, the initial state of the biochemical system. Therefore, the τ–leap realisation can be determined by:

$$\dot{X}_{x_{\tilde{\text{N}}}}(\tau_{lp}) = \dot{X}_{x_{\tilde{\text{N}}}}(\tau_0) + \sum_{y=1}^{M} Y_y \left(\sum_{lp'=1}^{lp-1} a_y \left(\dot{X}_{x_{\tilde{\text{N}}}}(\tau_{lp'}) \right) (\tau_{lp'+1} - \tau_{lp'}) \right) v_y. \tag{15}$$

For a detailed description of SSA, refer to (Gillespie, 1977). The SW uses SSA as a function to extract the number of samples at time t. The SW is more suitable for biochemical systems that have few reactions with high propensities compared to the rest of the reactions in the network. Due to this non-uniformity in the firing of reactions, the shape of the domain becomes rectangular in terms of one co-ordinate being the largest and the second being the smallest. This formation of the domain is constructively captured by the SSA function when it follows the direction of reactions firing at high rates which forms this shape. The SW by (Wolf et al., 2010) also demonstrated that it has better computing efficiency compared to *r-step reachability* when it comes to stiff systems.

On other hand, it is not feasible to use SW for large CME problems either, as creating hyper-rectangles is an unattainable task. At least four-times the number of SSA simulations are required to minimise the error to half. This is because the convergence rates of routines in MC are very slow. The original SSA takes a long time since one simulation may have a number of different R_M. It is important to note that the number of simulations of SSA should increase with the growing domain in order to avoid the truncation of modes when approximating the solution of the CME. Currently, there is no standard rule mentioned anywhere in the literature for selecting the number of simulations. When approximating the solution of the CME, sufficient amount of the time is invested in finding the right domain, which is to be solved for the approximation. To differentiate the distribution solution of the approximation of the Eq. (7) and the stochastic simulations, several SSA realisations are required to build the distribution for the required error, and appropriate states are required in domain to produce the approximate distribution with a similar error.

Once SW captures the states via SSA, solution methods like the Krylov subspace and global uniformisation can be employed (Wolf et a., 2010). The Krylov subspace has speed-up factor of 1.5 times and; therefore, the computational time of the Krylov subspace is superior to the global uniformisation method. The proficiency of global uniformisation for non-stiff biochemical network systems is better compared to the Krylov subspace, which makes it suitable for stiff systems.

Before we move on to biological examples, in section 2.4.1, we will discuss these solution methods in brief in the following section, section 2.4.

2.4 Existing Numerical Solution Methods

In this section we will briefly discuss two numerical solution methods; namely, the Uniformization and the Krylov subspace methods that are widely employed by most CME approximation algorithms (Burrage et al., 2006; Dinh et al., 2017; Fallis et al., 2012; Wolf et al., 2010). Some existing numerical solution methods such as in Figure 2.4 are used to approximate the solution of Eq. (9).

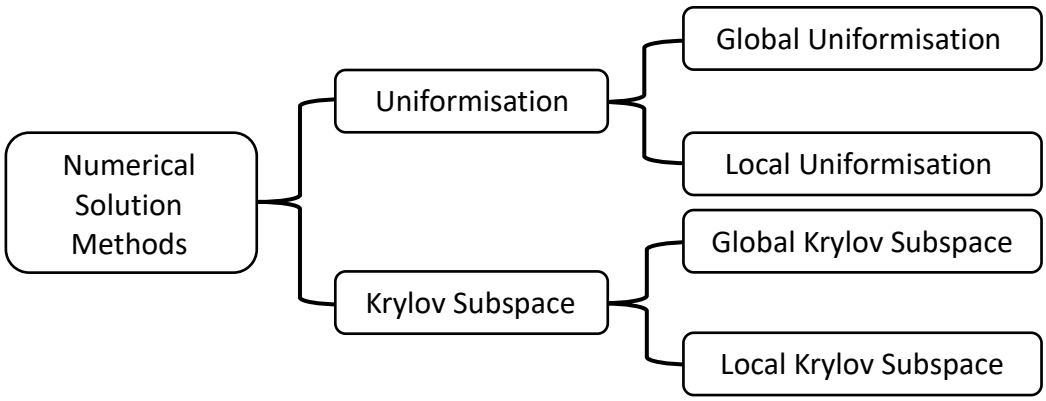


Figure 2.4. Types of numerical solution methods – Uniformisation and Krylov subspace methods. Each is categorised based on global and local solution.

Both methods are used to approximate the global solution Eq. (9) of the CME, as well as the local solution, Eq. (17). The general solution of Eq. (7), is given by:

$$P^{(t_f)} = P^{(t_0)}.e^{At_f}. \tag{16}$$

In SW, the implementation only considers the finite submatrix of A_j that contains entries of the states present in SW window $W^{(j)}$, where $j \in \{1,2 \dots \dots j'\}$ and is defined as

$$\dot{P}^{(t)} = \dot{P}^{(t_{j-1})}.D_j.e^{A_j t_j}, \tag{17}$$

where, D_j is the diagonal matrix whose diagonal entries are one.

2.4.1 Uniformisation Method

The uniformisation defines the discrete-time Markov chain (CTMC) and a Poisson process (Wolf et al., 2010). The Poisson distribution with parameter λt is given by:

$$P_r = e^{-\lambda t} \frac{(\lambda t)^y}{y!}, \tag{18}$$

where $y \geq 1$ and λt is the uniformisation rate. The solution of transient state probabilities in Eq. (16), can be rewritten as:

$$P^{(t)} = P^{(0)} \sum_{y=0}^{\infty} e^{-\lambda t} \frac{(\lambda t)^y}{y!} P^y , \tag{19}$$

$$= \sum_{y=0}^{\infty} e^{-\lambda t} \frac{(\lambda t)^y}{k!} W^y \tag{20}$$

W^y can be computed inductively by $W^{(0)} = P^{(0)}, W^{(j)} = W^{(j-1)}P$. If P is sparse, W^y can be calculated even if the size of the state-space is large. The upper and lower summation bounds can be determined such that for each X the error,

$$P^{(t)}(X) - \sum_{y=L}^{U} e^{-\lambda t} \frac{(\lambda t)^y}{y!} W^y(X) = \sum_{\substack{0 \leq y < L, \\ U < y < \infty}} e^{-\lambda t} \frac{(\lambda t)^y}{y!} W^y(X) \tag{21}$$

$$\leq \sum_{\substack{0 \leq y < L, \\ U < y < \infty}} e^{-\lambda t} \frac{(\lambda t)^y}{y!}$$

where, X is any state, U is the upper summation bound, L is the lower summation bound.

Thus, $P^{(t)}$ can be approximated with arbitrary accuracy by

$$P^{(t)} \approx \sum_{y=L}^{U} e^{-\lambda t} \frac{(\lambda t)^y}{y!} W^{(y)} \tag{22}$$

as long as the required number of summands is not extremely large.

When uniformisation method is combined with the SW method it is called as local uniformisation. We invoke uniformisation method to approximate Eq. (17). $P_j = I + \frac{1}{\lambda_j} A_j$ is a

substochastic transition matrix and uniformization rate $\lambda_j = \max_{X \epsilon W^{(y)}} \lambda_X$. By using the similar calculations as in Eq. (22), we get a substochastic vector, where:

$$P^{(t_j)} = P^{(t_{j-1})} D_j \sum_{y=L}^{U} e^{-\lambda_j t_j} \frac{(\lambda_j t_j)^y}{y!} \; P_j^y \tag{23}$$

$$= \sum_{y=L}^{U} e^{-\lambda_j t_j} \frac{(\lambda_j t_j)^y}{y!} \; P_j^y \tag{24}$$

L and *U* are the truncation points depending on uniformisation rate and $W'^{(y)}_j = \dot{P}^{(t_{j-1})}.D_j.P_j^y$. In SW, local uniformisation computation time is saved compared to global uniformisation, if computational complexity $\sum_{j=1}^{j'} \mathfrak{v}_j \lambda_j t_j \ll \mathfrak{v} \lambda t$, where $\lambda = \sup_{X\, S} \lambda_X$ and $\mathfrak{v}_j$ is the number of nonzero elements in P_j.

2.4.2 Krylov Subspace

As discussed in section 2.2, the expansion of states results in repeatedly increase in the size of the A matrix, which makes computing of the matrix exponential expensive with time. When the projection size increases with states having low probabilities then it wastes the computation; whereas, if the projection size is not sufficient to achieve good acurracy of the solution, then the whole process is repeated up to the desired time, t_f. The process of state-space expansion and computing the matrix exponential can be done simultaneously. This is one of the improvements made by (Burrage et al., 2006) to FSP approximation by employing the Krylov subspace method.

We know that the global of solution of the CME is given by Eq. (7). Following this, we determine the approximation of $e^{(tA)}v$, where v is a column vector of a length equal to the number of different species present in the system and A is the square matrix. The approximation can be obtained by selecting $v = (P^{(0)})^T$ and the large sparse matrix to be converted into a small matrix. The dimension, dim, of the sub-matrix is usually small (for example in SW, $dim = 20$ or 30 is usually chosen), which can be changed dynamically with the expansion. Therefore, the basis of the Krylov subspace is defined as:

$$Span\{v, Av, \dots \dots A^{(dim-1)}v\}, \tag{25}$$

and approximate the solution for $e^{(tA)}v$ from this subspace. Working straight with this subspace is unsteady; therefore, an orthonormal basis $\{v_1, v_2, \dots \dots v_{dim}\}$ is established for the subspace by the application of the *Arnoldi method* (Burrage et al., 2006) to $v, Av, \dots \dots A^{(dim-1)}v$. If $\bar{H}_{dim}$ is the upper matrix (Hessenberg Matrix) solved by the *Arnoldi method* and $\bar{h}_{dim+1,dim}$ be the value of normalisation then:

$$AV_{dim} = V_{dim}\bar{H}_{dim} + (\bar{h}_{dim+1,dim})(v_{dim+1})(e_{dim}^T) \tag{26}$$

$$V_{dim}^T AV_{dim} = \bar{H}_{dim} \tag{27}$$

If $e^{(tA)}v$ is to be approximated for $t > 0$, then the approximation error is controlled by calculating it in a step-wise manner such that $e^{(t_{step(1)}+t_{step(2)})A}v = e^{(t_{step(1)})A}.e^{(t_{step(2)})A}v$ for $t_{step(1)}, t_{step(2)} \geq 0$. For step t_{step}, if β is the 2nd-norm of v then approximation of $e^{(tA)}v$ is given by

$$\beta \bar{B}_{dim+1}\, exp(t_{step}\, \bar{H}_{dim+1})\, e_1 \tag{28}$$

where, e_1 is the first unit basis vector. The squaring and scaling of $exp(t_{step}\ \bar{H}_{dim+1})$ is computed using pade approximation (Moler et al., 2003).

In following section, section 2.5, we discuss some examples based on the biological models for which the CME has been applied.

2.5 Biological Examples

We discuss some numerical examples of the exisitng algorithms that have been applied to biochemical systems to solve for the CME solution. Every example illustrates ways about how probabilities of the species evolve over time; however, a key problem domain still remains in finding the optimum support for the efficient solution of the CME.

Example 1: Solving T-cell homoeostasis (two participating clono-types species) for the CME solution using *r-step reachability* for the expansion and FSP for the approximation.

It is a network of four reactions defined by:

$$\begin{aligned}
&\boldsymbol{R_1}\colon\ \star \to A && a_1([A],[B]) = \frac{60[A]*0.5}{([A]+[B])} + \frac{60[A]*0.5}{([A]+1000)}\\
&\boldsymbol{R_2}\colon\ A \to \star && a_2([A],[B]) = [A]\\
&\boldsymbol{R_3}\colon\ \star \to A && a_3([A],[B]) = \frac{60[B]*0.5}{([A]+[B])} + \frac{60[B]*0.5}{([B]+1000)}\\
&\boldsymbol{R_4}\colon\ B \to \star && a_4([A],[B]) = [B]
\end{aligned}$$

with initial copy counts of $A_0 = 20$, $B_0 = 20$ and the reaction propensities are unusual, as they are time dependent. These species counts are used as a state-space to define the system. The *r-step reachability* is applied and the FSP method is then used to find the solution of the CME upto $t_f = 10$ *secs*. Figs (A) to (F) in Figure 2.5 show the probability distributions at different time points. It is also seen in the Fig (F) of Figure 2.5, that it divides into non-convergent subsets.

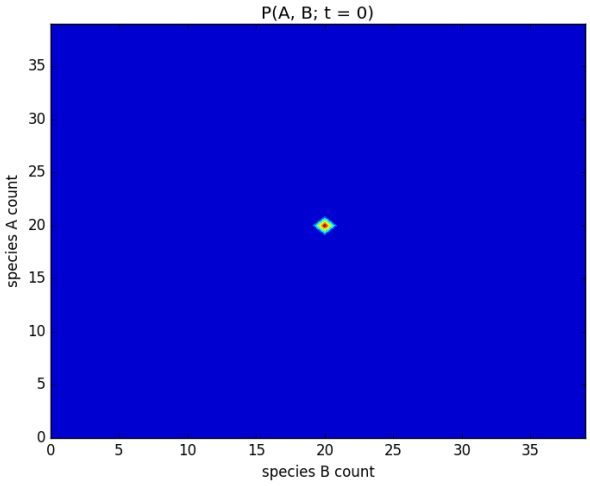


Fig (A). Joint Probability at $t = 0$ *sec*

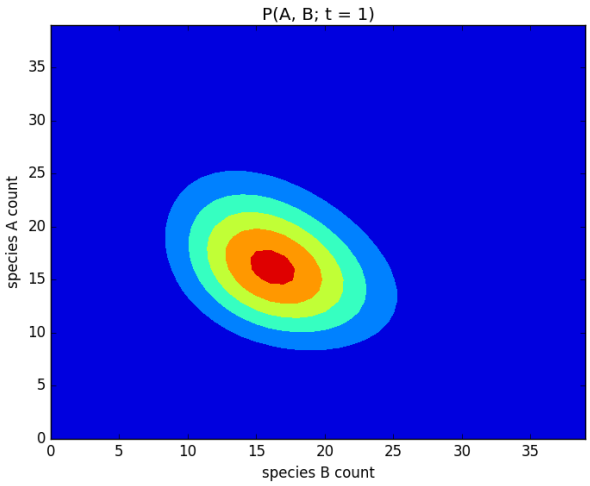


Fig (B). Joint Probability at $t = 1$ *sec*

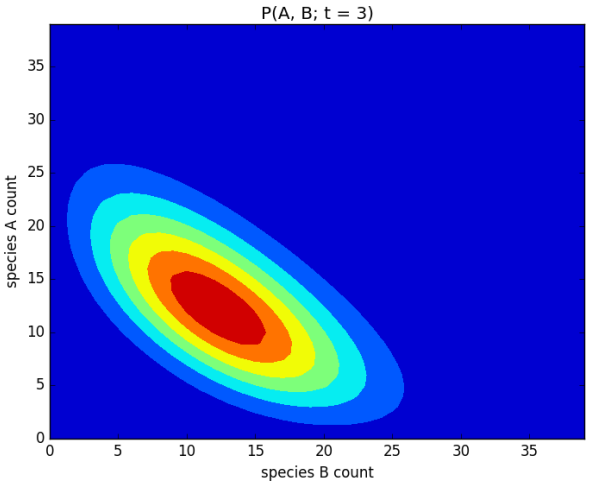


Fig (C). Joint Probability at $t = 3$ *sec*

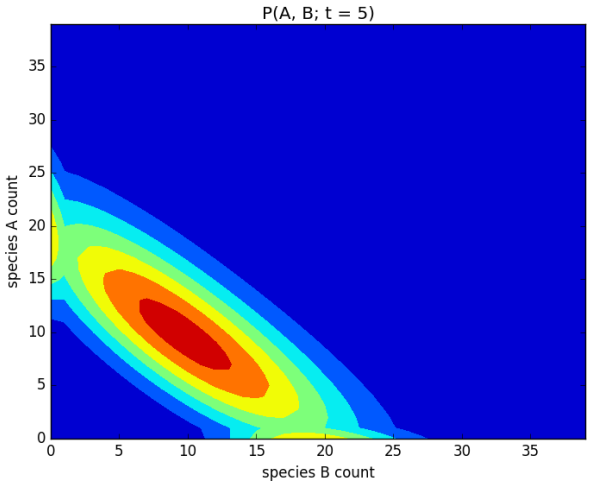


Fig (D). Joint Probability at $t = 5$ *sec*

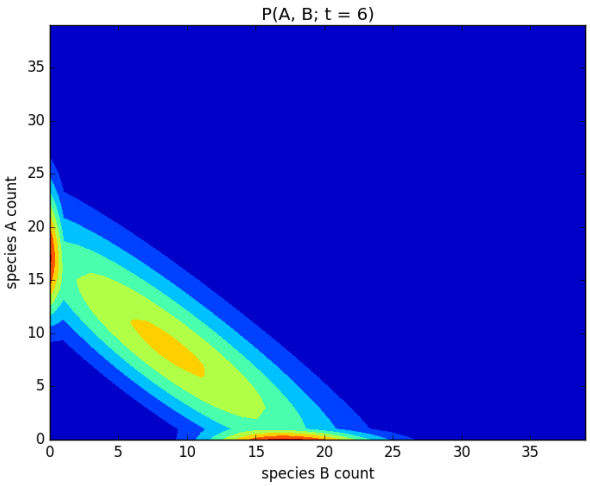


Fig (E). Joint Probability at $t = 6$ *sec*

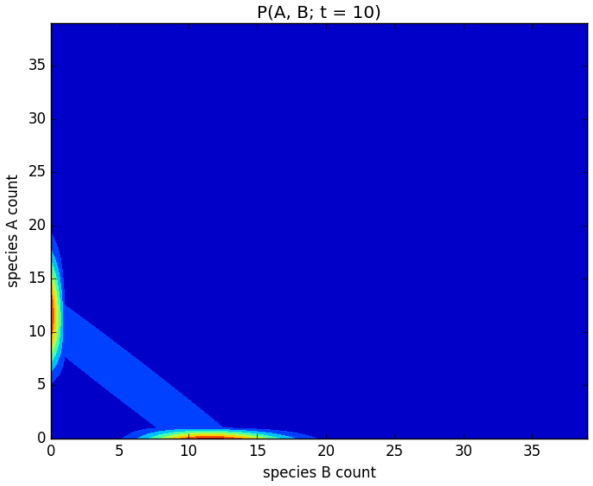


Fig (F). Joint Probability at $t = 10$ *sec*

Figure 2.5. CME solution of T-cell homoeostasis at different *times (0, 1, 3, 5, 6, 10 secs)* with *A* (y-axis) and *B* (x-axis). The significant part of the probability mass is initially located at *t=0 sec* as given in Fig (A), then shifted further, as given in Fig (F) at *t=10 secs.*

Example 2: Solving a gene expression model involving mRNA and protein to: (a) generate SSA realisations, and (b) for the CME solution using *r-step reachability* for expansion and FSP for approximation.

It is a network of four reactions defined by

$$\boldsymbol{R_1}\text{:}\ \star \xrightarrow{k_1} \mathrm{m}RNA \qquad a_1([\mathrm{m}RNA],[Protein]) = k_1(1 + k_{fd}[Protein])$$

$$\boldsymbol{R_2}\text{:}\ \mathrm{m}RNA \xrightarrow{k_2} \mathrm{m}RNA + Protein \qquad a_2([\mathrm{m}RNA],[Protein]) = k_2([\mathrm{m}RNA])$$

$$\boldsymbol{R_3}\text{:}\ \mathrm{m}RNA \xrightarrow{k_3} \star \qquad a_3([\mathrm{m}RNA],[Protein]) = k_3([\mathrm{m}RNA])$$

$$\boldsymbol{R_4}\text{:}\ Protein \xrightarrow{k_4} \star \qquad a_4([\mathrm{m}RNA],[Protein]) = k_4([Protein])$$

which represents the transcription, translation and degradation of mRNA and protein. The state of the system is defined by the $(\mathrm{m}RNA, Protein)$. Where, $k_{fd} = 0$ is the negative feedback value, $k_1 = 0.1$, $k_2 = 0.1$, $k_3 = 0.1$, $k_4 = 0.002$ and the initial point density is assumed to be $(0,0)$. The SSA is applied to generate the realisation upto $t_f = 2000$ *secs* as shown in Fig (A) and (B) of Figure 2.6 and, simulatenously, *r-step reachability* is also applied with FSP method to find the solution of the CME at $t_f = 2000$ *secs*.

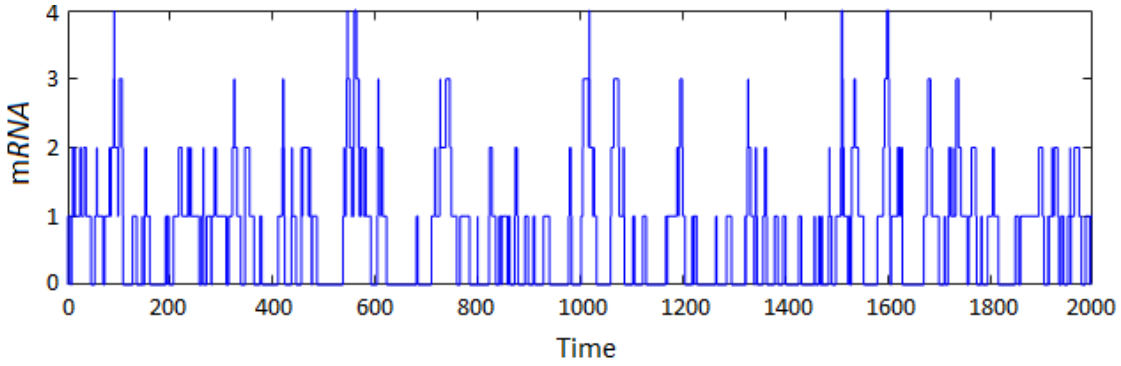


Fig (A). SSA trajectory of mRNA upto $t_f = 2000$ *secs*

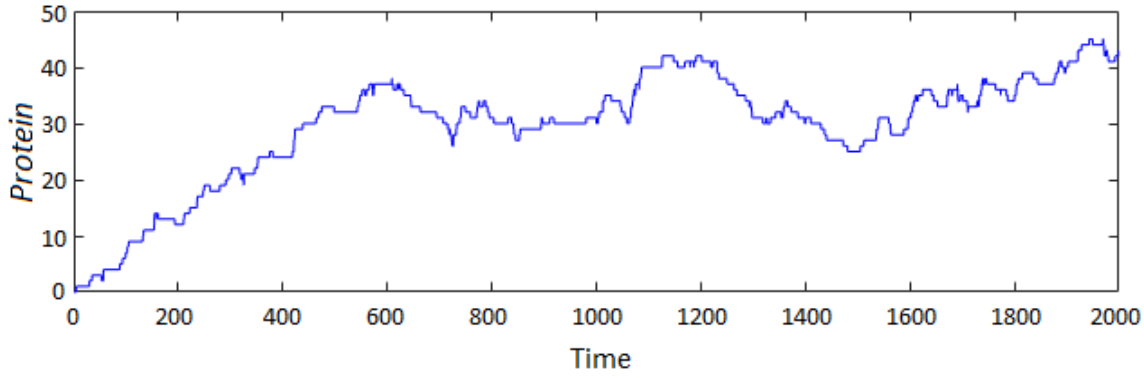
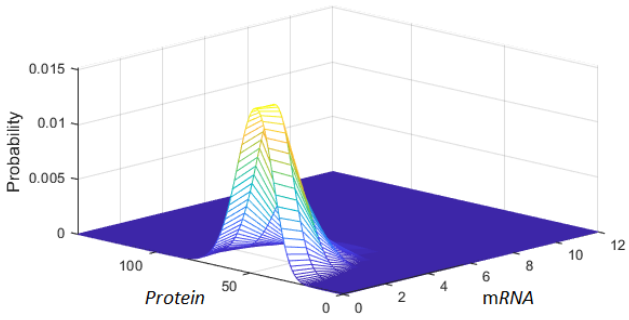


Fig (B). SSA trajectory of protein upto $t_f = 2000$ *secs*

Figure 2.6. SSA trajectories of mRNA and protein of a gene expression model upto $t_f = 2000$ *secs*.

Fig (A) of Figure 2.7 is the joint probability distribution of the species (mRNA and protein); whereas, Fig (B) of Figure 2.7 shows the stochastic versus deterministic response at t_f. The conditional probability of the species (mRNA and protein) in Fig (C) and Fig (D) of Figure 2.7 determine that protein attain high probability at t_f as compared to mRNA, when number of molecular counts of protein is between 40 and 50.



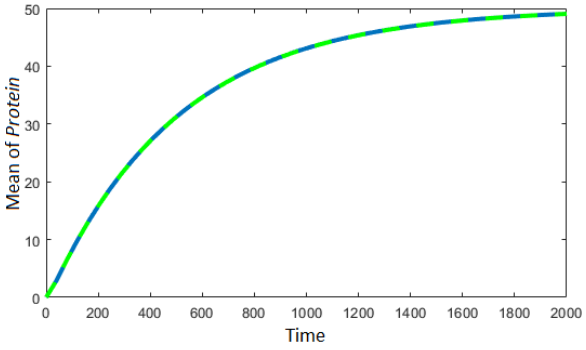
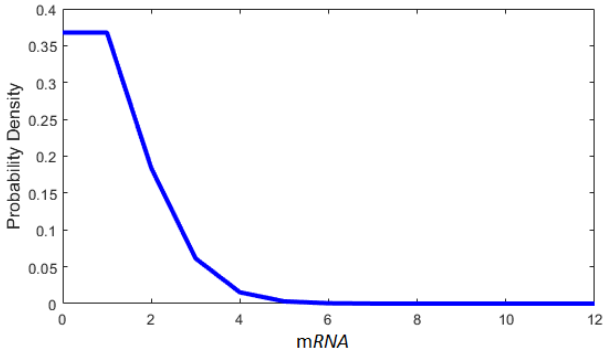


Fig (A). Joint probability distribution at t_f **Fig (B). Stochastic vs Deterministic upto t_f**



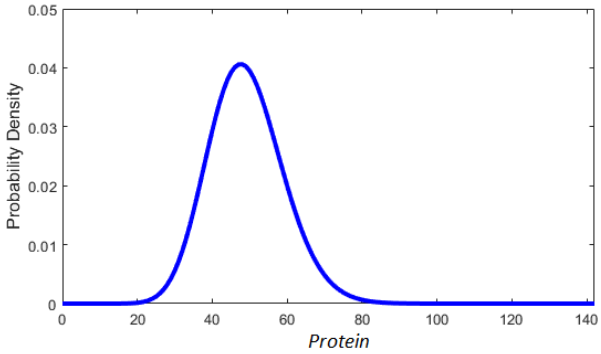


Fig (C). Probability distribution of mRNA **Fig (D). Probability distribution of protein**

Figure 2.7. Probability distribution of species of gene expression model at $t_f = 2000$ *secs*. Fig (A) is the joint probability distribution of the species, Fig (B) is the stochastic versus deterministic response, Fig (C) and Fig (D) are the conditional probabilities of mRNA and protein respectively.

Example 3: Solving Michaelis Menten reaction system for the CME solution using the $OFSP$ algorithm.

It is a network of three reactions defined by

$$\boldsymbol{R_1}\text{:}\ S + E \xrightarrow{k_1} C \qquad a_1([S],[E],[C],[P]) = k_1([S],[E])$$

$$\boldsymbol{R_2}\text{:}\ C \xrightarrow{k_2} S + E \qquad a_2([S],[E],[C],[P]) = k_2([C])$$

$$\boldsymbol{R_3}\text{:}\ C \xrightarrow{k_3} P + E \qquad a_3([S],[E],[C],[P]) = k_3([C])$$

with initial copy counts $S_0 = 50$, $C_0 = 0$, $P_0 = 0$, $E_0 = 10$, and reaction rate parameters $k_1 = 0.01$, $k_2 = 35$, $k_3 = 30$. In this biochemical system, substrate S will transform into product P via complex C when enzyme E acts as a catalyst for the reaction. The *r-step reachability* is applied and the $OFSP$ method is used to find the solution of the CME upto $t_f = 10$ *secs*, with time step 0.1 and compression window of 1. Figs (A) to (D) in Figure 2.8 show the probability distribution of species at $t_f = 10.0$ *sec*.

$OFSP$ explore 338 states at the end of t_f. The response of $OFSP$ in Figure 2.9 show that, the copy counts of S continuously decrease over time due to R_1 resulting in transformation into product P, which increase its copy counts; whereas, enzyme E remains constant.

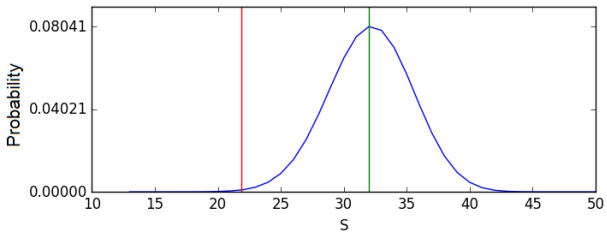


Fig (A). Probability of species S over t_f

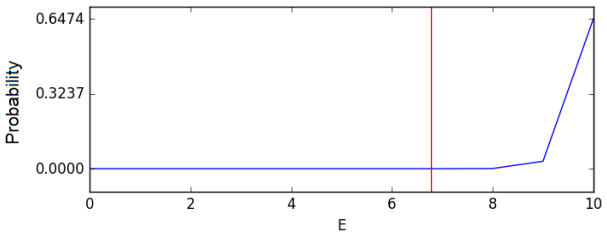


Fig (B). Probability of species E over t_f

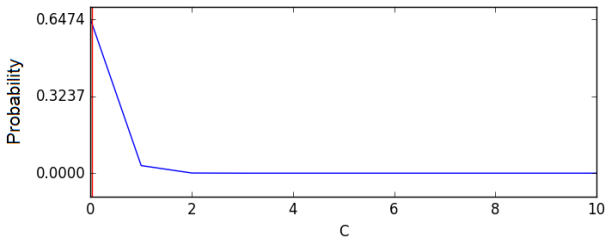


Fig (C). Probability of species C over t_f

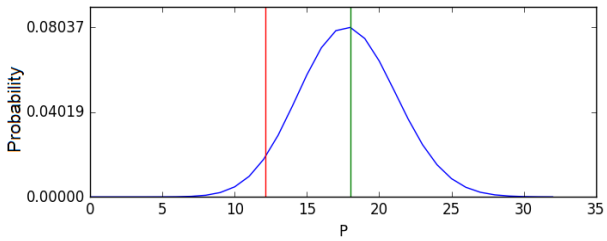


Fig (D). Probability of species P over t_f

Figure 2.8. Conditional probability of the Michaelis Menten reaction system species evaluated at $t_f = 10.0$ *sec*, using $OFSP$.

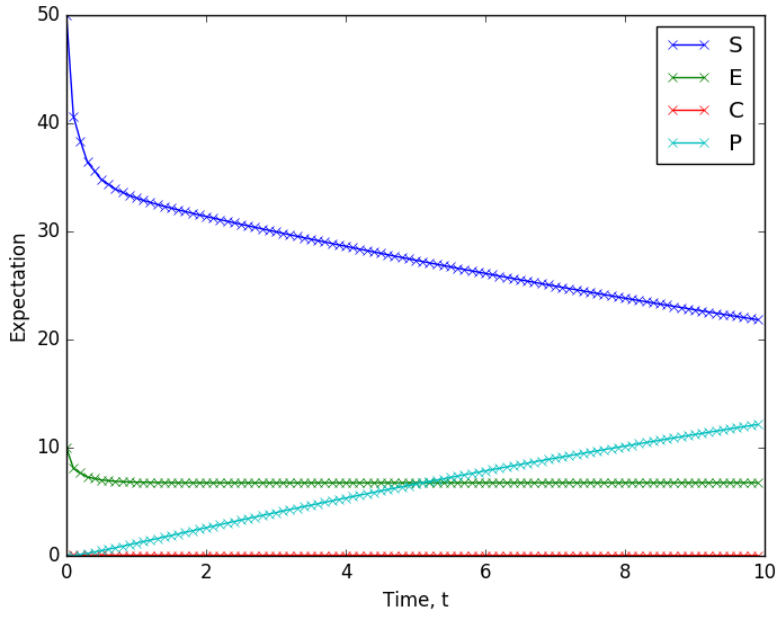


Figure 2.9. Change in molecular counts of the species of Michaelis Menten reaction system over t_f.

2.6 Discussion and Conclusions

In this chapter, we first discussed the types of stochastic processes, particularly the Markov process, which is an homogeneous nonstationary process, that helps in understanding the stochastic behaviour of the biochemical reaction network and how it is affected by the time shift. To understand this time evolution behaviour of a system, the chemical master equation (CME) is used which is an oridinary differential equation (ODE) given by Eq. (6). Usually, the Eq. (7) vector form is used to approximate the solution of the CME; however, when the dimension of the system is large, then the size of A matrix becomes large with time and this makes the numerical solution of the CME computationally expensive. Broadly, the solution to the CME is a combination of: (a) a state-space expansion followed by (b) an approximation of Eq. (7). This study focused on developing a state-space expansion method to create the optimum domain to improve the computational efficiency as well as the solution of the CME.

There are several state-space expansion methods that have been developed such as the *r-step reachability* and SSA (used for expansion by a sliding windows algorithm), as discussed in section 2.3. The *r-step reachability* has been developed for the FSP approach to create the projection for the approximation; whereas, SSA, also called as Gillespie's algorithm, was originally developed to generate stochastic simulations and comes under numerical solution methods. The SW algorithm is a FSP based algorithm that uses SSA for expansion of state-space. In section 2.3, we discussed about the variants of these algorithms (such as Krylov-FSP, Krylov-FSP-SSA, $OFSP$, $FSP\ GORDE$ etc) that have been introduced to the literature over recent years. Most of these variants uses the *r-step reachability* method to expand the state-space with some variations introduced in $OFSP$ (with compression), $GORDE$ (weightage function for FSP) and, based on the literature, $OFSP$ was proved to be most efficient among all other variants of expansion methods based on *r-step reachability*.

On other hand, we have also briefly discussed the existing numerical solution methods (such as the Uniformisation method and the Krylov subspace method) in section 2.4 which are commonly employed to approximate the solution of the CME once the domain is formed. In section 2.5, through biological examples, we have seen how the probabilities of the species evolves with time. Example 1 of T-cell homoeostasis is based on *r-step reachability* for expansion and FSP for approximation, Example 2 of the gene expression model is based on: (a) generating SSA realisations; and (b) the solution of the CME using *r-step reachability* for expansion and FSP for approximation, Example 3 of the Michaelis Menten reaction system is based on the $OFSP$ method, which uses *r-step reachability* for an expansion with compression

and FSP for approximation.

Based on our literature discussion (see section 2.3, 2.4, 2.5) it is evident that CME has been employed and solved explicitly for relatively small biological systems, while computationally complaisant but accurate solutions are still unknown for most significant systems and also for large systems having infinite states. In addition, the FSP (or more concretely *r-step reachability*) performance is better than SSA in terms of computational efficiency and accuracy. Further, this is also due to the small numbers of population of species in these examples, where FSP outperforms the SSA or τ leaping method as it is based on the support size but not on realisations. It is also noted that $OFSP$ is one of the most computionally efficient and optimum domain creation algorithm present in the literature. On the other hand, SSA employed in SW proved to be efficient for stiff problems compared to FSP (*r-step reachability* for expansion). Therefore, we will compare the performance of $OFSP$ and SSA (τ leaps adaptive) with our methods in terms of finding the domain, accuracy and computational efficiency. In such cases, a crucial part of solving the CME remains in finding the right projection size (domain) for large models, which would then make the approximation efficient.

Chapter 3
Markov Chain Tree and Graphs for Markov Process

In this chapter we use the concept of a Markov chain and its process applications set out in Chapters 1 and 2 to define the stochastic process and broadly frame its fundamentals for our methods. Such processes associated with Markov chains are most important in biochemical networks for solving the CME. This chapter also introduces the key functions and theory that are used to describe these processes as Markov chain trees and graphs as well as deriving some conditions that must be followed by the functions. We quickly discovered that the state-space problem can be simplified by characterisation of a Markov chain that requires the form what is called as graphs that adheres to the Markov chain tree properties. In this chapter we also show, how a Markov chain tree can be used to visualise stochastic processes, and we define that phenomenon fully as we can without following ourselves to the exact form of a Markov chain.

In this chapter, in section 3.1, we first give an overview of Markov chains, and the important definitions and preliminaries associated with the events in the Markov chain processes. In section 3.2, we discuss the finite state Markov chains to create a sample space for searching. We also discuss the classification of the states in Markov chains that relate to the type of reactions. In section 3.3, we set the standards to visualise the Markov chain as a Markov chain graph or tree (a special type of graph). In section 3.4, we define the problem state-space model for biochemical reaction networks where the searching is to be done. Section 3.5, discusses the adoptation of *AI* search standards for states space expansion. Based on the standards we define the data structure that includes important elements required for efficient searches and a detailed visualisation of the state-space. To enforce the expansion towards a high probability mass we will develop a function based on Bayes' theorem to support the expansion strategy. Section 3.6, provides important definitions for algorithm's *time* and *space* complexity analysis. Lastly, in section 3.7, we give a brief discussion and conclusions of this chapter.

3.1 Introduction

The behaviour of a biochemical system is often represented by describing the occupancy of different states and indicating how the system stays and moves between these states over time. A biochemical system is represented by a *Markov process* if the time spent by the system in any state is distributed exponentially. In particular, if the biochemical system does not have an exponential distritbution then it is possible to invoke the relevant representation implicitly. Because of this, the Markov process is used extensively for biochemical systems.

Large biochemical systems possibly have infinite set of states associated with the Markov process, and this is assumed to occupy a single state at any moment of time. Once the system leaves the initial state, the new state becomes the initial state for future states. These transitions between the states in the Markov process define the evolution of the system. These transitions are instantaneous and consume zero time when moving from one state to another. If any biochemical system posseses this Markov property it is referred to as a Markovian system whose future evolution is based only on the present state and not on the past states. The information we are interested to draw from the biochemical system is the knowledge of the probabilities about remaining in the current state at a certain time after the system becomes active and jumps to a new state. Often the influence of the given initial state is erased as the time taken to remain in any particular state is sufficiently more. Another interest is the time taken to reach a certain state for the first time.

For example, consider the response of a butterfly who flies from one flower to another, where all the flowers constitute the state of the system. This shift of the butterfly depends only on what information can be deduced from the current flower (state) as it has no memory to recall about previous states visited before the present state, nor the length of time spent on previous and present state. The time spent in the air while flying is negligible compared to the time spent stitting on flowers (state), which justifies the assumption of instantaneous transitions. If one flower is not a regular flower but one of those who eats flies, then we would be interested in knowing for how long the butterfly can survive before being devoured by the flower.

In the theory of Markov chains, the subject of the tree indexed processes associated with its graphs did not exist before. The walk from the initial state to the new state and its stochastic processes are interesting problems and the results are usually represented in terms of the sample space and the paths between nodes in the graph. Contemporary, the aim of the study of walks on graphs to see the characteristics of the corresponding biochemical systems are reflected when the sample space is explored as the state changes. Typically, if the graph is

associated with a large biochemical system then the ordinary walk will be transitory and the sample space will only contain a tiny part of the states. As a result, the probablistic classification of the expander returns the probabilities involved with a low probability mass and rates of mixing, rather than the behaviour of the sample path. Here, associated tree indexed walks defines the processes by providing a sample paths in space that are sufficient to reflect most of the probabilities of a large graph while maintaining the Markov property of the processes.

3.1.1 Definitions and Preliminaries

In this section, we will provide some definitions and the preliminaries required to understand the concepts of graphs, probability mapping and associated events. We also define the building blocks of the stochastic processes of any biochemical system in which the Markov process is crucial.

Principles of True events. Given φ is the sample space of any biochemical system, then class $\varphi' \subset \varphi$ contains all the events satisfying the following principles:

1. φ is considered to be a system of events.
2. For every event $'E_1$, $'E_2$…., sequence, the union $\cup_y$ of all the events, $'E_y$, is an event.
3. Complement $'E^c$ is an event for every $'E_y$.

then φ is called as $'E_y$ space of a biochemical system, where y is a real positive integer.

Principles of Probability in sample space. Given that φ is the sample space of any biochemical system and φ' be any class of the event space, the probability function $P^{(t)}$ maps each $'E_y \in \varphi'$ to a finite number such that the following probability conditions match:

1. $P^{(t)}(\varphi) = 1$
2. $P^{(t)}('E_y) \geq 0$, for every $'E_y$.
3. Summation of individual probabilities of uncommon events are given by the probability of union of any $'E_1$, $'E_2$…., sequence

$$P^{(t)}\left(\cup_{y=1}^{\infty} \text{'E}_y\right) = \sum_{y=1}^{\infty} P^{(t)}(\text{'E}_y) \tag{29}$$

defines the mapping of the sequence of events with the probability function, so we can now describe the random variable, which is an important concept in our study.

Definition 3.1. Random variables are real value functions that convert the sample space of a biochemical system to a probability space; thus, it has possible outcomes from the sample space only if the probability distribution is properly defined. A stochastic process has an infinite group of random variables and such random processes in biochemical systems are usually indexed by time, so that each sample in the space gives rise to the path when each sample point maps to the real value functions and these paths vary with continuous time in Markov chains. In the following sections we will work with the Markov processes that are associated with continuous time Markov chain (CTMC) and discrete states.

3.2 Finite state Markov Chains as sample space

In a finite state Markov chain, if the process changes its state at any moment in time and, if the states of a Markov process are discrete, then this type of stochastic process is a continuous-time Markov chain (CTMC).

At every $t \geq 0$, the random variable $X(t)$ is the state at t and the collection of these random variables forms CTMC, where $\{X(t); t \geq 0\}$. These continuous time processes consist of experimental probabilities $P^{(t)}(X)$ that are defined on φ and the set of functions assigning time function in φ for each outcome. In addition to these processes, the Markov chain strictly follows the Markov property as:

Definition 3.2. A continuous time process $\{X(t); t \geq 0\}$ is the property through which the values of variables, such as $\{X(t); t \geq 1\}$, stay in the finite set X_J and depend on the recent variable, $X(t-1)$. For all t and $\{i, \dots \dots i'\}$ in $\mathbf{X_J}$,

$$P(X_{i'}(t) \mid X_i(t-1)\,; X(t-2), \dots \dots X_0) = P(X_{i'}(t) \mid X_i(t-1)). \quad (30)$$

Definition 3.3. A property of a Markov chain that is a continuous time process $\{X(t); t \geq 0\}$ and is homogeneous for $0 < t_1 <. \dots \dots < t_y$, and $\{i_0, i_1, \dots \dots i_y\} \in \varphi$,

$$P(X_{i_0}(0), X_{i_1}(t_1), X_{i_y}(t_y)) = \pi(i_0) P_{i_0,i_1}(t_1) P_{i_1,i_2}(t_2 - t_1) \dots\dots\dots P_{i_{y-1},i_y}(t_y - t_{y-1}) \quad (31)$$

Eq. (31) clearly shows that the distribution of CTHMC (homogeneous CTMC) is governed by the initial distribution

$$\pi(i_0) = P(X_{i_0}(0)) \qquad i \in \varphi$$

and its transition probabilities depend on $\{i_0, i_1, \dots\dots i_n\}$

$$P(t) = [P_{i,i'}(t)]_{i,i' \in \varphi}$$

where $P(0) = I$ or I^t is the identity matrix. We note that for all $s > 0$

$$P(X_{i'}(t) \mid X_i(t-1)) = P_{i \to i'} \; or \; P_{i,i'} = P(X_{i'}(t+s) \mid X_i(s)) \quad (32)$$

The transitional probabilities depend on the difference of t, s and $t + s$ but not on the actual times $(s, t + s)$ that are responsible for the homogeneity in the Markov chains. We can see that Eq (32) equals to

$$P\{X_{i'}(t_{y+1}) \mid X_{i_0}(0), X_{i_1}(t_1)\ldots\ldots\ldots X_i(t_y)\} = P\{X_{i'}(t_{y+1}) \mid X_i(t_y)\} = P_{i,j}(t_{y+1} - t_y) \quad (33)$$

Therefore, CTMC has a Markov property alike DTMC (Discrete Time Markov Chain).

Theorem 3.1. The Chapman-Kolmogorov (CK) equations for any $s \leq s' \leq t$ can be obtained through Markov property and are given by

$$P_{i,i'}(s,t) = \sum_{all\ k} P_{i,k}(s,t)\, P_{k,j}(s',t) \quad (34)$$

for all $i, i' = 0,1,\ldots\ldots,$. While going from state X_i at, s, to $X_{i'}$ at t, it goes through transit state, X_k, at transit time, s'. In this case of CTHMC CK becomes

$$P_{i,i'}(\alpha) = \sum_{all\ k} P_{i,k}(\alpha - \alpha')\, P_{k,j}(\alpha) \quad (35)$$

for $0 \leq \alpha' \leq \alpha$. If,

$$P_{i,i'}(t + \partial t) - P_{i,i'}(t) = \textstyle\sum_{all\ k}\{P_{i,k}(t + \partial t - \alpha) - P_{i,k}(t - \alpha)\}\, P_{k,i'}(\alpha) \quad (36)$$

Dividing both sides of Eq. (36) by ∂t while taking $\partial t \longrightarrow 0$, $\alpha \longrightarrow t$ we get

$$\frac{\partial P_{i,i'}(t)}{\partial t} = \sum_{all\ k} Q_{i,k}(t) P_{i,k}(t) \quad (37)$$

Eq. (37) is also known as the *Kolmogorov backward equation* for all $i, i' = 0,1,\ldots\ldots,$. and in the matrix form it is

$$\frac{\partial P^{(t)}}{\partial t} = Q(t)P(t). \quad (38)$$

Similarly, the *Kolmogorov forward equation* for all $i, i' = 0,1,\ldots\ldots,$. can be deduced as

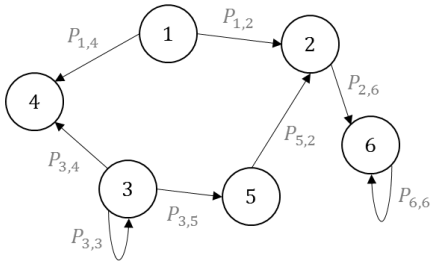
$$\frac{\partial P_{i,i'}(t)}{\partial t} = \sum_{all\ k} Q_{k,i'}(t) P_{i,k}(t) \quad (39)$$

and the matrix form is,

$$\frac{\partial P^{(t)}}{\partial t} = P(t)Q(t). \quad (40)$$

3.2.1 Sample Space for Biochemical Systems

For biochemical networks, we define the graphical representation of (A) in Figure 3.1 as a collection of states with arcs or straight lines as transitions to denote the reaction propensity values or transition probability. Each graph carries at least two states of the system and has a transition probability, as shown in (B) in Figure 3.1.



(A) Graphical

$$P = \begin{bmatrix} P_{1,1} & P_{1,2} & \dots & P_{1,6} \\ P_{2,1} & P_{2,2} & \dots & P_{2,6} \\ \vdots & \vdots & \vdots & \vdots \\ P_{6,1} & P_{6,2} & \cdots & P_{6,6} \end{bmatrix}$$

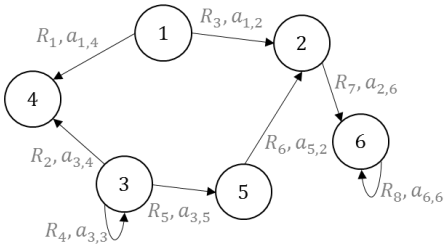
(B) Matrix

Figure 3.1. Representation of Markov chain with six states and a transition matrix [$\boldsymbol{P}$].

The transition or arc between state X_i and state, X_{i+1}, is ignored if $P_{i,i+1} = 0$, otherwise the values between all other states are represented as transition matrix [P] called a *stochastic matrix* or [$A_{i,j}$] if the propensity values are considered and satisfy following conditions:

$$P = \begin{cases} P_{i,j} \geq 0 \; for \; all \; i \; and \; j, & \text{condition (1)} \\ \sum_{all} j, P_{i,j} = 1 \; for \; all \; i, & \text{condition (2)} \\ \text{At least one non zero element in each column.} & \text{condition (3)} \end{cases}$$

Condition 1, implies that all $P_{i,j}$ values form non-negative matrix and the stochastic matrix is a subset of them. Condition 2, implies a guaranteed transition to at least one state and the elements in each row sum to 1. Condition 3 specifies that there will be at least one non zero element in each column as there are no states that last for a very short time (Gillespie, 1992b). If any Markov chain graph has $S^{\tilde{N}}$ states, then the transition matrix will be of $S^{\tilde{N}}$ x $S^{\tilde{N}}$ dimension with $a_{i,j}$ elements for any biochemical network. If $a_{i,j}$ represents the propensity of the reactions in biochemical networks as shown in (A) of Figure 3.2 then matrix $A_{i,j}$ represents the transition matrix of the network showing the states of the system, as in (B) of Figure 3.2.



(A) Graphical

$$A_{i,j} = \begin{bmatrix} a_{1,1} & a_{1,2} & \dots\dots & a_{1,6} \\ a_{2,1} & a_{2,2} & \dots\dots & a_{2,6} \\ \vdots & \vdots & \vdots\vdots & \vdots \\ a_{6,1} & a_{6,2} & \dots\cdot\cdot & a_{6,6} \end{bmatrix}$$

(B) Matrix

Figure 3.2. Representation of the Markov chain of biochemical network with reactions and relevant propensities and transition matrix $[A_{i,j}]$.

Matrix $A_{i,j}$ is important as transition between X_i and X_{i+1} is given by element $a_{i,i+1}$ or $a_{i,j}$. Every state in Fig (A) in Figure 3.2 retains only its most recent history due to the Markov property. Thus, the memory of the past is lost with the increasing transitions between the states. These sets of states and transitions, together, form the sample space φ of the biochemical systems. In terms of probability, φ is the combination of several events, Ɛ_y, that defines all possible outcomes. The number of Ɛ_y present in φ depends on the number of random processes connected through reactions in the system. The changes in the states are not fully predictable but are defined by probability distributions π and kinectic parameters. The probability distribution has some type of dependency, where the conditional distribution π_c of future states of the system gives some knowledge about the previous states, depending on the most recent state of the system.

The size of φ indirectly depends on the number of reactions and species in the system that create the processes and triggers Ɛ_y. For small biochemical systems having a small number of reactions (say ≤ 10), the size of φ is considerably smaller and the number of states can be identified easily within Ɛ_y. Any size of biochemical system defined by a number of states having Ɛ_y events should satisfy:

$$\text{Ɛ}_y \leq states. \tag{41}$$

For our proposed methods it is important to classify the states in terms of nodes and transitions and how they are bound to φ. In the next section we discuss the classification of states of Markov chains for any size of biochemical system along with some definitions.

3.2.2 States Classification of Markov Chain for Biochemical System

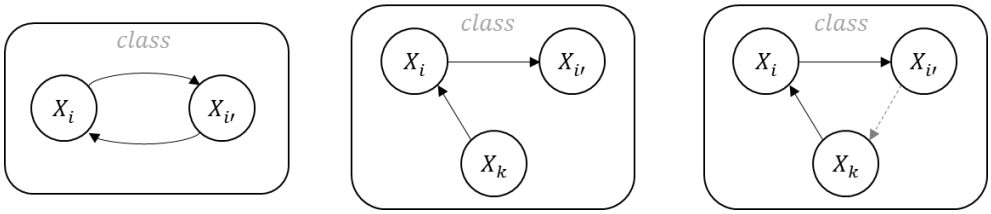
A chain of nodes, $\left\{N_1, N_2, \ldots \ldots . N_{S^{\tilde{N}}}\right\} \geq 1$, where transitions arc from N_{y-1} to N_y for each y, $1 \leq y \leq S^{\tilde{N}}$. If any biochemical system has R_M forward reactions then no nodes are repeated in the transition if it creates a simple *path* from one node to another in space. If any node is repeated, i.e. first and any intermediate node is the same, it creates a *cycle* and represents the backward reaction in the system. This also applies to the condition if the first and last node are the same and no other intermediate nodes are present in the system.

Definition 3.4. For the forward reaction in a biochemical system, state X_{i+1} is accessible and reached from state X_i (abbreviated as $X_i \longrightarrow X_{i+1}$ in $Dict$) by a transition denoted as an arc in the Markov chain graph from i to $i+1$ in space. If there is a backward reaction, state X_i is accessible and reached from state X_{i+1} (abbreviated as $X_{i+1} \longrightarrow X_i$ in $Dict$) by a transition denoted as an arc in the Markov chain graph from X_{i+1} to X_i in space. For example, in (A) in Figure 3.1, the path from node 3 to node 6 passes through node 5 and 2, so state 6 can be accessed from node 3.

Definition 3.5. If state X_i is reachable from X_{i+1} and X_{i+1} is reachable from X_i, then both these states tend to communicate and this denotes the reversible reaction of the biochemical system by $X_i \leftrightarrow X_{i+1}$.

Definition 3.6. If any finite state Markov chain representing biochemical system has transitions to a previous state X_i, and is accessible from other states of the system then X_i is recurrent state if $X_i \longrightarrow X_{i+1} \Longrightarrow X_{i+1} \longrightarrow X_i$. If the initial state X_i is not a recurrent state then it has non zero probability and the system never returns to X_i.

Definition 3.7. A class, φ', of non empty set of states where every $X_i \in \varphi'$ have access to set of states $X_i, \in \varphi'$ but not to other set of states $X_i, \notin \varphi'$. This property of accessibility to other states defines some characteristics of the states affiliated with the same *class* φ'.



(A) (B) (C)

Figure 3.3. Classification of transient and recurrent states.

If a set of states in a Markov chain $X_i, X_{i'} \in \varphi'$ are accessible to each other, i.e., $X_i \leftrightarrow X_{i'}$ (see Figure 3.3(A)). Now assume that if X_i is transient ($X_i \rightarrow X_{i'}$ but $X_{i'} \nrightarrow X_i$) then $X_k \rightarrow X_{i'}$ if $X_k \rightarrow X_i$ and $X_i \rightarrow X_{i'}$ (see Figure 3.3(B)). If $X_{i'} \rightarrow X_k$ (see Figure 3.3(C)), the transition from $X_{i'}$ to X_k could be extended to state X_i which will make X_i recurrent and would be contradictory; therefore, there should be no transition from $X_{i'}$ to X_k as that makes state X_k transient. This draws the conclusion that if one state in the φ' is transient, then all the states will be transient otherwise they will all be recurrent.

In section 3.3, we will discuss how a finite state Markov chain's sample space and its states can be visualised in terms of a Markov chain tree.

3.3 Markov Chain as a Markov Chain Tree

We define the Markov chain tree, $\mathbb{H}\mathbb{X}$, (Anantharam et al., 1989) as infinite, locally finite, connected to a special type of graph with a prominent vertex is called a parent node without loops or cycles. Let graph G_{mc} be a state-space of the finite state Markov chain with $\{P(X_i, X_{i'}) \mid X_i, X_{i'} \in G_{mc}\}$ transition probabilities meeting the condition $\sum_{X_{i'}} P(X_i, X_{i'}) = 1$, then the induced Markov chain tree is a combination of valued G_{mc} random variables with distributions inductively defined from $P(X_i, X_{i'})$ and with an initial state, $X_i \in G_{mc}$. That being the case, it is easy to expand this class of Markov random field through a Markov chain tree structuring for biochemical systems. Also, the Markov chain tree and Markov processes can be equated as explained by (Aldous et al., 1992) for the stochastic analysis.

Since we are interested in aperiodic states in expansion of state-space, we shall assume the reducibility or simplification of the G_{mc}; namely for each $X_i, X_{i'} \in G_{mc}$ through $\mathbb{H}\mathbb{X}$. Therefore, the reader is urged to concentrate on the case where G_{mc} is considered as a locally finite connected graph, but the transition probabilities of each state are not equal due to the propensities and parameters of different reactions in the biochemical system. Consequently a Markov chain tree, $\mathbb{H}\mathbb{X}$, can used to visualise a biochemical system process to exhibit a transition matrix as directed trees of its associated graph (Goutsias et al., 2013; Weber et al., 2012). In addition, the Markov chain tree, $\mathbb{H}\mathbb{X}$, generates a sample space for the system to represent the Markov processes associated with the Markov chain and the transition matrices of biochemical reaction networks. Further in this section, we will discuss the details we will need during representing Markov models on trees and working with graphs for state-space.

Let $\mathbf{X}_J$ be the finite set of cardinality $\{1,2 \ldots \ldots . K\}$ of a Markov chain $\mathbb{X}_c$, and A be the transition probability matrix associated with $\mathbf{X}_J$. A state-space is, substantially, a class of a set of states containing the unique state of the system, and the arcs between the states represent the transitions from the initial state to the end state. This transition defined as transient and communicating class in graphs. When all the transitions are combined, every state-space takes the form of a graph and creates the state-space of the system, as shown in Figure 3.4 below.

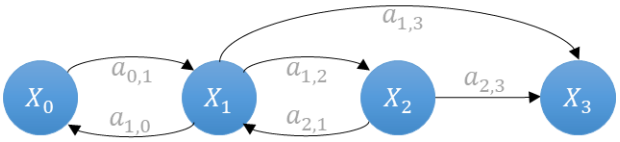


Figure 3.4. Markov chain graph showing forward and reversible reactions through four different states.

We can now associate chain $\mathbb{X}_c$ with the directed graph $G_{mc} = (\mathbf{X}_J, V_\mu)$, where $V_\mu = [v_1; v_2; \ldots\ldots v_\mu]$ and v_μ defines the transition from state X_i to $X_{i'}$ and is denoted as $v_\mu = \{(X_i, X_{i'});\ a_{i,j} > 0\}$. For every transition $(X_i, X_{i'}) \in \mathbf{X}_J$, then weight $\omega(X_i, X_{i'})$ is $a_{i,j}$.

Suppose G_{mc} has a cycle, which starts and terminates at some state, $X_i \in \mathbf{X}_J$. If there is a transition from X_i to $X_{i'}$, but we add a unique transition by creating a cycle from X_i back to itself and then consider the original transition from X_i to $X_{i'}$. This contradicts the uniqueness of the walk in tree (Diaconis et al., 1985). When relating this to the CTMC of a biochemical system process, the change in molecular population is defined by a stoichiometric vector, so, in G_{mc}, there must be at least one intermediate state that will send the system back to the previous state to create the cycle. This process categorises the *forward* and *backward* reactions given the initial state, X_0, of the system. The transient class of the transition leads the system to a unique state that defines the *forward* reaction in the system. Whereas, the communicating class of a transition defines the reversible reaction in the system. In this section, we will define such systems as transient class systems and communicating class systems. Large biochemical systems are usually a combination of both classes.

Definition 3.8. If U is any set, then cardinality is defined by the measure of number of elements in U, denoted by $|U|$. For example, set $U = \{5,7,8,10\}$ contains 4 elements, then cardinality of $|U|$ is 4.

A biochemical system is visualised as tree $\mathbb{H}\mathbb{X}$ (Anantharam et al., 1989) for the expansion of the state-space. A tree, $\mathbb{H}\mathbb{X}$, is a special form of graph in data structure constituting set of nodes and a collection of edges (or arcs) that each connects with an ordered pair of nodes. G_{mc} is considered as a directed tree, $\mathbb{H}\mathbb{X}$, and is rooted with $N_0 = (X_0, \mathfrak{d}_l)$ if it contains a unique walk to $N_i = (X_i, \mathfrak{d}_l + 1)$ and does not contain any cycles, while $X_i \in \mathbf{X}_J \backslash \{X_0\}$ has exactly one outgoing transition away from X_0 – in which case, it is called an arborescence, or making its transition towards $N_0 = (X_0, \mathfrak{d}_l)$ – in which case, it is called as anti-arborescence. An arborescence is a subset $\subseteq V_\mu$ that has one edge out of every node containing no cycles, and having a maximum cardinality.

If X_i and $X_{i'}$ be the states other than the initial X_0 state. There is a transition from X_i to $X_{i'}$, so X_i has at least one transition. Now, suppose X_i has two walks, (X_i, X_{i+1}) and (X_i, X_{i+2}). Concatenating these walks to the walks (X_{i+1}, $X_{i'}$) and (X_{i+2}, $X_{i'}$), respectively, we have two distinct changes in state from X_i to $X_{i'}$ in G_{mc} but in $\mathbb{H}\mathbb{X}$, this concatenation is not considered, which makes them Directed Acyclic Graphs (DAG) as shown in Figure 3.5. Most of the

biochemical models G_{mc} can be visualised as DAGs irrespective of the nature of reactions present in the model.

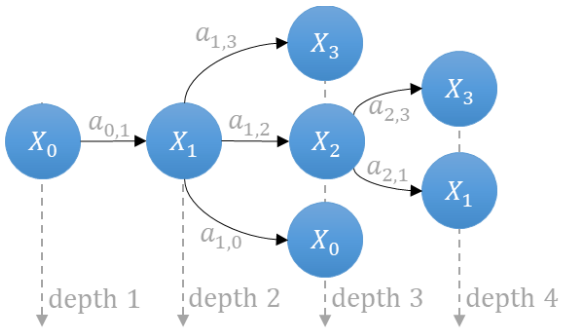


Figure 3.5. Equivalent tree of Markov chain graph, as shown in Figure 3.4. It is a special form of graph that has no cycle, no self-loops and depicts the state-space of the system in the form of a tree (DAG).

Figure 3.5 shows the equivalent tree of G_{mc} as shown in Figure 3.4. The trees are less complex as they have no cycles, no self-loops, and are still connected to depict the state-space. The weight of the tree containing all e edges is defined by $\omega(\text{HK}) = \prod_{e\in\text{HK}} \omega(e)$, where $\omega(e) = \omega\left(X_i, X_{i'}\right) = a_{i,j}$ is the weight of an edge starting from X_i and ending at $X_{i'}$ when $e \in$ HK (Gursoy et al., 2012). For systems having both *forward* and *backward* reactions, if $\mathbf{n}_J$ is the total number of nodes indexed by $\{1,2 \ldots \ldots . K\}$ same as states, $\mathbf{n}_K$ be the set of nodes carrying $\mathbf{X}_K$, and $\mathbf{n}'_K$ be the set of nodes carrying $\mathbf{X}'_K$ given N_0 root node of the tree HK, then the walk from one node to another node is given by:

$$\{f\left(N_i, N_{i'}\right), f\left(N_{i'}, N_i\right) \mid N_0\} \in \mathbf{X}_J, \tag{42}$$

and HK is formed by superimposing the forward transitions between states X_i and $X_{i'}$, with the reverse orientation, $X_{i'}$ and X_i, for backward reactions but these are graphically denoted as an individual edge from $N_i = (X_i, \mathrm{d}_l)$ to $N_{i'} = (X_{i'}, \mathrm{d}_l + 1)$ to $N_i = (X_i, \mathrm{d}_l + 2)$ in a tree. The N_i of $\mathrm{d}_l + 2$ can be renamed to a new node, N_{i+1}, as it is at different depth from N_i of d_l but storing the same state X_i. In the expansion repeated states are not considered in the domain; therefore, any node carrying a similar state will be considered the same regardless of the level and indexing. Consideration of trees for the state-space expansion in ISP will not only help in reducing the complexity but also improve the accuracy of the solution of Eq. (9) by identifying nodes carrying probable states. If the Markov chain graph starts in state $X_i \in \mathbf{X}_J$, then the mean number of transits to any state $X_{i'}$ converging to $\overline{a_{X_i,X_{i'}}}$ is given by the $(i, i')^{\text{th}}$ value of

$$\bar{\mathrm{A}} = \lim_{n\to\infty} \left(\frac{1}{n}\right) \sum_{k=0}^{n-1} A^k. \tag{43}$$

Let U be the set of all arborescences. Let $U_{X_i,X_{i'}}$ be the set of all arborescences which have a transition from X_i to $X_{i'}$ and $\left\|U_{X_i,X_{i'}}\right\|$ be the sum of the weights of the arborescences in $U_{X_i,X_{i'}}$ then according to Markov chain tree theorem (Anantharam et al., 1989),

$$\overline{a_{X_i,X_{i'}}} = \frac{\left\|U_{X_i,X_{i'}}\right\|}{\|U\|} \tag{44}$$

$\overline{a_{X_i,X_{i'}}}$ is probabilistic in nature. This nature is not only restricted to the systems having irreducible Markov chains in which graph G_{mc} is strongly connected while carrying probable state-spaces but also for systems that can be simplified by converting to a Markov chain tree

and then reducing the tree by ignoring the states having low probabilities in space φ (Anantharam et al., 1989; Gursoy et al., 2012). However, before a search problem can be solved, it must be represented as a state-space in terms of the Markov chain tree. Therefore, in section 3.4, we will discuss the *problem state-space* model, *problem instance*, *problem state-space graph* or tree and its objects where a searching technique is to be applied.

3.4 Problem State-Space Model of Biochemical networks

The *problem state-space* of a biochemical network is the domain in which an exploration and search of the end state takes place. This problem of the state-space consists of set of states and operators that change the current state to a new state. The initial state and end state of the biochemical system together with problem space, brings into existance the *problem instance* where all the searching should take place. For example, in a puzzle of 16 blocks the states would be the different possible permutations of the tiles, where a function called operators slide a tile into the blank position within these blocks. Similarly, in biochemical systems, finding and applying operator sequences that map an initial state to the end state is the problem solving task. The end state is given explicitly or implicitly depending on the nature of the system, such as in a 16 block puzzle when the end state is given explicitly; whereas, in the Eight Queen problem (Bell et al., 2009) the end state is given implicitly, as described by properties satisfied by the end state.

The *problem state-space graph* or converted to a *tree* is used to represent the Markov process of the biochemical systems in the problem space. The set of states in the *state-space* of the graph are represented by *nodes*, and the transition function (operator) by arcs between the nodes. All the arcs in the state-space are directed which defines a reaction with its rate constant irrespective of whether their corresponding transition function is invertible in the case of a reversible reaction. The task for this problem is to find the set of states in the path from initial state to end state that carry the majority of the probability mass of the system.

In most of biochemical systems the Markov process corresponds to graphs with more than one directed arc between a pair of nodes having a set of states and, for simplicity, we visualise this in terms of *trees*, where the root (at *level 1*) of the tree represents the initial state of the system. In the tree visualisation, the multiple directed arcs connecting nodes are represented by duplicate nodes and this is the simplication cost of the graph. When any state in the nodes is reached by two different transitions then it will be represented by these duplicate nodes and a unique state of the biochemical system. The benefit of using trees is that it greatly simplifies the problem of searching in the state-space and, in the absence of cycles, visualisation of the biochemical systems remains an easy task.

In terms of *artificial intelligence* (*AI*) (Barr et al., 1983; Chijindu et al., 2012; Korf et al., 1985, Korf et al., 1996) search standards, for a real-world example, in case of chess game, the corresponding graph has more than 10^{40} nodes, whereas a twenty four block puzzle has more than 10^{25} nodes. In both cases, the estimated minimum number of states are 10^{40} and 10^{25}

respectively. As a result of this explosion, the *problem state-space* graph of a biochemical system is never defined explicitly by a set of states, but implicitly by the initial state of the system, and the set of functions that generate new states from the current states. The function depends on the conditions, as discussed in section 4.2, and applied only on the states that do not produce any negative stoichiometric indexing. For example, in any biochemical system, if state A and B have stoichiometric indexing leading to [2 5 1 0] and [2 -1 0 6], respectively, then state B will not be considered for expansion as it does not satisfy the function conditions (discussed further in section 4.2).

The size of a biochemical system is not expressed in terms of the number of nodes in the problem space graph but by the number of states stored by these nodes. The ratio of the total number of walks between different nodes and the total number of nodes explored defines the average transitioning factor (Ŧ). Together with Ŧ, the depth of states determine the efficiency of the expansion algorithm.

$$Ŧ = \frac{\text{Total no. of walk between different nodes}}{\text{Total no. of nodes explored}} \tag{45}$$

As Ŧ defines the average number of child of a given node, then the end state of the problem instance is the cost of transition, ($Ć_{N_i,N_i'}$), from state X_i to $X_{i'}$ in space, where $X_{i'}$ is the end state. If the end state (X_{10}) of a system were in the last row (i.e. depth 4) of Figure 3.6, then the depth of the problem state-space graph is represented by the X_i (from node N_1 at depth 1) at the root would have three moves ($N_1 \rightarrow N_7 \rightarrow N_8 \rightarrow N_{10}$).

In section 3.5, we will extend our discussion to AI standards for Markov chain trees or graphs (G_{mc}) and derive a function (a.k.a operator) that can define the next step (i.e. state) and other information in expansion implicitly.

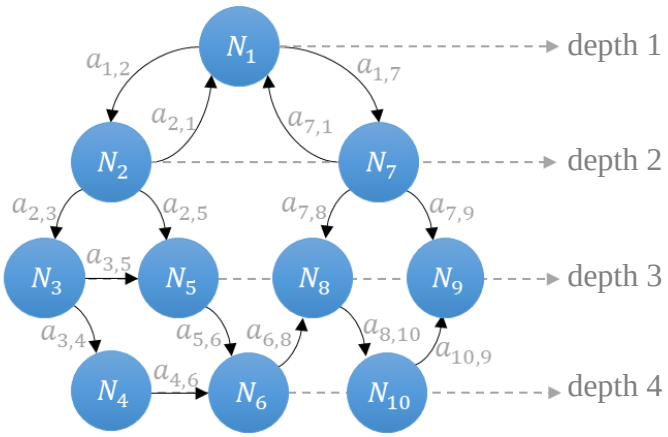


Figure 3.6. Markov chain graph (G_{mc}) with n_J nodes carrying $\approx X_J$ states and the arcs showing transitions between them while, together, they form a Markov process.

3.5 Intelligent Search and Tracking

Much of AI is about searching that can be applied to many areas, such as systems biology, biochemical networks for representing knowledge as well as to study the biochemical systems. Over recent years, awareness has greatly increased the understanding of large biochemical systems by solving the CME. Similarly, the expansion of state-space for large biochemical systems can be formulated as a problem search in AI. For this purpose, we will employ a Markov chain as a Markov chain trees or graph (G_{mc}) (as discussed in section 3.3) as a state-space for biochemical systems. Factors, such as type of state-space search problem and how this problem knowledge can be represented, decides the structure of the search algorithm.

3.5.1 Artificial Intelligence for CME

Although not common yet, AI standards can be adopted and applied to explore the state-space for large biochemical systems to solve the CME. AI is the field of study of the intelligent agents (Rudowsky, 2004) of a system that perceives and take actions to successfully achieve goals. Most of the problems can be formulated as searches and solved by reducing to one of searching a graph or a tree. AI techniques (Alan Bundy, 1997; Nelson et al., 1992) set the standard for treating the data structures (trees or graphs) to find the goal state. However, in our stochastic *problem state-space* there is no single overall goal state (or end state) that defines the solution of the CME because the size of the state-space of large biochemical systems are usually considered as infinite. Furthermore, the explosion of state-space is still a major issue in biochemical systems; especially in large models. To visualise the Markov chain graph or tree as a *problem state-space* in biochemical systems, for the solution of the CME, we must specify the successor operator S_{uc}, in detail to define the sequence of actions leading from the initial to the multiple goal states at *different time intervals* that then lead to the solution at the required time.

Definition 3.8. A tuple (X_0, *Action*, S_{uc}, $\mathbf{X}_J$) defines the transition in the system, where $X_0 \in \mathbf{X}_J$ is the initial state, $Action$ is a finite set of labels (action) for operator S_{uc} that maps the present state with the future state, $S_{uc} \subseteq \mathbf{X}_J \times Action \times \mathbf{X}_J$ is the relation of transition and $\mathbf{X}_J$ is a finite set of states, which is unknown a priori. A transition (X_i, a_μ, $X_{i'}$) $\in S_{uc}$, denotes $X_i \xrightarrow{a_\mu} X_{i'}$, specifies that biochemical system can move from state X_i to $X_{i'}$ by performing $Action\ a_\mu$. Here, a_μ denotes the propensity of R_M reaction leaving state X_i at depth $\mathfrak{d}_l$ in G_{mc}.

In terms of AI, we will now define the *state-space* as a set of states in the system we can get to by applying S_{uc} to explore new states of a biochemical network. S_{uc} can be applied explicitly, which maps the current state to the output states or defined implicitly that act on current state and transform into a new state. In the *problem state-space graph* for biochemical networks, we will not define the goal state (or end state) explicitly but it is required to be defined by S_{uc} implicitly in intervals based on nature (*fast, slow, reversible and irreversible*) of reactions in the system and duration of expansion to introduce the stochasticity of the system. This should systematically expand the state-space from $\mathbf{X}_K$ at t to $\mathbf{X}_{K+1}$ at $t+1$ by going through each node $\mathbf{n}_J$ at depth $\mathfrak{d}_l$ of the Markov chain graph to evaluate the Markov processes, which aims to occupy most of the probability mass during $[\mathbf{X}_K + \mathbf{X}_{K+1}]$, and can be solved for probability distribution at $t+1$.

Let $\mathbf{X}_J$ be the finite set of states and $G_{mc} = (\mathbf{X}_J, V_\mu)$ be the Markov chain graph on $\mathbf{X}_J$ associated with $A = [a_{i,j}]$, given X_0 as the initial state and $\mathbf{X}_K$ as the set of the explored state, where $X_0 \in \mathbf{X}_K$ then implicit successor is defined as,

$$S_{uc} \longrightarrow V_\mu(\mathbf{X}_K(t)), \tag{46}$$

defines the new states of the system, where, V_μ is the set of stoichiometric vectors v_μ function defining the state transitions from any present state $X_i \in \mathbf{X}_K$ to a new state $X_{i'} \notin \mathbf{X}_K$. The sample space in the graph contains the unique state of the system stored in a transition matrix, which satisfies the conditions in Eq. (42). This transition matrix is a compressed row format (CSR) (Koza et al., 2012; Lawlor et al., 2013) based on the index of row $\longrightarrow$ column delimited by commas generating the dictionary $Dict$ of the model that defines the transitions between nodes in the state-space and mapping of states. Through S_{uc}, we can know nothing more than the neighbours (child nodes) of the current node (states reachable through a single reaction), then we consider these neighbours (child nodes) as our only goal states and there can be many in numbers. In such a situation, we call our search trails as *blind* or *uninformed search*. In next section 3.5.1.1 , we will work on the infrastructure of an *uninformed search* to define the type of data structure we will be dealing with.

3.5.1.1 Infrastructure for Searching

A data structure is required to retain the search track in the graph for *problem state-space* expansion. For each node, N_i, of the tree, we create a structure consisting of five elements:

(1) N_i.State: represents state X_i in the state-space corresponding to N_i;

(2) N_i.Parent: represents the parent node of child node N_i;

(3) N_i.Depth: represents the depth of state state X_i;

(4) N_i.Cost: represents the cost $\mathfrak{C}_{N_i,N_i'}$ of the transition from N_i to N_i' in the state-space;

(5) N_i.Action: represents the action applied via S_{uc} on parent node to reach N_i.

To explore the new states in the system, we will consider the initial state $state(N_1) = (X_0, \mathfrak{d}_l)$ as input to the successor, S_{uc}. Once the expansion is initiated, the $Dict$ will temporarily (in run-time) store the information for the transition from one node to another in the state-space that binds to the reaction propensities a_μ. This shift is denoted by an arrow $\longrightarrow$, which shows multiple transitions from the parent nodes to child nodes containing the end state. The set of nodes $\mathbf{n}_J = \{N_1, N_2, \ldots\ldots N_{S^{\tilde{N}}}\}$ is a data structure that incorporates the Markov chain graph G_{mc}. We will explore all the nodes that store the set of states $\mathbf{X}_K$ as well as some additional information about the state, such as depth and transition cost, from one state to another in the system. If a set of $states(\mathbf{n}_J) = \mathbf{X}_J$, then $\mathfrak{C}_{1,N_i'}$ is the transition cost to reach $state(N_i') = X_{i'}$ from $state(N_1) = X_1$ and $depth(\mathbf{n}_J) = \mathfrak{d}_l$ defines the depth of the set of nodes in G_{mc}, then the standard relation between a set of nodes and a set of states is given by $\mathbf{n}_J = (\mathbf{X}_J, \mathfrak{d}_l)$ or $\mathbf{n}_J = (\mathbf{X}_J, \mathfrak{d}_l, \mathfrak{C}_{N_i,N_i'})$ and the standard relation between a single node and a single state is given by $N_i = (X_i, \mathfrak{d}_l)$ or $N_i = (X_i, \mathfrak{d}_l, \mathfrak{C}_{N_i,N_i'})$ if the transition cost is considered.

For example, Figure 3.6 shows the Markov chain graph, G_{mc}, with $\mathbf{n}_J = 10$, $\mathfrak{d}_l = 4$ and its equivalent tree HX is shown in Figure 3.7 with $\mathrm{n}_J = 15$, $\mathfrak{d}_l = 5$. In the tree nodes $N_1 = N_{11} = N_{12}$ carries the same state, X_1 at $\mathfrak{d}_l = 1, 2$ and 3, respectively, where walk $N_2 \rightarrow N_{11}$ and $N_7 \rightarrow N_{12}$ represents the backward reaction of the forward reaction represented by walk $N_1 \rightarrow N_2$ and $N_1 \rightarrow N_7$, respectively.

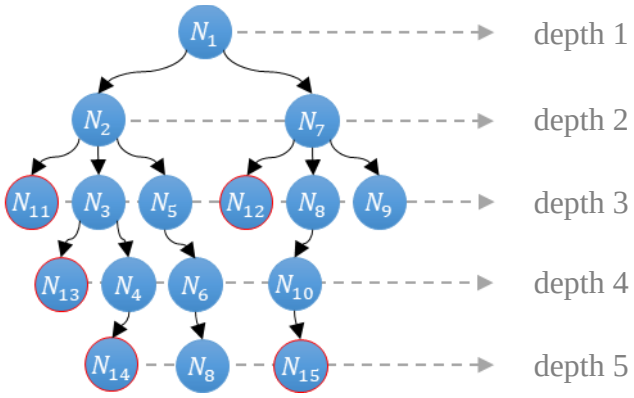


Figure 3.7. Equivalent tree ℍ of G_{mc} (see Figure 3.6) as DAG representing the state-space of the system.

The set of nodes with states are represented as

$$\mathbf{n}_{1,2,\ldots.10} = \left(\mathbf{X}_{1,2\ldots\ldots K}, \mathfrak{a}_{1,2,3,4}\right) \text{ or} \tag{47}$$

$$\mathbf{n}_{1,2,\ldots.10} = \left(\mathbf{X}_{1,2\ldots\ldots K}, \mathfrak{a}_{1,2,3,4}, \mathfrak{C}_{N_i,N_i'}(min)\right) \tag{48}$$

In general, the transition cost, $\mathfrak{C}_{N_i,N_i'}$, is defined as:

$$\mathfrak{C}_{N_i,N_i'} = a_{1,2} + a_{2,3} + \cdots + a_{N-1,N}. \tag{49}$$

$\mathfrak{C}_{N_i,N_i'}$ is the summation of all the propensities a_{μ} of the R_M reactions that takes the system to its final state. For example, $\mathfrak{C}_{N_1,N_{10}}$ to expand to $state(N_{10}) = X_{10}$ of Figure 3.6 is given by

$$\mathfrak{C}_{N_1,N_{10}} = \begin{cases} \textbf{\textit{Path}}\ \mathbf{1}\text{:}\ a_{1,2} + a_{2,3} + a_{3,4} + a_{4,6} + a_{6,8} + a_{8,10} \\ \textbf{\textit{Path}}\ \mathbf{2}\text{:}\ a_{1,2} + a_{2,3} + a_{3,5} + a_{5,6} + a_{6,8} + a_{8,10} \\ \textbf{\textit{Path}}\ \mathbf{3}\text{:}\ a_{1,2} + a_{2,5} + a_{5,6} + a_{6,8} + a_{8,10} \\ \textbf{\textit{Path}}\ \mathbf{4}\text{:}\ a_{1,7} + a_{7,8} + a_{8,10} \\ \textbf{\textit{Path}}\ \mathbf{5}\text{:}\ 0, \text{if not reachable} \end{cases}$$

If these are the possible paths for the expansion that expands $\mathbf{X}_K$ at every iteration then $\mathfrak{C}_{N_1,N_{10}}(min)$ will be defined by the only path that has the lowest $P^{(t)}(\mathbf{X}_K')$. This can be generalised as follows:

$$\mathfrak{C}_{N_i,N_i'}(min) \propto \frac{1}{P^{(t)}(\mathbf{X}_K)}, \tag{50}$$

which means that in order to have a minimum cost of the expansion for the optimal domain $\mathbf{X}_K$ at least one path should have states with high probabilities for $\mathbf{X}_K$ and; therefore, it is beneficial to follow the path with $\mathfrak{C}_{N_i,N_i'}(min)$, which leaks the minimum probabilities of the system.

For large biochemical models there exists infinite cases when the node is unreachable from the initial or another node, and such cases are ignored when $\mathfrak{C}_{N_i,N_i'}(min) = \{Path\text{: } 0\}$ because some probabilities are always dropped in the approximation. Therefore, $\mathfrak{C}_{N_i,N_i'}(min)$ as defined by the lowest $P^{(t)}(\mathbf{X}_K')$ is strictly limited to,

$$P^{(t)}(\mathbf{X}_K) > \mathfrak{C}_{N_i,N_i'}(min) > 0, \tag{51}$$

Upon expanding the root node N_1, we expand the child nodes carrying new states, and then the child-child nodes are explored. The walk between nodes $N_i \xrightarrow{V_\mu(\mathbf{X}_K(t))} N_{i+1}$ is defined by dictionary $Dict$ and this represents the occurrence of R_M reactions through M elementary channels. For Figure 3.6, the typical form of dictionary is given below:

$$\begin{aligned} D = ([1 \longrightarrow 2,7], [2 \longrightarrow 1,3,5], [3 \longrightarrow 4,5], [4 \longrightarrow 6], [5 \longrightarrow 6], [6 \longrightarrow 8], \\ [7 \longrightarrow 1,8,9], [8 \longrightarrow 10], \ [9 \longrightarrow Nil], [10 \longrightarrow 9]), \end{aligned} \tag{52}$$

and is indexed with the propensities, $[a_{i,j}]$, for all the R_M reactions. As the propensities are changing by $\Delta a_{i,j}$, we will consider the recent values of $a_{i,j}$ in every iteration of ISP that corresponds to the reactions involved. To make the $\mathfrak{C}_{N_i,N_i'}(min)$ feasible for any type of biochemical system (*stiff, non-stiff*) for capturing probable states, it is important to consider the cost of expansion for small t_{step} (time step). This may be because there are some cases when $\mathfrak{C}_{N_i,N_i'}(min)$ to reach two or more different child nodes are equal or very close to each other. In addition, we intend to expand the state-space in the direction of carrying states with high probability mass. To achieve this, we need some provision made to treat or convert our *uninformed search* to an *informed search* infrastructure at run-time to have intuitive knowledge beyond our reach. Figure 3.8 shows the limits of our visibility in the state-space.

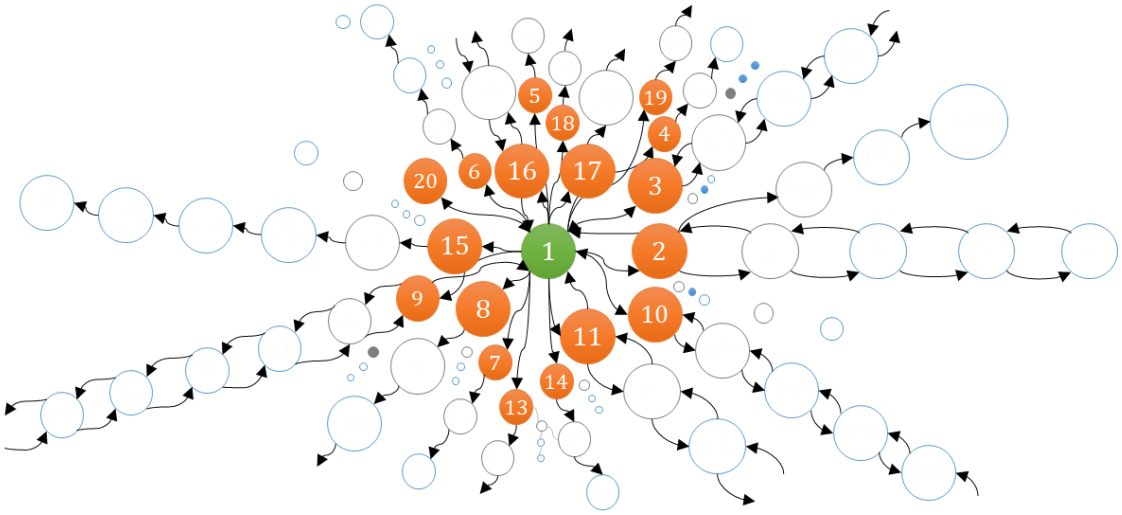


Figure 3.8. Limits of our visibility in the state-space before expansion, visualised by a Markov chain graph, where ● is the initial node and ● nodes are directly reachable nodes from the initial node when exactly one R_M occurs. When a further R_M occur, the system jumps to other ○ nodes.

Consequently, it is important to track the reactions having high propensity function values. In such, cases it is difficult to determine the direction of the expansion; therefore, in following section 3.5.2, we first develop the post successor function on Bayes' theorem (Fahidy, 2011; Schlecht, 2014) to priortise the expansion direction based only on those reactions that can be triggered at a particular time point and then, in Chapter 4, we will layout the direction strategy with the depth and bounds of the expansion.

3.5.2 Bayesian Likelihood Node Projection Function

Bayesian methods (Fahidy, 2011; Manoukian, 1986) are based on the principle of linking prior with the posterior probability through Bayes' theorem (Fahidy, 2011; Schlecht, 2014). For an event, the posterior probability is the improved form of the prior probability, via the likelihood of finding factual support for a valid fundamental hypothesis. Therefore, we will employ the standards of Bayes' theorem to develop a function targeted to advocate the quality of the expansion based on R_M reactions active in the network at any particular moment. For a concise definition for the purpose of fundamentals, refer to Appendix D.

To improve the quality of expansion through a projection function, one may find useful to remove the set of states having low probabilities before calculation of Eq. (7); however, this will compromise the accuracy as the step error will increase at every t. Moreover, it will greatly affect the solution, as defined at t_f (at which solution is required), for large dimension systems having large state-spaces, as the step error will be much higher due to dropping of probabilities without solving Eq. (7). In large systems, any species may change its behaviour after a certain number of firing of reactions triggering inactive reactions in the network that will affect the probabilities of the states. If the change in behaviour increases the probabilities of certain states, then removing them in an earlier stage is not suitable.

Through the *Bayesian Likelihood Node Projection* ($BLNP$) function, we seek to predict the posterior probability based on the parent state's probability and calculate the likelihood of the occurance of reactions that will take the system from the present state to the future state. Through $BLNP$ we can capture a sense of knowledge about the situation of the system that will help us to make better predictions about the future state and still keep good accuracy of the solution with optimal domain.

Statement of the Problem: Assume that $state(N_0) = X_0$ is the present state of the system, and further we have two choices to make among $state(N_1) = X_1$ and $state(N_2) = X_2$ based on reactions and copy counts (as shown in Figure 3.9) for the future state of the system.

It is clear that species with higher copy counts will interact quickly compared to other species with lower copy counts. Depending upon the association of higher copy count species within reactions R_1 or R_2, the system will jump to X_1 or X_2 respectively. Now we make an assumption that there are almost equal propensities of reactions, as shown in Figure 3.10, at any time t of the expansion phase.

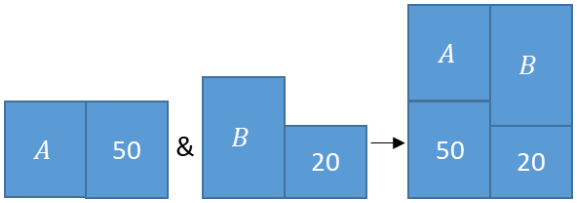


Figure 3.9. *A* and *B* with copy counts 50 and 20, respectively, creating a system of two species, *A* and *B* with 70, total copy counts.

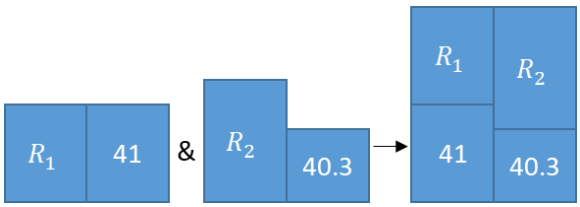


Figure 3.10. Two states X_1 of *A* and X_2 of *B* with almost equal propensities. This creates a dilemma when choosing the direction for expansion.

In such a situation it is important to decide the direction of the expansion when choosing the future state of the system as any reaction can occur and take the system to any new state. To understand this situation more closely on a node level, we assume a Markov chain graph as shown in Figure 3.11 of this system having almost the same number of species count. In Figure 3.11, the expansion is at intermediate position as the initial state $state(N_0) = X_0$ is already expanded and now the expansion of $state(N_2) = X_2$ is to be undertaken. To calculate the likelihood of the occurrence of reactions R_1, R_2, R_5, we consider the propensities $a_{i,j}$ as a parameter and $\Delta a_{i,j}$ depends on the kinetic parameter of the reaction. To assign weight to our belief, we deduce a function that will calculate the probability of occurrence of reactions and prioritise the expansion in order from reactions resulting in states with high probabilities to reaction giving states with low probabilities. It is important to note that none of the probabilities will be removed before the calculation of Eq. (9). With this function the likelihood of occurance of R_M can be computed.

We consider each node as a junction of the prior reactions $\{R'_1 \dots \dots R'_M\}$ with propensities $\{a'_{1,N} \dots \dots a'_{N',N}\}$ having prior likelihood values $\{b'_{1,N} \dots \dots b'_{N',N}\}$ and future reactions $\{R_1 \dots \dots R_M\}$ with propensities $\{a_{1,N'} \dots \dots a_{N,N'}\}$ having likelihood values $\{b_{1,N'} \dots \dots b_{N,N'}\}$, as given in Figure 3.12

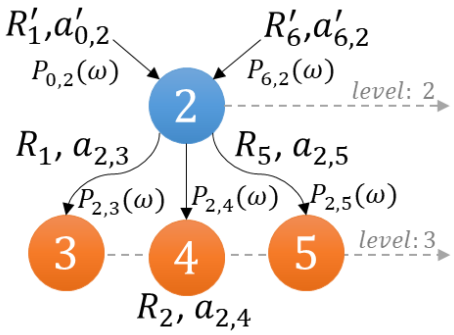


Figure 3.11. Current $state(N_2) = X_2$, and future $states(N_{3,4,5}) = (X_{3,4,5}, d_{l=3})$ with corresponding reactions R_1, R_2, R_5 and assumed propensities $a_{2,3} = 38, a_{2,4} = 39, a_{2,5} = 40$, respectively, at any time t, given $b_{0,2} = 0.4871$, $b_{6,2} = 0.5128$.

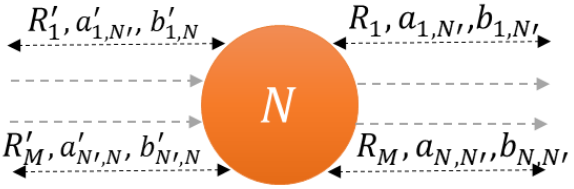


Figure 3.12. Node N as a junction of forward and backward reactions R_M, where $a'_{1,N'}, \ldots, a'_{N,N'}$ are propensities of the prior reactions' and $b'_{1,N}, \ldots, b'_{N',N}$ are the likelihood of the prior reactions.

To calculate the likelihood of the reactions, it is important to have prior information about the occurrence of reactions. If the expansion is to be done at initial node $state(N_0) = X_0$ (at level 1) then the prior likelihood value $b'_{N,N}$, is considered as the initial probability or as ≈1. Once the initial node is explored, we can calculate the likelihood of the reactions inductively. To calculate the probabilities $b_{1,N'}, \ldots, b_{N,N'}$ of the occurrence of $R_1, \ldots, R_M$, we first calculate the weighted probabilities $P_{N,1}(\omega)$,……,$P_{N,N'}(\omega)$ of a system leaving any state by:

$$\frac{a_{\mu}(X - v_{\mu})}{\sum_{\mu=1}^{M} a_{\mu}(X - v_{\mu})} = \frac{\text{Propensity of } R_M \text{ reaction leaving state } X_i \text{ at } \mathfrak{d}_l}{\text{Sum of propensities of all the reactions leaving state } X_i \text{ at } \mathfrak{d}_l} \tag{53}$$

and multiply with the prior probability $b'_{1,N} \ldots \ldots b'_{N',N}$ of the system. This will calculate the likelihood inductively in exactly R_M that is responsible to shift the system to present $state(N_i) = X_i$ at t, leading to a function,

$$b(N_{N1,..NM}|b'_{1,N....N',N}) = \frac{a_{\mu}(X - v_{\mu})}{\sum_{\mu=1}^{M} a_{\mu}(X - v_{\mu})} * b'_{N',N}(X - v_{\mu})' \tag{54}$$

where,

$$P_{N,1....N,N'}(\omega) = \frac{a_{\mu}(X - v_{\mu})}{\sum_{\mu=1}^{M} a_{\mu}(X - v_{\mu})}, \tag{55}$$

$$b(N_{N1,..NM}|b'_{1,N....N',N}) = P_{N,1....N,N'}(\omega) * b'_{N',N}\big(X - v_{\mu}\big)'. \tag{56}$$

Once $b(N_{N1,..NM}|b'_{1,N....N',N})$ is calculated for all the adjacent nodes, the values are arranged in descending order. Every value is bound to one reaction and represents the likelihood of the occurance of that reaction that takes the system from the present node to the child nodes. Based on likelihood values (highest to lowest) the corresponding reactions are considered one by one and labelled as *true* events for expansion. For example, if system has R_1, R_2, R_3 reactions that bound to $BLNP$ likelihood values in order from highest to lowest, respectively, then it is considered as three events of the system that takes the system to new state. When R_1 is considered for expansion, R_2 and R_3 are labelled as *false* events and R_1 as the *true* event. When second highest $BLNP$ likelihood value is considered, which is for R_2, then it is labelled as a *true* event and the others, R_1, R_3 are labelled as false. Similarly, the last and lowest

$BLNP$ likelihood value is for R_3, which is labelled as a *true* event and the others as *false* events. All states are added in the domain in order from the 1st *true* event to the 3rd true event. The Eq. (56) of probabilities $b(N_{N1,..NM}|b'_{1,N....N',N})$ is what we call a $BLNP$ function.

In Figure 3.11 Markov chain tree, for selection (see Appendix D for example) present at level 2 (assuming that the initial node is already expanded), we calculate the weighted probability of a system leaving $state(N_2) = X_2$ by:

$$P_{2,3}(\omega) = \frac{a_\mu(X - v_\mu)}{\sum_{\mu=1}^{3} a_\mu(X - v_\mu)} = 0.3247$$

similarly, $P_{2,4}(\omega) = 0.3333$ and $P_{2,5}(\omega) = 0.3418$.

At level 2, the conditional probability of the occurrence of reaction R_1 given the probability of occurrence of reaction R_1 at level 1 is given by:

$$b\left(N_{2,3}\middle|b'_{0,2}\right) = \frac{a_\mu(X - v_\mu)}{\sum_{\mu=1}^{3} a_\mu(X - v_\mu)} * b'_{0,2}(x - v_\mu)',$$

and, similarly, the occurrence of reaction R_1 at level 2 given the probability of occurrence of reaction R_6 at level 1, is given by:

$$b\left(N_{2,3}\middle|b'_{6,2}\right) = \frac{a_\mu(X - v_\mu)}{\sum_{\mu=1}^{3} a_\mu(X - v_\mu)} * b'_{6,2}(x - v_\mu)'.$$

If at level 1, $state(N_1) = X_1$ and at level 2, $state(N_2) = X_2$ are explored through R_1 then we say that this is a *true* event and temporarily consider other events *false* events with respect to the other reactions. Such a condition holds *true* for the other two cases, when, at level 1, $state(N_1) = X_1$ is explored through R_1 followed by an exploration of $state(N_2) = X_2$ through either by R_2 or R_5. Given $b'_{0,2}(X - v_\mu)'$ and $b'_{6,2}(X - v_\mu)'$, we will calculate the likelihood of all the R_M events, as given in Table 3.1. The likelihood values of future reactions cannot be equal as they are based on the probabilities of occurrance of prior reactions.

Table 3.1. Events with the likelihood of the future reactions. Where, *True* events define the expansion of nodes.

$b_{N,N'}$	$N_{0,2}$	$N_{6,2}$	$b_{N,N'}$(Value)	R_{next}
$b(N_{2,3}\|b'_{0,2})$	True	False	0.1581	$R_{1,1}$
$b(N_{2,3}\|b'_{6,2})$	False	True	0.1665	$R_{6,1}$
$b(N_{2,4}\|b'_{0,2})$	True	False	0.1623	$R_{1,2}$
$b(N_{2,4}\|b'_{6,2})$	False	True	0.1709	$R_{6,2}$
$b(N_{2,5}\|b'_{0,2})$	True	False	0.1664	$R_{1,5}$
$b(N_{2,5}\|b'_{6,2})$	False	True	0.1752	$R_{6,5}$

From Figure 3.11 and Table 3.1, we can infer, based on the prior reactions for R_M, where $M = \{1,6\}$ as such that:

Case 1 (R_1): At level 2, if the prior reaction is R_1 and holds a *True* event for $N_0 \longrightarrow N_2$ then:

$$b(N_{2,5}|b'_{0,2}) > b(N_{2,4}|b'_{0,2}) > b(N_{2,3}|b'_{0,2})$$

as per $b_{N,N'}$ and likelihood of occurrence of reactions will be in the order $R_5 > R_2 > R_1$.

Case 2 (R_6): At level 2, if the prior reaction is R_6 and holds a *True* event for $N_6 \longrightarrow N_2$ then

$$b(N_{2,5}|b'_{6,2}) > b(N_{2,4}|b'_{6,2}) > b(N_{2,3}|b'_{6,2})$$

as per $b_{N,N'}$ and likelihood of occurrence of reactions will be in the order $R_5 > R_2 > R_1$.

There will be M number of cases (equal to elementary chemical reaction channels) if there are R'_M prior reactions in the system that bring the system to the current node and the value of the likelihood will change based on $b'_{N',N}(X - v_\mu)'$. The $BLNP$ function cannot be used standalone for expansion because it only assign weightage to direction for expansion. Therefore, in Chapter 4, we will derive the condition for our expansion strategies to work with the Markov chain graph state-space and define the criteria for the formation of bounds (domain formed at anytime t) with time. The $BLNP$ function with expansion strategies will choose the probable states in large biochemical systems where it is important to capture the moments at time t that define the behaviour of a system. $BLNP$ will be useful in identifying the most active reactions in the system while guiding the expansion towards the set of states with high probability mass.

3.6 Complexity of Optimal Solutions

Computational complexity, such as *Time* and *Space,* depend on the size of the Markov chain tree and quantifies the amount of time and memory required to run the algorithm as a function of the length and weight of the input that is measured together in the *Order of growth*. The space complexity is the summation of static (constants and instructions) and variable (run-time variables, recursion) parts of the solution. Computing speed and memory has been greatly improved by recent advancements in computer systems (Devi et al., 2011; Nasar, 2016). Therefore, for the algorithm performance review, our prime focus will be on *Time* (run-time) first and then *Space* (memory).

The total amount of time required in problem solving is equivalent to the summation of compilation and run-time. It is also not necessary that routines should be compiled before running. The *time complexit*y will be the focus in our study when it comes to measuring computational performance. To interpret the actions of the algorithm for inputs (particularly large values) it is important to discuss the theory of *asymptote analysis* (Ω, O, Θ) as preliminary. The *asymptote* is the curve (or a line) that a graph proceeds towards but never converges with. The *asymptote* of a curve is particularly a line, such that the interspace between line-curve tends to 0, while approaching ∞ (or higher value).

For the *Order of growth* we have lower limit which is indicated by $\Omega(Algo(y))$, where Ω is *Big-omega*; therefore, definition:

Definition 3.4. Let $f(y)$ and $Algo(y)$ be the positive real value functions, where $f(y)$ is a function of integer y, y_0, with positive constant c. Then $f(y)$ is $Algo(y)$ such that $\Omega(Algo(y))$ defines the *asymptotic lower limit* and a set of functions,

$$\Omega\big(Algo(y)\big) = \{0 \leq c * Algo(y) \leq f(y) \qquad for\ all\ y \geq\ y_0\}. \tag{57}$$

Similarly, for the *Order of growth* we have an upper limit, which is indicated by $O(Algo(y))$, where O is a *Big-oh*; therefore, definition:

Definition 3.5. Let $f(y)$ and $Algo(y)$ be the positive real value functions, where $f(y)$ is a function of integers y, y_0, with a positive constant c. Then $f(y)$ is $Algo(y)$ such that $O(Algo(y))$ defines the *asymptotic lower limit* and a set of functions,

$$O\big(Algo(y)\big) = \{0 \leq f(y) \leq c * Algo(y) \qquad for\ all\ y \geq\ y_0\}. \tag{58}$$

Lastly, for the *Order of growth* we have a tight limit, which is indicated by $\Theta(Algo(y))$,

where Θ is *Big-theta*; therefore, definition:

Definition 3.6. Let $f(y)$ and $Algo(y)$ be the positive real value functions, where $f(y)$ is a function of integers y, y_0, with positive constants c_1, c_2. Then $f(y)$ is $Algo(y)$ such that $\Theta(Algo(y))$ defines the *asymptotic tight limit* and a set of functions,

$$\Theta\big(Algo(y)\big) = \{0 \leq c_1 * Algo(y) \leq f(y) \leq c_2 * Algo(y) \qquad for\ all\ y \geq y_0\}. \qquad (59)$$

$\Theta(Algo(y))$ provides more information than $\Omega\big(Algo(y)\big)$ and $O\big(Algo(y)\big)$; however, the later two provide a specific idea about the amount of time (minimum for Ω and maximum for O) required to run the algorithm. The lower value order transitions between the nodes in the tree are relatively insignificant for large biochemical models and will generally ignored in the *Order of growth*. Therefore, while working with our algorithm, mostly $O\big(Algo(y)\big)$ will be considered because it gives the upper limit of the execution time (Woeginger, 2004).

3.7 Discussion and Conclusions

In this chapter, we have defined the principles of true events and their probabilities in the sample space. Further, the phenomenon of random variables as objects of a class called stochastic processes were defined for the sample space. This sample space is associated with finite state Markov chains based on Markov property to create a sample space for biochemical reaction networks. Large biochemical reaction networks usually have $R_{M(fs)}$ and $R_{M(sr)}$ types of *forward* and *reversible* reactions, which further categorises the states into transient and communicating states. The transient state usually represents the *forward* reaction if it is shifting the system to a unique state; whereas, the communicating state represents the *reversible* reaction of the system which shifts the system to a non-unique state.

As we were interested in unique states while expansion of the state-space we assumed the reducibility or simplification of the finite state Markov chain by visualising it as Markov chain tree to define the biochemical system procesess to exhibit the transition matrix as a directed tree of its associated graph. The transition between different nodes of the graph is represented by *forward* and *reversible* reactions. When concatenation is done in walks between the nodes, this represents the set of biochemical reactions of the system. However, when this concatenation is not considered it makes them directed acyclic graphs (DAG), called as Markov chain tree – a special type of graph that still represents the set of biochemical reactions even when there are no cycles between the nodes.

The initial state and end state of the biochemical system, together with the problem space, bring into existence the *problem instance* where all the searching should take place. This creates the *problem state-space graph* for large biochemical systems where functions or operators are applied to search for the new state of the system by going to new nodes in the graph. To deal with the *problem instance*, we have introduced the standards of *artificial intelligence* (AI) into the state-space expansion of biochemical systems for the solution of the CME. Recent trends in AI have motivated us to utilise the benefits of its searching standards architecture to treat *problem state-space graph* as *sample space*. To make it compatible for searching, we first defined the implicit successor, which takes the action on present state to reach future state. To retain the search track in the graph, we have defined the new data structure for the state-space of a biochemical system.

Consequently, to enforce the direction of expansion based on reactions that are firing at high rates at a particular time point, we have developed the post successor *Bayesian Likelihood*

Node Projection ($BLNP$) function (Eq. (56)) based on Bayes' theorem to support the implicit successor movement of the expansion. The $BLNP$ will be useful in identifying the most active reactions in the system based on propensity values while guiding the expansion towards the set of states with high probability mass. Lastly, we have discussed about the *time* and *space complexity* for algorithm analysis to understand important definitions.

Chapter 4
Intelligent State Projection

In this chapter we present various algorithms and methods that assist in the systematic expansion of the domain arising in biochemical network problems. The curse of dimensionality usually makes it too challenging to solve the CME directly and so it is important to choose the right strategy for updating the state-space for the best approximation that contributes to the most of the probability mass of the system. A large state-space results in a massive size of the transition rate matrix A and the time complexity of the evaluation of the matrix exponential increases. Therefore, we propose *Intelligent State Projection* (*ISP*) algorithm, which aims to improve the strategy of the state-space expansion and computing efficiency.

In this chapter, we first give an overview of the methods in section 4.1 in terms of the modules and sub-modules. In section 4.2, we derive and discuss the expansion conditions in detail and the complexity of the data structure. In sections 4.3 and 4.4, we give the details of the *ISP* variants - *Latitudinal Search* and *Longitudinal Latitudinal Search* of *Intelligent State Projection* method including expansion, updation, and its performance under different conditions. We also demonstrate the application of these variants on biological systems in the respective sections. In section 4.5, we will give a brief discussion and the chapter conclusions.

4.1 Introduction

A number of aspects need to be taken into consideration when developing a method to represent, explore and update the state-space of biochemical systems. The aspects that must be addressed are: what is the end state of the system at particular moment, what are the actions required to reach the end state, and what details need to be represented in the state-space description of the biochemical system that compliments its Markov process. We laid the foundation of these aspects previously in Chapter 3, and we will now define the infrastructure of our comprehensive method and derive the conditions for expansion.

To understand and predict the dynamics of state-space responses in biochemical systems, we developed an analytical method called *Intelligent State Projection* (*ISP*) that integrates the reactions propensities and parameters describing the Markovian processes through nodes that govern the states of the system. The *ISP* states expansion strategy is based on fundamentals

(as discussed in section 3.5), and the relative successor operator or function that perform operations on its inputs. In ISP, most of the biochemical network state-space problems can be formulated as searches and solved by reducing them to one of searching a graph or a tree. To use this approach, we will consider the successor operator S_{uc} together with $BLNP$, which defines the sequence of actions, leading from the initial state to the end state at different time interval, that lead to the solution. The number of successive steps required for the solution of a problem are not known a priori in ISP, but they are often determined through an exploration of the alternatives using a systematic trial and error strategy.

The ISP method can be classified into two subroutines $-$ *Latitudinal Search* (LAS) and *Longitudinal Latitudinal Search* $(LOLAS)$ consisting of several modules that incorporates a set of inputs and functions within eight compartments. Figure 4.1 depicts the modules of the ISP, and its integrated form is discussed later in sections 4.3 and 4.4. The ISP has six major modules (*steps*): (i) state module; (ii) explore module; (iii) sort module; (iv) dictionary module; (v) update module; and (vi) queue/stack module, (see Table 4.1). First, the input condition components are identified in the model, which are then input into different functions within the modules. In addition to the modules that are marked in Figure 4.1, the steps of both the ISP variants are presented separately in Table 4.2 and Table 4.7.

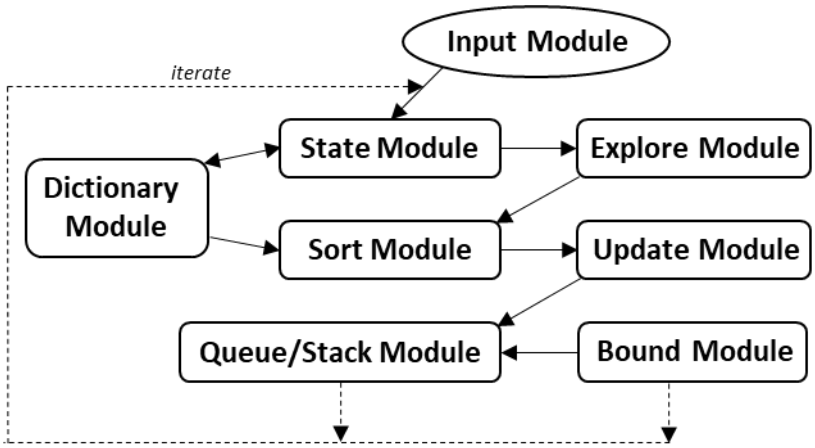


Figure 4.1. Comprehensive *ISP* method flow chart. A description of the modules (steps), sub-modules and the list of derived components considered in *ISP* are provided briefly in Table 4.1.

Table 4.1. Modules/sub-modules, components of the comprehensive *ISP* method.

Module (Steps)	Sub-module (Steps)	Components
Input module	–	N_0, a_μ, V_μ, τ_m, t_f, t_{step}
	Provide inputs to the sub-modules at t_d	
State module	$N_i = (X_0, \mathfrak{d}_l)$	N_i, X_0, $\mathfrak{d}_l$, t_d
	Describes the graph initial node and corresponding set of states at t_d	
Explore module	$1 - I^T exp(t.A_j).P^{(t)}(X_0) \geq \tau_m(leak)$ or $1 - I^T exp(t_f A_j).P^{(t)}(X_0) \geq \tau_m * \frac{(\text{no.of } R_{M(sr)})}{(\text{no.of } R_{M(fs)})}$ or $1 - I^T exp(t_f A_j).P^{(t)}(X_0) \geq \tau_m * \frac{(\text{no.of } R_{M(fs)})}{(\text{no.of } R_{M(sr)})}$	A_j, exp, X_i, X_0, τ_m, t, I^T
	Defines the validation functions for exploration at t	
Sort module	➢ $exp(t.A_j).P^{(t)}(X_0)$ ➢ $P^{(t)}(\mathbf{X}_K) \geq \tau_m(leak) > P^{(t)}(\mathbf{X}'_K)$ ➢ $\mathbf{X}_K \leftarrow \mathbf{X}_K - \mathbf{X}'_K$	A_j, exp, X_i, X_0, $\tau_m(leak)$, t, $\mathbf{X}_K$, $\mathbf{X}'_K$
	Follows the sorting and bunking of $P^{(t)}(\mathbf{X}'_K)$	
Dictionary module	➢ $Dict$ ➢ $\mathbf{n}_K = \left(\mathbf{X}_K, \mathfrak{d}_l, \mathfrak{C}_{N_i,N'_i}(min)\right)$	$Dict$, $\mathbf{X}_K$, $\mathfrak{d}_l$, $\mathbf{n}_K$, R_M, $\mathfrak{C}_{N_i,N'_i}(min)$
	Accounts for mapping of nodes by keeping the track of transitions at t.	
Update module	➢ $\mathbf{n}_K = \left(\mathbf{X}_K, \mathfrak{d}_l, \mathfrak{C}_{N_i,N'_i}(min)\right) \in domain$, ➢ $domain \leftarrow domain_{previous} \cup domain$, ➢ $\mathbf{n}_K = \left(\mathbf{X}_K, \mathfrak{d}_l, \mathfrak{C}_{N_i,N'_i}(min)\right) \notin domain$, ➢ $b\left(N_{N1,..NM} \middle| b'_{1,N....N',N}\right)$ ➢ $queue$ or $stack$.	$\mathbf{X}_K$, $\mathfrak{d}_l$, $\mathbf{n}_K$, R_M, $\mathfrak{C}_{N_i,N'_i}(min)$, $P_{N,N'}(\omega)$, $b'_{N',N}$, $queue$ or $stack$, $domain_{previous}$.
	Responsible for updating the states in the domain; and pre-conditioning at t.	
Queue or stack module	➢ $\mathbf{n}_K = \left(\mathbf{X}_K, \mathfrak{d}_l, \mathfrak{C}_{N_i,N'_i}(min)\right)$, ➢ $queue$ or $stack$	$\mathbf{X}_K$, $\mathfrak{d}_l$, $\mathbf{n}_K$, R_M, $\mathfrak{C}_{N_i,N'_i}(min)$, $queue$ or $stack$, $domain_{previous}$.

	Consider new $\mathbf{n}_K$ unique set of nodes for addition in the domain.	
Bound module	➢ $count(Б_{limit}) = Б_{limit}$, ➢ $Bound_{upper} = \{domain\}$, ➢ $Bound_{lower} \leftarrow Bound_{upper}$, ➢ $count(Б_{limit}) < Б_{limit}$.	$count(Б_{limit})$, $Б_{limit}$, $Bound_{lower}$, $Bound_{upper}$.
	Defines $Bound_{lower}$ for next iteration and conditions if $count(Б_{limit})$ reaches the $Б_{limit}$.	

(see Notations and section 4.2, 4.3 and 4.4 for details)

The *state module* describes the initial node of the graph and maps with the corresponding set of states at any time, t_d. It takes the initial values of the components from the input module and stores the information of a number of nodes, sets of states, depth of nodes in the state-space or the bound limit in every iteration.

The *explore module* defines the validation functions for exploration. At t it flags the current node to be explored, and looks for a set of child nodes carrying a set of states, as long as the validation Eq. (82) (or Eq. (80) or (81) based on section 4.2.2 conditions) holds true; otherwise it stops the exploration. The cease of validation defines the stopping criteria of the exploration (see section 4.2).

The *sort module* organises the probabilities based on the conditions of the sub-module (as shown in Table 4.1) and bunks the probabilities that provide minimal weightage in the approximation as it is computationally expensive to re-compute these probabilities up to t_f.

The *dictionary module* keeps track of adjacent sets of $\mathbf{n}_K \in \mathbf{n}_J$ nodes in real-time and walks between them. The dictionary $Dict$ is a compressed row format that records different nodes in a single transition, but uses a similar $\mathbf{n}_K$ set of nodes in all possible transitions.

The *update module* is responsible for pre-conditioning and updating the domain with the new set of new states and validating it for duplicate states as a result of the previous iterations. The corresponding set of nodes carrying a set of states are not considered if the states are already present in the domain; however the changes in propensity values are updated.

The *queue* and *stack* module provides a unique node to the successor operator Eq. (46) to consider new states by pulling the node out from the *queue* in LAS or by popping the top node from the *stack* in $LOLAS$. In LAS, these new states act as an input to the *state module* for the next iteration, as well as for approximation methods (Dinh et al., 2016; Fallis et al., 2012; Huy D.Vo et al., 2017; Wolf et al., 2010) while, in $LOLAS$, it goes through a *bound module*, which defines the upper and lower bounds of the iteration and the bound limit that satisfies the sub-module conditions.

These modules and sub-modules constitute the ISP method, such that they track key changes in the components that follows the changes in the reaction propensities by population and activation of the species and describes the dynamics of the biochemical system. This method also permits the time form quantification of the state-space based on the size and dimensions of the model. Further details of the method conditions are discussed in the following sections.

4.2 Derivation of the Method Conditions

In this section we derive the conditions and theory in substantial detail to prove the efficiency of our *ISP* method and test them by computing real biochemical models separately in the following chapters. We also compare the computation at different times points to study the state-space expansion. The *ISP* infrastructure gives the facility to employ the data structure (as discussed in Chapter 3) to visualise the state-space and the expansion of states built upon an intelligent search and tracking (as discussed in 3.5). To design the time stepping expansion criteria, we first consider the probability exponential form of the CME, given by Eq. (9), to construct the domain closest to the optimum (small as possible) sized domain by being able to cheaply find the minimum number of states with a maximum probability mass.

4.2.1 Expansion Criterion for States Space!

The states are indexed by $\{1,2 \ldots \ldots . K\}$ in the domain denoted by set $\mathbf{X}_J$. To derive the time based on the state-space expansion conditions, the probability exponential form of the CME Eq. (9) is evaluated for approximation up to the desired final time t_f in steps. To focus on the probable states that contribute most to the probability mass in the domain, we first define the set of non-probable states (having the least probability mass) as $\mathbf{X}'_K$, which are to be bunked. A number of the states will usually be infinite without selecting probable states for the domain. By doing this we can avoid unnecessary recalculation of probabilities and decrease the computational efforts by keeping the domain small. This bunking can also be applied to the initial distribution of the system at t_0. If submatrix A'_j contains the non-probable set $\mathbf{X}'_K$ of states, then probability of set will be,

$$P^{(t)}(\mathbf{X}'_K) = exp\big(t.A'_j\big).P^{(t)}(X_0). \tag{60}$$

The criterion for defining the non-probable states is determined by the τ_m tolerance value. A'_j will only be considered to have non-probable states if,

$$A'_j = \begin{cases} \text{nonprobable states}, & \text{if } P^{(t)}(\mathbf{X}'_K) < \tau_m \\ \text{else}, & \\ \text{probable states}, & \text{if } P^{(t)}(\mathbf{X}'_K) \geq \tau_m \end{cases} \tag{61}$$

Similarly, submatrix A_j has a probable set $\mathbf{X}_K$ of states if $P^{(t)}(\mathbf{X}_K) \geq \tau_m$ otherwise, the states from $\mathbf{X}_K$ are bunked to $\mathbf{X}'_K$ if $P^{(t)}(\mathbf{X}_K) < \tau_m$. For any iteration, if $P^{(t)}(\mathbf{X}'_K) \geq \tau_m$ then (from Eq. (61)) some states from $\mathbf{X}'_K$ return to $\mathbf{X}_K$ in the next iteration to increase the accuracy of the approximate solution (Æ). The column sum of the approximate solution (Æ) of these states is

defined as:

$$Æ = I^T exp\left(t_f A_j\right).P^{(t)}(X_0), \tag{62}$$

where, $I = (1, \ldots 1)^T$ is of an appropriate length. Declaring some states as non-probable and removing them before calculation of the probabilities as seen in (Sunkara, 2013) will decrease the accuracy of Æ with the cumulative step errors. This can be validated from the state probabilities that have been ignored in the domain:

$$Æ = 1 - P^{(t)}(\mathbf{X}'_K). \tag{63}$$

We define the step error in terms of the probabilities bunked. If $e_{rror} \propto P^{(t)}(\mathbf{X}'_K)$ then,

$$e_{rror} = 1 - I^T exp\left(t_f A_j\right).P^{(t)}(X_0) \tag{64}$$

$$e_{rror} = 1 - Æ \tag{65}$$

Every expansion step explores at least one new state and change $\{\mathbf{X}_K\}$ but not necessarily $\{\mathbf{X}'_K\}$ as long as:

$$P^{(t)}(\mathbf{X}_K) \geq \tau_m > P^{(t)}(\mathbf{X}'_K), \tag{66}$$

is satisfied. For ideal systems with a given initial probability of $P^{(t_0)}(X_0)$, the $\{\mathbf{X}'_K\}$ should be *null* and so $P^{(t_f)}(\mathbf{X}'_K) = 0$. For such systems $\{\mathbf{X}_K\}, \{\mathbf{X}'_K\}\ \epsilon\ \{\mathbf{X}_J\}$ for final projection and,

$$P^{(t_f)}\left(\mathbf{X}_J\right) = P^{(t_f)}(\mathbf{X}_K) + P^{(t_f)}(\mathbf{X}'_K), \tag{67}$$

$$P^{(t_f)}\left(\mathbf{X}_J\right) = P^{(t_f)}(\mathbf{X}_K) + 0. \tag{68}$$

$P^{(t_f)}\left(\mathbf{X}_J\right)$ in Eq. (68) is the solution of Eq. (7) after the state-space is expanded to $\mathbf{X}_K$. However, for large biochemical systems, Eq. (68) may not hold completely true due to the nature (*fast* ($R_{M(fs)}$) and *slow* ($R_{M(sr)}$)) of some reactions present in the system; therefore, the condition in Eq. (61) will pass the states from $\mathbf{X}'_K$ to $\mathbf{X}_K$ and then the states with lowest probabilities will be bunked when:

$$P^{(t)}(\mathbf{X}'_K) << P^{(t_f)}(\mathbf{X}_K), \tag{69}$$

which improve the solution. Removing without calculating the probabilities of some states is one of the lag, current methods (Dinh et al., 2016, Dinh et al., 2017; Munsky et al., 2006,

Munsky et al., 2007; Sunkara, 2013; Sunkara et al., 2010; Wolf et al., 2010) are facing over the cost of achieving the truncated domain and saving computation time. To address this, we set a $P^{(t)}(\mathbf{X}'_K)$ leakage point based on:

$$P^{(t)}(\mathbf{X}_K) \geq \tau_m(leak) > P^{(t)}(\mathbf{X}'_K), \tag{70}$$

where, $\tau_m(leak)$ for systems will reform the Eq. (66) as:

$$P^{(t)}(\mathbf{X}_K) \geq \tau_m * 0.4 > P^{(t)}(\mathbf{X}'_K), \tag{71}$$

which would then zip the $\mathbf{X}'_K$ further by leaking the highest probabilities to $\mathbf{X}_K$ so the probability sum is no longer conserved. The motivation of setting this ration is to reconsider (up to 40% of $\mathbf{X}'_K$) the bunked states to improve the Æ solution and decrease the expansion step error. While modelling the biochemical system, if *slow* and *fast* reaction criterion is considered during expansion, then $\tau_m(leak)$ will be defined as,

$$= \begin{cases} \tau_m * \dfrac{(\text{no. of } R_{M(sr)})}{(\text{no. of } R_{M(fs)})}, & \text{if no. of } R_{M(sr)} < \text{no. of } R_{M(fs)} \\ \text{else,} & \\ \tau_m * \dfrac{(\text{no. of } R_{M(fs)})}{(\text{no. of } R_{M(sr)})}, & \text{if no. of } R_{M(sr)} > \text{no. of } R_{M(fs)} \\ \text{else,} & \\ \tau_m * 0.4, & \text{if no. of } R_{M(fs)} = \text{no. of } R_{M(sr)}. \end{cases} \tag{72}$$

We consider Eq. (71) criterion throughout the experiments in this study. The conditions in Eqs. (71) and (72) will then lead to an optimal set of states as,

$$\mathbf{X}_K \longleftarrow \mathbf{X}_K - \mathbf{X}'_K, \tag{73}$$

at t_d in the domain. When $\mathbf{X}_K$ is updated at every t_{step} before reaching t_f, it creates several intermediate domains which we define as $Bound$. At t_0, the domain only has the initial state of the system; therefore, we define the $Bound$ as:

$$Bound_{lower} = \{domain, \mathrm{d}_{l=1}\}. \tag{74}$$

After a single t_{step} of expansion, if $\mathbf{X}_K$ is updated with a new state or set of states, it creates:

$$Bound_{upper} = \{domain, \mathrm{d}_l\} \tag{75}$$

at t_d. Here, d_l denotes the depth level of the latest state or set of states that has been added in

the domain to form $Bound_{upper}$. Further, this $Bound_{upper}$ is re-labelled and considered as $Bound_{lower}$ for the next t_{step} of the expansion. If the expansion is to be limited in the number of $Bounds$, then every $count(б_{limit})$ leads to:

$$count(б_{limit}) = б_{limit}, \tag{76}$$

where, $б_{limit}$ is the bound limit. For example, if $б_{limit} = 2$, then $count(б_{limit})$ will be from $0 \overset{\text{to}}{\rightarrow} 1 \overset{\text{to}}{\rightarrow} 2$. If the $count(б_{limit})$ is increased up to $б_{limit}$ for ${I_{tr}}^{\text{th}}$ iterations, then $Bound_{upper}$ in the current iteration will be $Bound_{lower}$ for the next iteration. Every $Bound_{lower}$ state will be the strict subset of every consecutive $Bound_{upper}$ given as:

$$Bound_{lower}(Z) \subset Bound_{upper}(Z). \tag{77}$$

and the upper bound as:

$$Bound_{upper}(Z) = \{domain\ at\ Z^{th}\ iteration, д_{l}\}, \tag{78}$$

where Z is the number of $Bounds$ (or intermediate domains). The *2D pyramid* domain in Figure 4.2 showcases the increase in the population of states in the domain with the increase in iterations (I_{tr}). The apex of the pyramid represents the initial state X_0 of the system at $Bound_{lower}(1)$ at t_0, whereas the base represents the deepest level where the system ends with the final domain carrying set $\mathbf{X}_K$ with maximum probability mass.

For large biochemical systems, the number of $Bounds$ created are based on I_{tr} and have million/billions of states. The expansion can be terminated by defining time t_f at which the solution is required. In order to have auto break-off point in the expansion, it is important to define the criteria that limits I_{tr} when no more new states can be searched. Therefore, in the section 4.2.2, we define this criteria which also fits biochemical systems having *fast* and *slow* reactions.

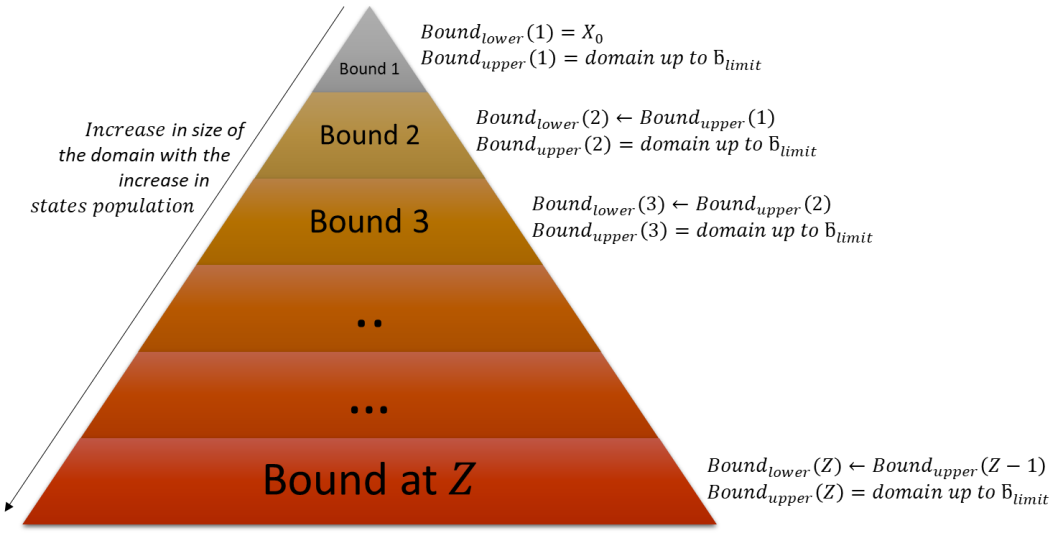


Figure 4.2. General framework of *2D pyramid* domain showing the increase in the size of the domain with the increase in state with an increase in the bounds. $Bound_{lower}(1)$ represents the initial condition, whereas $Bound_{upper}(Z)$ represents the final domain carrying explored set of states of the system.

4.2.2 Cease of Criterion after Updating

In every expansion step, the domain is validated by Eq. (71) and new states are added in $\mathbf{X}_K$ as long as:

$$1 - I^T exp\left(t_f A_j\right).P^{(t)}(X_0) \geq \tau_m, \tag{79}$$

is satisfied for probable states and stops if it is not. This led to a point at t_f where $e_{rror} < \tau_m$, but the expansion can be extended further to meet the accuracy by re-considering the criteria as:

$$1 - I^T exp\left(t_f A_j\right).P^{(t)}(X_0) \geq \tau_m * \frac{(\text{no. of } R_{M(sr)})}{(\text{no. of } R_{M(fs)})}, \tag{80}$$

$$1 - I^T exp\left(t_f A_j\right).P^{(t)}(X_0) \geq \tau_m * \frac{(\text{no. of } R_{M(fs)})}{(\text{no. of } R_{M(sr)})}, \tag{81}$$

$$1 - I^T exp\left(t_f A_j\right).P^{(t)}(X_0) \geq \tau_m(leak)\,, \tag{82}$$

before steps to t_f. However, the size of $\mathbf{X}_K$ obtained through Eqs. (80), (81) and (82) at t_f will be greater compared to the size of $\mathbf{X}_K$ obtained by Eq. (79) at t_f, as the latter will have fewer number of states. In Eqs. (80), (81) and (82), with the increase in size of A_j, the value of the left-hand side also increases resulting in an improvement in *Æ*. When considering any Markov process of a biochemical system of any size in which the probability density expands according to Eq. (7) then Eqs. (80), (81) and (82) will approximate the solution within $\tau_m * \frac{(\text{no. of } R_{M(sr)})}{(\text{no. of } R_{M(fs)})}$, $\tau_m * \frac{(\text{no. of } R_{M(fs)})}{(\text{no. of } R_{M(sr)})}$ and $\tau_m(leak)$, respectively, of the true solution of the CME, which is Eq. (7).

In section 4.3, we use the principles and conditions discussed in Chapter 3, sections 4.2.1 and 4.2.2, and introduce the first subroutine of *ISP* called the *Latitudinal Search*.

4.3 Latitudinal Search Strategy

In this section, we delve deeper into the first subroutine of ISP called the *Intelligent State Projection Latitudinal Search* ($ISP\ LAS$). Figure 4.3 manifest the infrastructure of the LAS strategy, showing G_{mc}, the $queue$ and the domain. The $queue$ data structure of LAS is based on $FIFO$ (First In, First Out) method. It assume that oldest state added to the $queue$ is considered first. In particular, we will define and exploit the direction of expansion step-by-step based on intuitive knowledge (as discussed in section 3.5) gained from probability of future reaction events and follow the conditions (as discussed in Section 4.2). Furthermore, we show step-by-step how the nodes are explored and states updated in the domain in I_{tr} iterations. To give a broader perspective, we compare this class of subroutine in Chapter 5 with the existing techniques discussed in Chapter 2. Finally, we demonstrate the $ISP\ LAS$ application on a large biochemical system in Case study 2 - Chapter 7. We also provide some practical guidelines for setting up $ISP\ LAS$ for the experiment.

The states at level $\mathfrak{d}_l$ are expanded only after all the states at level $\mathfrak{d}_l - 1$ have been expanded, i.e. the search is undertaken *level-by-level* and depth $\mathfrak{d}_l$ increases in every I_{tr} iteration. In the case of networks with *reversible* reactions, the ISP condition will prevent LAS from returning to the state it came from and also prevent transitions containing cycles resulting in DAG with no repetition of any state whatsoever; however, the changes in propensities $a_{i,j}$ are validated. Verifying the explored states in $\mathbf{X}_K$ in iterations ensures that the algorithm completes and a deadlock in the state transition cycles cannot occur. The *time complexity* of LAS depends on the average transitioning factor $\mathfrak{T}$ and depth $\mathfrak{d}_l$ and is given by (see Appendix E for detailed discussion),

$$1 + \mathfrak{T}^1 + \mathfrak{T}^2 + \cdots + \mathfrak{T}^{\mathfrak{d}_l} + (\mathfrak{T}^{\mathfrak{d}_l+1} - \mathfrak{T}) = O(\mathfrak{T}^{\mathfrak{d}_l+1}), \tag{83}$$

where,

$$\mathfrak{T} = \frac{\text{Total no. of walk between different nodes}}{\text{Total no. of nodes explored}} \tag{84}$$

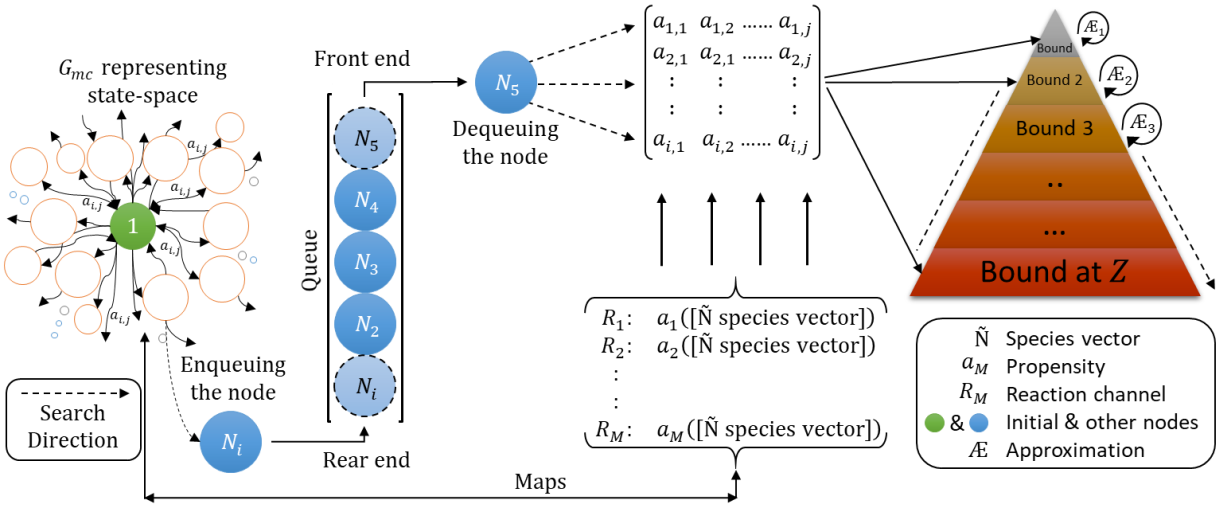


Figure 4.3. Infrastructure of the *Latitudinal Search* strategy, showing G_{mc}, the *queue* and the domain.

For the nodes at the deepest level $\mathfrak{d}_l$, all walks are valid except for the very last node which stores the end state of the system. Therefore, once the end state is found, based on Eq. (70), LAS will zip $\mathbf{X}'_K$ further leaking the highest probabilities to $\mathbf{X}_K$ for the solution of Eq. (7) which includes the end state of the system. As no state is ever repeated in the domain, *space complexity* will decrease when the set of states $\mathbf{X}'_K$ is bunked at t'seconds in iterations if Eqs. (69) and (70) are satisfied. In Eq. (63), $P^{(t)}(\mathbf{X}'_K)$ is computed according to Eq. (9) (exponential form of the CME), where τ_m is the tolerance and I is the identity matrix. Due to this stepping bunking of $\mathbf{X}'_K$ from $\mathbf{X}_K$, the time complexity $O(\mathfrak{T}^{d+1})$ reduces to $O(min(\mathfrak{T}^{\mathfrak{d}_l+1}, \mathfrak{T}|\mathbf{X}_J|))$, where $|\mathbf{X}_J|$ is the size of the state-space (Burrage et al., 2006). In contrast, the expansion of new nodes carrying similar states tends to increase $O(min(\mathfrak{T}^{\mathfrak{d}_l+1}, \mathfrak{T}|\mathbf{X}_J|))$; however, repetitive states are ignored.

If the input τ_m is too small, the default value of $sqrt(eps)$ is automatically used by the algorithm. Where, $sqrt$ is the square root and eps is the default value of the epsilon on machine. The expansion of child nodes containing $state(N_i) = X_i$ stops if the condition of Eq. (82) is not satisfied. If criterion of *slow and fast* reaction is considered, then condition of Eq. (81) or (82) is used depending on number of $R_{M(sr)}$ and $R_{M(fs)}$. The Table 4.2 shows the steps of the LAS method with embedded $BLNP$ function from step 4a to 5b.

Table 4.2. Steps of *ISP* Latitudinal Search (*LAS*) Algorithm.

Step 0:	**Inputs:** Initial node N_0, a_μ, v_μ, tol τ_m, t_f, t_{step} **Initialise:** $Bound_{lower} = \mathbf{X}_K$, $b'_{N',N}\left(X - v_\mu\right)' = P^{(t)}(X_0)$, $A = [\,]$
Step 1:	Start from parent node $N_i = (X_0, \mathfrak{a}_l) \longleftarrow$ Current State of the system at t_d,
Step 2:	Flag the current node as explored, update A and add the state X_i in the domain so that; *if* $1 - I^T exp\left(t.A_j\right).P^{(t)}(X_0) \geq \tau_m(leak)$ holds true go to *Step 3*; *else* stop the algorithm
Step 3:	Sort $exp\left(t.A_j\right).P^{(t)}(X_0)$ and shift the set of states in $\mathbf{X}'_K$ at t' having smallest probabilities, if $P^{(t)}(\mathbf{X}_K) \geq \tau_m(leak) > P^{(t)}(\mathbf{X}'_K)$ and at t_d update $\mathbf{X}_K \longleftarrow \mathbf{X}_K - \mathbf{X}'_K$
Step 4a:	Extend the graph dictionary $Dict$ by $v_\mu(X_i(t))$ by 1 level to check all the nodes $\mathbf{n}_J = \left(\mathbf{X}_J, \mathfrak{a}_l, \mathfrak{C}_{N_i,N'_i}(min)\right)$ adjacent to N_i: $Bound_{upper} \leftarrow R_M(Bound_{lower})$ reachable by exactly R_M reactions (from fast to slow) having $\mathfrak{C}_{N_i,N'_i}(min)$. If $\mathbf{n}_K = \left(\mathbf{X}_K, \mathfrak{a}_l, \mathfrak{C}_{N_i,N'_i}(min)\right)$ be the set of adjacent nodes such that $\mathbf{n}_K \in \mathbf{n}_J$ then go to next *Step*,
Step 4b:	Compute the $BLNP$ function for $\mathbf{n}_K \in Bound_{upper}$: $b\left(N_{N1,..NM} \middle| b'_{1,N....N',N}\right) = P_{N,1....N,N'}(\omega) * b'_{N',N}(X - v_i)'$
Step 5a:	If $\mathbf{n}_K = \left(\mathbf{X}_K, \mathfrak{a}_l, \mathfrak{C}_{N_i,N'_i}(min)\right) \in domain$, then *update* the values of the set of states $\mathbf{X}_K$ present in $domain$ and take $domain \longleftarrow domain_{previous} \cup domain$ and go back to *Step 1*; *else* If $\mathbf{n}_K = \left(\mathbf{X}_K, \mathfrak{a}_l, \mathfrak{C}_{N_i,N'_i}(min)\right) \notin domain$, then add it to the $queue$ in order, according to reachability and go to the next *Step*,
Step 5b:	*sort* $b\left(N_{N1,..NM} \middle| b'_{N1,..NM}\right)$ in descending order and update $queue \longleftarrow (queue\ ;\ b\left(N_{N1,..NM} \middle| b'_{1,N....N',N}\right))$
Step 6:	Pull out the nodes $\mathbf{n}_K = \left(\mathbf{X}_K, \mathfrak{a}_l, \mathfrak{C}_{N_i,N'_i}(min)\right)$ from the $queue$ in order and add the set of states $\mathbf{X}_K$ in the domain as $domain \longleftarrow domain + \mathbf{X}_K$ and take $domain_{previous} \cup domain$, then go back to *Step 1*,
	Output: $domain$ with probable states

LAS will be optimal if the transitions between all the states are uniform, i.e., all the R_M reactions have the same propensity values; however, in real biochemical models this condition is unusual. To demonstrate the $ISP\ LAS$ algorithm, let us assume a system with three species, C, A, T, with initial molecular counts of 5, 0, 0, respectively, undergoing reactions as

$$\boldsymbol{R_1}: C \xrightarrow{k_1} A, \qquad \boldsymbol{R_2}: A \xrightarrow{k_2} T, \qquad \boldsymbol{R_3}: T \xrightarrow{k_3} C \tag{85}$$

with $k_1 = k_2 = k_3 = 1$ and depicted as a network in Figure 4.4,

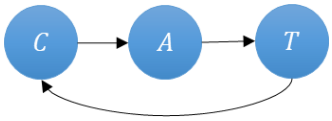


Figure 4.4. Toy model network. Showing three $\tilde{N} = 3$ species, $\boldsymbol{C}, \boldsymbol{A}, \boldsymbol{T}$ in a network defining reactions, as given in Eq. (85).

These species counts are used as a state-space to define the toy model and these copy counts are tracked as $([C], [A], [T]) \in \tilde{N} := (x_0, x_1, x_2)$. In reaction R_1, the copy counts of C are reduced by 1, which increases the copy count of A by 1. Whereas, reaction R_2 decreases the count of A by 1 and increases the counts of T by 1. In R_3, the counts of T are decreased by 1, which increases the counts of C by 1. We can now define the transitions associated with R_1, R_2, R_3 in stoichiometric vector V_M matrix as:

$$V_M = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \\ 1 & 0 & 1 \end{bmatrix}. \tag{86}$$

We express the propensity functions of the three reactions in terms of the states $([C], [A], [T]) \in \tilde{N}$ as:

$$\boldsymbol{R_1}: \quad a_1([S], [B], [C], [P], [E]) = k_1([S]),$$

$$\boldsymbol{R_2}: \quad a_2([S], [B], [C], [P], [E]) = k_2([B]),$$

$$\boldsymbol{R_3}: \quad a_3([S], [B], [C], [P], [E]) = k_3([B][P]).$$

Let, G_{mc} be the discrete-state continuous time Markov chain graph and $\mathbb{X}$ be the equivalent tree of G_{mc} as DAG representing state-space of the system. In growing Markov chain tree, the transition between the nodes:

$$N_i \xrightarrow{v_\mu(X_0(t),X_1(t),\ldots,X_K(t))} N_{i+1}\,, \tag{87}$$

is defined in the typical form of a dictionary $Dict$ (for example Eq. (52)). In the following section 4.3.1, we now demonstrate the LAS expansion and update strategy on the toy model.

4.3.1 Expansion and Update

Nodes $\mathbf{n}_J = \left(\mathbf{X}_K, \mathfrak{a}_l, \mathfrak{c}_{N_i,N_i'}(min)\right)$ carrying states are expanded, as shown by G_{mc} in number of stages ($\hat{\mathrm{S}}$). From Figs (A) to (F) in Figure 4.5 represent the systematic exploration of the nodes carrying states based on LAS and as depicted up to six stages ($\hat{\mathrm{S}} = 6$), and the walk represents R_M reactions with propensities $a_{i,j}$. This shows the change in state from $state(N_i) = X_i$ and $state(N_i') = X_{i'}$ of the system at any time t.

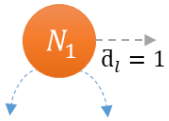


Fig (A). Stage 1, $\mathfrak{d}_l = 1$

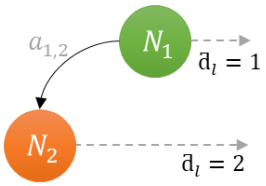


Fig (B). Stage 2, $\mathfrak{d}_l = 2$

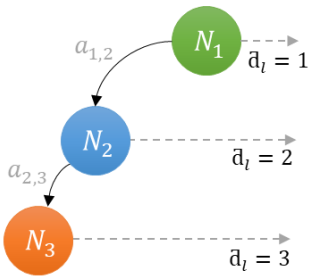


Fig (C). Stage 3, $\mathfrak{d}_l = 3$

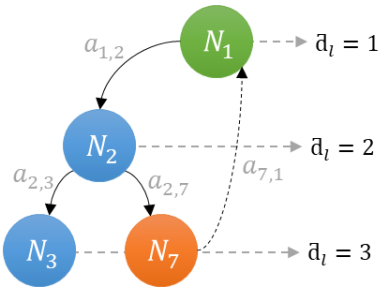


Fig (D). Stage 4, $\mathfrak{d}_l = 3$

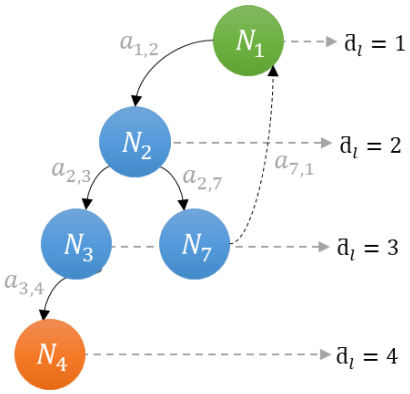


Fig (E). Stage 5, $\mathfrak{d}_l = 4$

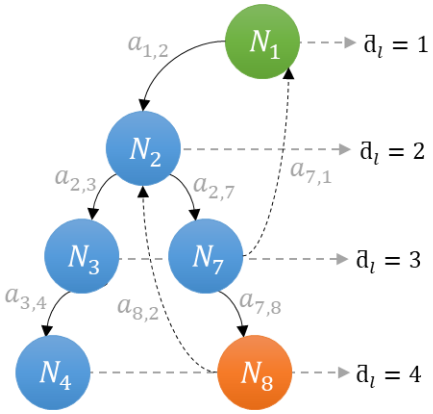


Fig (F). Stage 6, $\mathfrak{d}_l = 4$

Figure 4.5. Six stages of state exploration based on a *LAS* search for the toy model, given an average transitioning factor of $\mathrm{T} = 2$. Fig (A) is the first stage that denote the initial node carrying initial state of the system, Fig (B) is the second stage that denote all the nodes with new node at $\mathfrak{d}_l = 2$, Fig (C) is the third stage that denote all the nodes with new node at $\mathfrak{d}_l = 3$, Fig (D) is the fourth stage that denote all the nodes with new node at $\mathfrak{d}_l = 3$, Fig (E) is the fifth stage that denote all the nodes with new node at $\mathfrak{d}_l = 4$, Fig (F) is the sixth stage that denote all the nodes with new node at $\mathfrak{d}_l = 4$.

According to LAS strategy in Table 4.2, *step 0* defines the inputs based on the model and some are initialised with the given values. The model is solved for $t_f = 1.0\ sec$ by taking $t_{step} = 0.01sec$, $\tau_m = 1e-6$ with the initial state $state(N_1) = X_0$. In *step 1*, ISP starts with the given initial node and this carries the initial state of the system. The initial state is also called the parent state of the system. In *step 2*, the current node is flagged as explored, and the relevant state is added to the domain if Eq. (82) is satisfied otherwise stop the algorithm if it does not.

Step 3 sorts the probabilities of $exp(t_f A).P^{(t)}(X_j)$ and bunks them from the domain at t' and stores them in the bunker $\mathbf{X}'_K$ such that $P^{(t)}(\mathbf{X}'_K) < \tau_m(leak)$, resulting in a domain with unique probable states at t_d. *Step 4a* and *4b*, extend the graph dictionary $Dict$ by a successor function to check all nodes $\mathbf{n}_J = \left(\mathbf{X}_J, \mathfrak{a}_l, \mathfrak{C}_{N_i,N'_i}(min)\right)$ that are adjacent to the current node N_i and are reachable by exactly R_M reactions such that $Bound_{upper} \leftarrow R_M(Bound_{lower})$ having $\mathfrak{C}_{N_i,N'_i}(min)$. Let $\mathbf{n}_K = \left(\mathbf{X}_K, \mathfrak{a}_l, \mathfrak{C}_{N_i,N'_i}(min)\right)$ be the set of adjacent nodes such that $\mathbf{n}_K \subset \mathbf{n}_J$. Once the adjacent nodes are found the $BLNP$ function calculates $b\left(N_{N1,..NM} \middle| b'_{1,N....N',N}\right)$ for $\mathbf{n}_K \in Bound_{upper}$.

Steps 5a and *5b* validate if the set of states of set $\mathbf{n}_K$ is a part of the domain. Following that, it updates the values of the set of states present in the domain and applies a check to reassure the uniqueness of the states in the domain and resume the algorithm from *step 1*. Otherwise, if the set of states of set $\mathbf{n}_K$ is not part of the domain then nodes are added in the $queue$. Further, $queue$ is then updated according to sorted $b\left(N_{N1,..NM} \middle| b'_{N1,..NM}\right)$. *Step 6*, will pull the nodes from the $queue$ in order and add the set of states $\mathbf{X}_K$ in the domain followed by a check to reassure the uniquness of the states in the domain. Nodes $\mathbf{n}_J = \left(\mathbf{X}_K, \mathfrak{a}_l, \mathfrak{C}_{N_i,N'_i}(min)\right)$ are then expanded, as given in Table 4.3.

Table 4.3. *LAS* nodes expansion strategy for the toy model.

Iterations	Queue	Domain
1	$\{N_1\}$	[Empty]
2	$\{N_2, N_7\}$	$[N_1]$
3	$\{N_3, N_7\}$	$[N_2, N_1]$
4*	$\{N_7, N_4, N_8\}$	$[N_3, N_2, N_1]$
5	$\{N_4, N_8\}$	$[N_7, N_3, N_2, N_1]$
6*	$\{N_8, N_5, N_{10}\}$	$[N_4, N_7, N_3, N_2, N_1]$
7*	$\{N_5, N_{10}, N_9\}$	$[N_8, N_4, N_7, N_3, N_2, N_1]$
8*	$\{N_{10}, N_9, N_6, N_{13}\}$	$[N_5, N_8, N_4, N_7, N_3, N_2, N_1]$
9*	$\{N_9, N_6, N_{13}, N_{11}\}$	$[N_{10}, N_5, N_8, N_4, N_7, N_3, N_2, N_1]$
10	$\{N_6, N_{13}, N_{11}\}$	$[N_9, N_{10}, N_5, N_8, N_4, N_7, N_3, N_2, N_1]$
11*	$\{N_{13}, N_{11}, N_{17}\}$	$[N_6, N_9, N_{10}, N_5, N_8, N_4, N_7, N_3, N_2, N_1]$
12*	$\{N_{11}, N_{17}, N_{14}\}$	$[N_{13}, N_6, N_9, N_{10}, N_5, N_8, N_4, N_7, N_3, N_2, N_1]$
13*	$\{N_{17}, N_{14}, N_{12}\}$	$[N_{11}, N_{13}, N_6, N_9, N_{10}, N_5, N_8, N_4, N_7, N_3, N_2, N_1]$
14*	$\{N_{14}, N_{12}, N_{18}\}$	$[N_{17}, N_{11}, N_{13}, N_6, N_9, N_{10}, N_5, N_8, N_4, N_7, N_3, N_2, N_1]$
15*	$\{N_{12}, N_{18}, N_{15}\}$	$[N_{14}, N_{17}, N_{11}, N_{13}, N_6, N_9, N_{10}, N_5, N_8, N_4, N_7, N_3, N_2, N_1]$
16*	$\{N_{18}, N_{15}\}$	$[N_{12}, N_{14}, N_{17}, N_{11}, N_{13}, N_6, N_9, N_{10}, N_5, N_8, N_4, N_7, N_3, N_2, N_1]$
17*	$\{N_{15}, N_{19}\}$	$[N_{18}, N_{12}, N_{14}, N_{17}, N_{11}, N_{13}, N_6, N_9, N_{10}, N_5, N_8, N_4, N_7, N_3, N_2, N_1]$
18*	$\{N_{19}, N_{16}\}$	$[N_{15}, N_{18}, N_{12}, N_{14}, N_{17}, N_{11}, N_{13}, N_6, N_9, N_{10}, N_5, N_8, N_4, N_7, N_3, N_2, N_1]$
19*	$\{N_{16}, N_{20}\}$	$[N_{19}, N_{15}, N_{18}, N_{12}, N_{14}, N_{17}, N_{11}, N_{13}, N_6, N_9, N_{10}, N_5, N_8, N_4, N_7, N_3, N_2, N_1]$
20*	$\{N_{20}\}$	$[N_{16}, N_{19}, N_{15}, N_{18}, N_{12}, N_{14}, N_{17}, N_{11}, N_{13}, N_6, N_9, N_{10}, N_5, N_8, N_4, N_7, N_3, N_2, N_1]$
21*	$\{N_{21}\}$	$[N_{20}, N_{16}, N_{19}, N_{15}, N_{18}, N_{12}, N_{14}, N_{17}, N_{11}, N_{13}, N_6, N_9, N_{10}, N_5, N_8, N_4, N_7, N_3, N_2, N_1]$
22	{Empty}	$[N_{21}, N_{20}, N_{16}, N_{19}, N_{15}, N_{18}, N_{12}, N_{14}, N_{17}, N_{11}, N_{13}, N_6, N_9, N_{10}, N_5, N_8, N_4, N_7, N_3, N_2, N_1]$

To avoid repetition of the states, a check is done at every level as shown by * in Table 4.3. For example, when child node is explored, the state is validated and added to the domain, however if child node is carrying the state which is already in the queue for exploration then the node is not considered. For the toy model, the average number of walk between the nodes is $\mathbb{T} \approx 2$.

The states, $state(N_i) = X_i$ is updated in the domain in every iteration given in Table 4.4 based on the LAS update trend. For example, in *step* 4, nodes N_4 and N_8 were added to the *queue* when the state of N_3 was considered for the domain, but in the next *step* 5 when the state of N_7 was considered for the domain, no new node was added in the *queue* because node N_8 was already in the *queue*, this was the neighbour of both N_3 and N_7. The corresponding state in node N_8 will be added to the domain once and will not be repeated in the domain through N_7 but the changes in propensity $\Delta a_{i,j}$ will be validated. When the nodes are pulled from the *queue*, the states are updated according to occurrence of reaction.

Table 4.4. *LAS* state update strategy for the toy model.

Iterations	Depth of exploration	Corresponding propensities $\Delta a_{i,j}$
1	1	Empty
2	1	$\Delta a_{1,2}$
3	1 ⟶ 2	$\Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
4*	2 ⟶ 3	$\Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
5	3	$\Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
6*	3 ⟶ 4	$\Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
7*	4	$\Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
8*	4 ⟶ 5	$\Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
9*	5	$\Delta a_{10,13}, \Delta a_{10,11}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
10	5	$\Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{10,13}, \Delta a_{10,11}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
11*	5 ⟶ 6	$\Delta a_{6,17}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{10,13}, \Delta a_{10,11}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
12*	6	$\Delta a_{13,17}, \Delta a_{13,14}, \Delta a_{13,4}, \Delta a_{6,17}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{10,13}, \Delta a_{10,11}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
13*	6	$\Delta a_{11,14}, \Delta a_{11,12}, \Delta a_{11,8}, \Delta a_{13,17}, \Delta a_{13,14}, \Delta a_{13,4}, \Delta a_{6,17}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{10,13}, \Delta a_{10,11}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
14*	6 ⟶ 7	$\Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{11,14}, \Delta a_{11,12}, \Delta a_{11,8}, \Delta a_{13,17}, \Delta a_{13,14}, \Delta a_{13,4}, \Delta a_{6,17}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{10,13}, \Delta a_{10,11}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
15*	7	$\Delta a_{14,18}, \Delta a_{14,15}, \Delta a_{14,10}, \Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{11,14}, \Delta a_{11,12}, \Delta a_{11,8}, \Delta a_{13,17}, \Delta a_{13,14}, \Delta a_{13,4}, \Delta a_{6,17}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{10,13}, \Delta a_{10,11}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
16*	7	$\Delta a_{12,15}, \Delta a_{12,9}, \Delta a_{14,18}, \Delta a_{14,15}, \Delta a_{14,10}, \Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{11,14}, \Delta a_{11,12}, \Delta a_{11,8}, \Delta a_{13,17}, \Delta a_{13,14}, \Delta a_{13,4}, \Delta a_{6,17}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{10,13}, \Delta a_{10,11}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$

17*	7 ⟶ 8	$\Delta a_{18,19}, \Delta a_{18,13}, \Delta a_{12,15}, \Delta a_{12,9}, \Delta a_{14,18}, \Delta a_{14,15}, \Delta a_{14,10}, \Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{11,14}, \Delta a_{11,12}, \Delta a_{11,8}, \Delta a_{13,17}, \Delta a_{13,14}, \Delta a_{13,4}, \Delta a_{6,17}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{10,13}, \Delta a_{10,11}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
18*	8	$\Delta a_{15,19}, \Delta a_{15,16}, \Delta a_{15,11}, \Delta a_{18,19}, \Delta a_{18,13}, \Delta a_{12,15}, \Delta a_{12,9}, \Delta a_{14,18}, \Delta a_{14,15}, \Delta a_{14,10}, \Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{11,14}, \Delta a_{11,12}, \Delta a_{11,8}, \Delta a_{13,17}, \Delta a_{13,14}, \Delta a_{13,4}, \Delta a_{6,17}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{10,13}, \Delta a_{10,11}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
19*	8 ⟶ 9	$\Delta a_{19,20}, \Delta a_{19,14}, \Delta a_{15,19}, \Delta a_{15,16}, \Delta a_{15,11}, \Delta a_{18,19}, \Delta a_{18,13}, \Delta a_{12,15}, \Delta a_{12,9}, \Delta a_{14,18}, \Delta a_{14,15}, \Delta a_{14,10}, \Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{11,14}, \Delta a_{11,12}, \Delta a_{11,8}, \Delta a_{13,17}, \Delta a_{13,14}, \Delta a_{13,4}, \Delta a_{6,17}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{10,13}, \Delta a_{10,11}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
20*	9	$\Delta a_{16,20}, \Delta a_{16,12}, \Delta a_{19,20}, \Delta a_{19,14}, \Delta a_{15,19}, \Delta a_{15,16}, \Delta a_{15,11}, \Delta a_{18,19}, \Delta a_{18,13}, \Delta a_{12,15}, \Delta a_{12,9}, \Delta a_{14,18}, \Delta a_{14,15}, \Delta a_{14,10}, \Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{11,14}, \Delta a_{11,12}, \Delta a_{11,8}, \Delta a_{13,17}, \Delta a_{13,14}, \Delta a_{13,4}, \Delta a_{6,17}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{10,13}, \Delta a_{10,11}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
21*	9 ⟶ 10	$\Delta a_{20,21}, \Delta a_{20,15}, \Delta a_{16,20}, \Delta a_{16,12}, \Delta a_{19,20}, \Delta a_{19,14}, \Delta a_{15,19}, \Delta a_{15,16}, \Delta a_{15,11}, \Delta a_{18,19}, \Delta a_{18,13}, \Delta a_{12,15}, \Delta a_{12,9}, \Delta a_{14,18}, \Delta a_{14,15}, \Delta a_{14,10}, \Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{11,14}, \Delta a_{11,12}, \Delta a_{11,8}, \Delta a_{13,17}, \Delta a_{13,14}, \Delta a_{13,4}, \Delta a_{6,17}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{10,13}, \Delta a_{10,11}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
22	10 ⟶ 11	$\Delta a_{21,16}, \Delta a_{20,21}, \Delta a_{20,15}, \Delta a_{16,20}, \Delta a_{16,12}, \Delta a_{19,20}, \Delta a_{19,14}, \Delta a_{15,19}, \Delta a_{15,16}, \Delta a_{15,11}, \Delta a_{18,19}, \Delta a_{18,13}, \Delta a_{12,15}, \Delta a_{12,9}, \Delta a_{14,18}, \Delta a_{14,15}, \Delta a_{14,10}, \Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{11,14}, \Delta a_{11,12}, \Delta a_{11,8}, \Delta a_{13,17}, \Delta a_{13,14}, \Delta a_{13,4}, \Delta a_{6,17}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{10,13}, \Delta a_{10,11}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{8,10}, \Delta a_{8,9}, \Delta a_{8,2}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$

The response of LAS for the state-space expansion is shown in Figure 4.6. It clearly shows how the size of the domain (as *2D pyramids*) increases with the addition of new probable states. The model is solved for $t_f = 1.0sec$ by taking $t_{step} = 0.01sec$, $\tau_m = 1e-6$.

The probability bunked during the expansion and approximation of the model is shown in Figure 4.7. There were no reversible reaction in the model and all R_M have similar kinetic parameters, so the bunking error became constant from $t = 0.05\ sec$ after losing $2.21e-02$ of probability in the approximation until t_f.

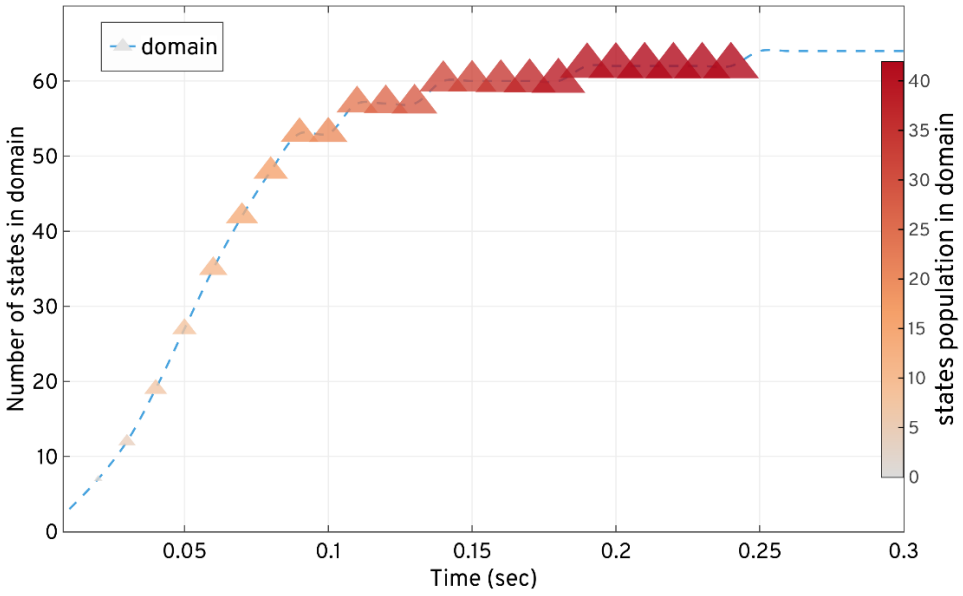


Figure 4.6. Based on the *LAS* strategy, the response shows how the size of the domain increases with addition of new states with time in the toy model.

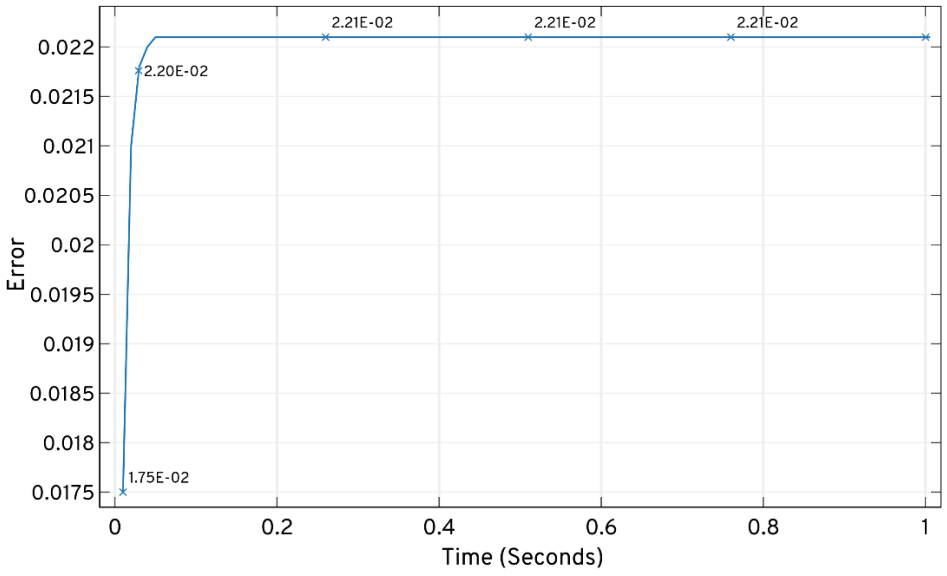


Figure 4.7. Based on the *LAS* strategy, the response shows the total bunked probability during the approximation of the toy model.

The *time-space* complexity of *LAS* for this model has a linear behaviour as shown in Figure 4.8, for cases like ($\mathrm{Ŧ} = 2, \mathrm{đ}_l = 4$), and ($\mathrm{Ŧ} = 5, \mathrm{đ}_l = 10$).

The stochastic behaviour of biochemical systems are such that the state-space depends on the propensity of each R_M reaction, and the size of the domain depends upon the number of state explored. *LAS* successfully creates the domain of an optimum order with 66 states at t_f by introducing new states to the domain with time, as shown in the Figure 4.9.

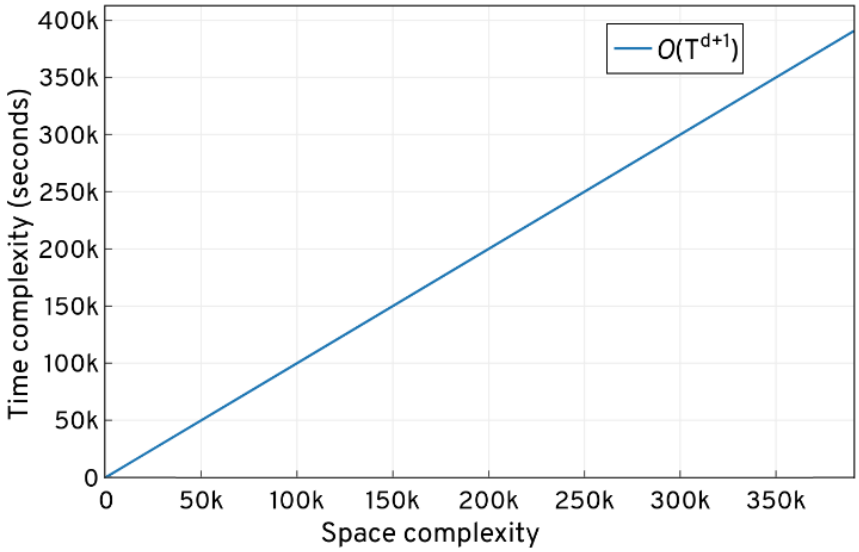


Figure 4.8. Time-Space complexity of the *LAS* response shows the linear behaviour at different average transition factors and the depth of the end state. For cases like ($\mathbb{T} = 2, \mathbb{d}_l = 4$), and ($\mathbb{T} = 5, \mathbb{d}_l = 10$), *LAS* has the same response.

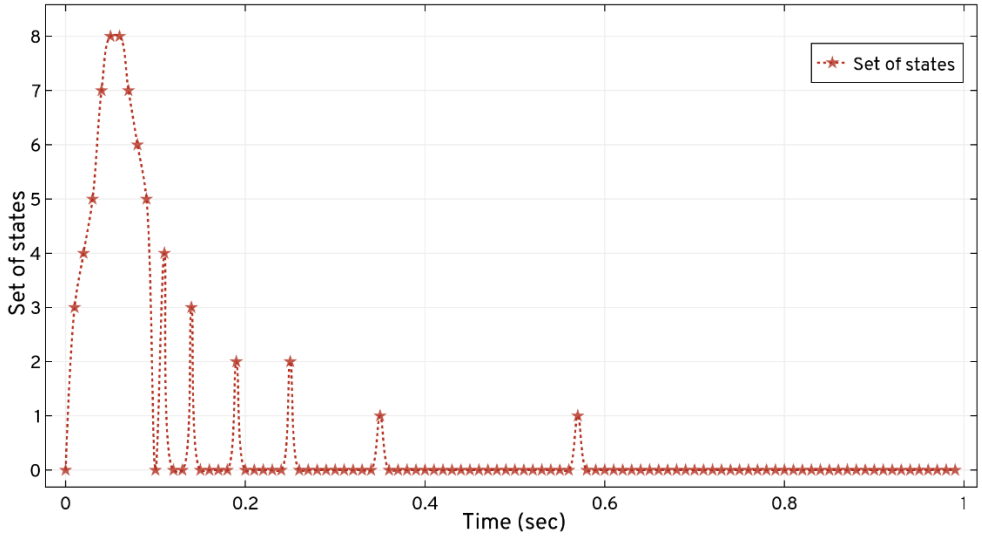


Figure 4.9. Shows the set of states explored for the toy model after every *LAS* iteration. Based on the reactions, *LAS* unfolds the state-space pattern to update states in the domain and expands 66 probable states in 1.0 *sec*. ★ shows the time point where new set of states is explored and updated in the domain.

The computational expense estimation for exploring this system is given at different time points in Table 4.5. For large biochemical systems, the expense and state-space will vary with the number of states explored. Ideally, if the species present in the model decay with time or because no new states are found in the expansion, then the size of the domain remains the same even if it is solved for a higher t_f.

Table 4.5. *LAS* expansion of the state-space and solution of the toy model at different time points.

Time (sec)	No. of states explored	Exploration performance time (sec)	Probability lost	Approximation (probability)
0.0	0	0.0	0	0
0.01	3	0.00332	0.0175	0.982
0.05	27	0.02082	0.0221	0.977
0.1	53	0.03842	0.0221	0.977
1.0	66	0.00910	0.0221	0.977

Similarly, in section 4.3.2, we apply the *ISP LAS* on the real biological model of the catalytic reaction system to study the expansion of the state-space and its solution.

4.3.2 Biological Example

Validation of the *ISP LAS* on the biochemical model - catalytic reaction system (Mastny et al., 2007) is shown below. To demonstrate the *ISP LAS* algorithm, we first consider the model defined by the reactions:

$$S \overset{k_1}{\rightarrow} B \overset{k_2}{\rightarrow} C, \quad B + P \overset{k_3}{\rightarrow} B + E \tag{88}$$

depicted as a network in Figure 4.10 as:

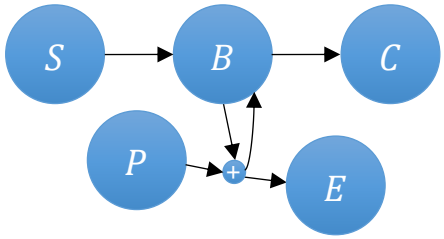


Figure 4.10. Catalytic reaction network having five $\tilde{N} = 5$ species *S*, *B*, *C*, *P*, and *E* in a network defining reactions, as given in Eq. (88).

In this biochemical system (dimension = 5), reactant P will transform into product E via complex B when reactant S acts as a catalyst for the reaction and produces C. We rewrite this catalytic reaction system as a network of three reactions:

$$\boldsymbol{R_1}: \quad S \overset{k_1}{\rightarrow} B, \tag{89}$$

$$\boldsymbol{R_2}: \quad B \overset{k_2}{\rightarrow} C, \tag{90}$$

$$\boldsymbol{R_3}: \quad B + P \overset{k_3}{\rightarrow} B + E\ ; \tag{91}$$

with initial copy counts $S_0 = 50$, $P_0 = 80$, $B_0 = C_0 = E_0$ and reaction rate parameters $k_1 = 1$, $k_2 = 1000$, $k_3 = 100$. These species counts are used as a state-space to define the model and these copy counts are tracked as:

$$([S], [B], [C], [P], [E]) \in \tilde{N} := (x_0, x_1, x_2, x_3, x_4).$$

In reaction R_1, the copy count of S is reduced by 1, which increases the copy count of B by 1. In reaction R_2 the copy count of B is reduced by 1, which increases the copy count of C by 1. Whereas, reaction R_3 decreases the count of B and P by 1 and increases the counts of B and E by 1. As in R_3, B act as a catalyst to convert P to E and B is retained in the same reaction.

We can now define the transitions associated with R_1, R_2, R_3 in stoichiometric vector V_M matrix as:

$$V_M = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}. \quad (92)$$

For LAS method compatibility, the associated Markov chain of this model is converted into a Markov chain tree, and the states in terms of the nodes with additional information such as the number of R_M reactions required to reach the state. In the growing Markov chain tree, the transition between the nodes:

$$N_i \xrightarrow{v_\mu(X_0(t), X_1(t), \ldots, X_K(t))} N_{i+1}\,, \quad (93)$$

is defined in the typical form of the dictionary $Dict$. We express the propensity functions of the three reactions in terms of the states $([S], [B], [C], [P], [E]) \in \tilde{\mathrm{N}}$ as:

$$\boldsymbol{R_1}: \quad a_1([S], [B], [C], [P], [E]) = k_1([S]),$$

$$\boldsymbol{R_2}: \quad a_2([S], [B], [C], [P], [E]) = k_2([B]),$$

$$\boldsymbol{R_3}: \quad a_3([S], [B], [C], [P], [E]) = k_3([B][P]).$$

Node $N_1 = (X_0, \mathrm{d}_1)$ carries the initial state X_0 of the system at an initial depth of level 1. Further $\mathbf{n}_J = (\mathbf{X}_K, \mathrm{d}_{1,2,\ldots\ldots})$ is expanded and the states updated by following the order of LAS. The corresponding propensities $\Delta a_{i,j}$ are updated in $A_{i,j}$ matrix in every iteration based on the LAS updating trend (see Table 4.4). Initially, the system started with $S_0 = 50$, $P_0 = 80$ and gradually all of the rectants are transformed to products, E and B, resulting in the system ending in $\mathbf{n}_J = (X_{1,2,\ldots\ldots 14666}, \mathrm{d}_l)$.

Figure 4.11 shows the response of the LAS method when solved with $\tau_m = 1e-6$ for $t_f = 0.5$ *sec*. Due to the nature of the model reaction rates, small steps $t_{step} = 0.01$ *sec* are taken to capture the moments based on non-negative non-zero states for the domain. LAS successfully creates the domain of an optimum order with 14666 states at t_f by introducing the new states to the domain with time, as shown in Figure 4.11.

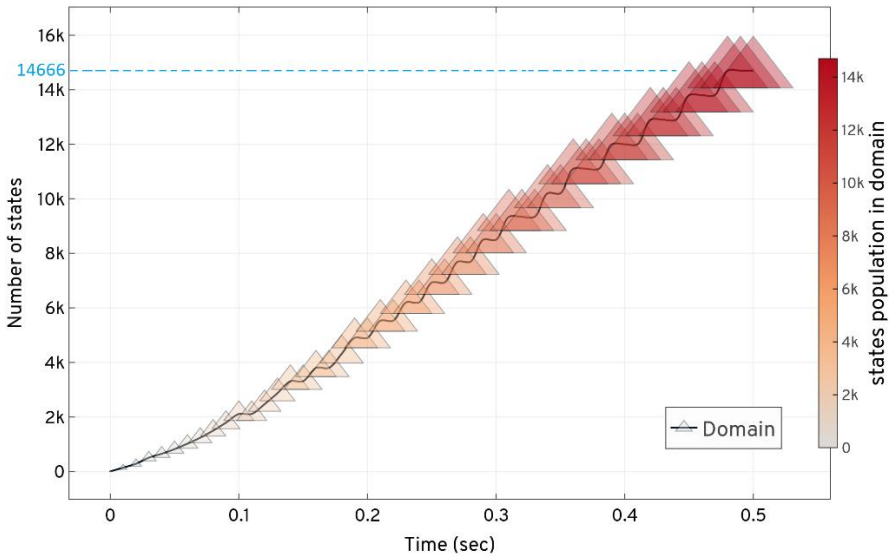


Figure 4.11. Expansion and updation of the states for the catalytic reaction system based on the *LAS* method. The state-space expansion increases the number of addition of new states in the domain. The size and colour of ▲ shows the increase in the size of the domain with the states population.

This pattern also depicts that the frequency (number of states at any time t) of expansion increases in depth when number of active reactions increase in the system. With the addition of probable states, the domain contains enough probability mass to approximate the solution up to t_f. The states are updated in sets as seen in Figure 4.12, for the catalytic system after every iteration.

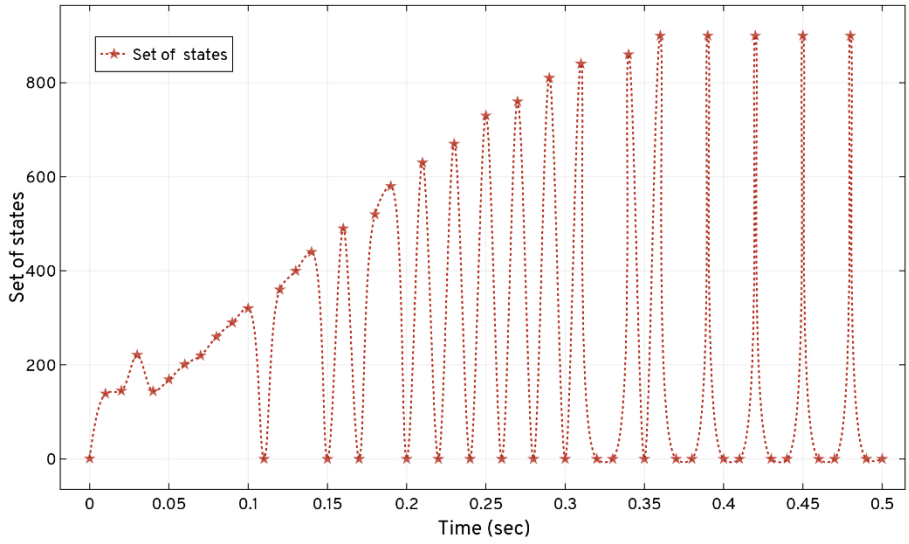


Figure 4.12. Shows the set of states explored for the catalytic system after every *LAS* iteration. Based on network of reactions, *LAS* unfolds the state-space pattern to update the states in the domain and expands 14666 probable states in 0.5 *sec*. ★ shows the time point where new sets of states are explored and updated in the domain.

The state-space pattern in Figure 4.12 can be used as a *blueprint* of the catalytic systems' state-space to compare with other model's *blueprints* for their characteristics and occurrence of the reactions. Such a pattern is considered to predict the behaviour of large network state-space expansions when the set of occurrences of the initial reactions are similar in different systems. The solution of Eq. (9) up to t_f, for the domain created by LAS is shown in Table 4.6 and the system's conditional probabilities based on its species are, as shown in Figure 4.13.

Table 4.6. LAS expansion response and solution at t_f for the catalytic system.

$t_f = 0.5$, $t_{step} = 0.01$	Run-time (sec)	Domain	Expansion time (sec)	Error at t_f
$ISP\ LAS$	4677	14666	0.5	$1.865\text{e}-05$

In three test runs, the run time of $ISP\ LAS$ for the catalytic system was 4677 *secs* when solving the Eq. (9) with 14666 states. The probability of the species in Figure 4.13 shows the nature of the reactions affecting the counts of each species in the system. The involvement of species B in all the reactions results in its highest probability at t_f. Species B also acts as a catalyst for R_3, converting species P to E; therefore, both have equal probabilities at the time of solution.

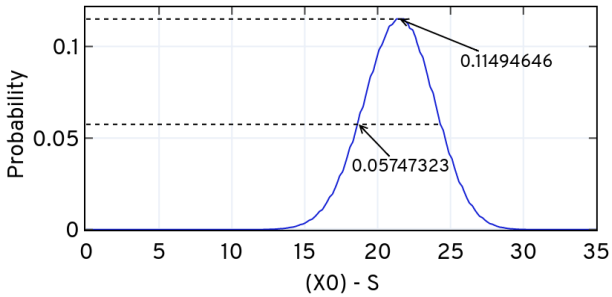


Fig (A). Probability of S over t_f

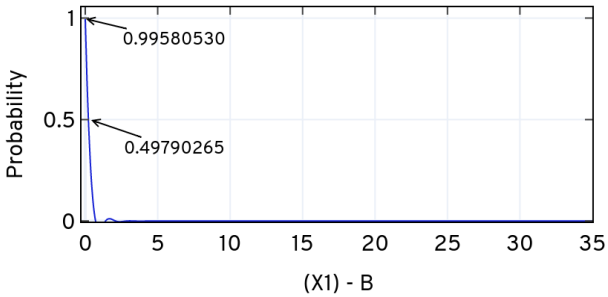


Fig (B). Probability of B over t_f

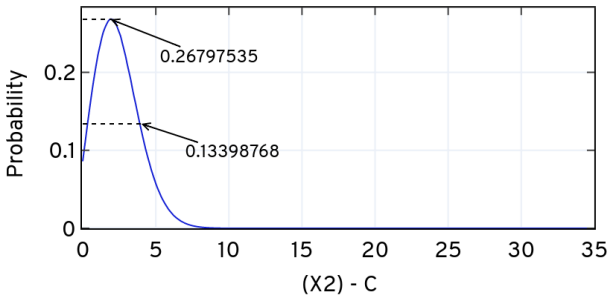


Fig (C). Probability of C over t_f

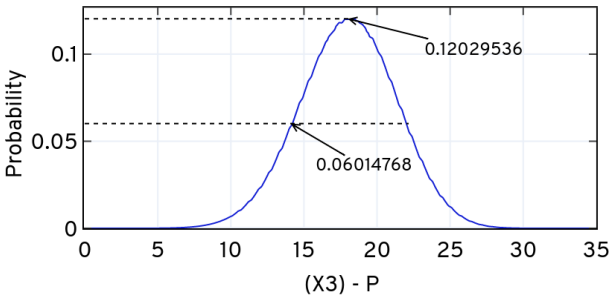


Fig (D). Probability of P over t_f

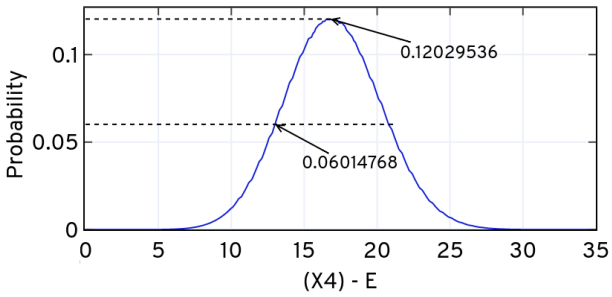


Fig (E). Probability of E over t_f

Figure 4.13. Conditional probability of the catalytic system evaluated at $t_f = 0.5$ *sec*, $t_{step} = 0.01$ using *LAS*. Fig (A) is the probability of the species S over t_f, Fig (B) is the probability of the species B over t_f, Fig (C) is the probability of the species C over t_f, Fig (D) is the probability of the species P over t_f, Fig (E) is the probability of the species E over t_f.

Figure 4.14 shows the total probability bunked at t' while progressing with the expansion. Bunking produces an error (w.r.t approximation) with time when the number of states increased with the expansion and provided that LAS produces minimal error of order 10^{-5}, as given in Table 4.6.

The comparative study of LAS with other methods such as SSA and *r-step reachability* used in SW and $OFSP$, respectively, are discussed further in section 5.1. In section 4.3.2, we use the principles and conditions discussed in Chapter 3, section 4.2.1 and 4.2.2, and introduce the second sub-routine of ISP called *Longitudinal Latitudinal Search*.

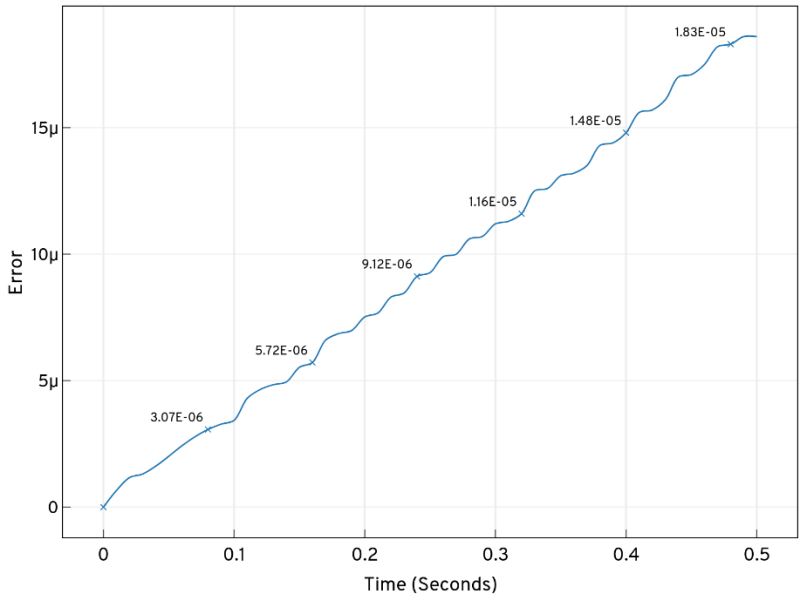


Figure 4.14. Total probability of states bunked at t' from the domain of catalytic system produced by *ISP LAS* iteration while expansion and solving the CME.

4.4 Longitudinal Latitudinal Search

In this section, we delve deeper into the second sub-routine of ISP called the *Intelligent State Projection Longitudinal Latitudinal Search* ($ISP\ LOLAS$). Figure 4.15 manifest the infrastructure of the $LOLAS$ strategy, showing the G_{mc}, $stack$ and the domain. The $stack$ data structure of $LOLAS$ is based on $LIFO$ (Last In, First Out) method. It assume that newest state added to the $stack$ is considered first. In particular, we define the bound limit (as discussed in section 4.2.1) and exploit the direction of the expansion step-by-step based on intuitive knowledge (as discussed in section 3.5) gained from the probability of future reaction events and follow the conditions (as discussed in section 4.2). Furthermore, we will show step-by-step how nodes carrying states are explored in bidirectional way and how these states were updated in the domain in I_{tr} iterations. To give a broader perspective, we compare this class of subroutine in Chapter 5 with the existing techniques, as discussed in Chapter 2. Finally, we demonstrate the $ISP\ LOLAS$ application on a large biochemical system in case study 1 - Chapter 6. We will also provide some practical guidelines for setting up $ISP\ LOLAS$ for the experiment.

The states at level $\mathfrak{a}_l$ are expanded only after the neighbouring states at level $\mathfrak{a}_l - 1$ have been expanded for R_M, i.e., the search is undertaken *level-by-level* and depth $\mathfrak{a}_l$ increases in the same I_{tr} iteration up to a certain $\mathfrak{b}_{limit}$ (bound limit). The limit of the expansion is set by $\mathfrak{b}_{limit}$, $\mathfrak{a}_{step}$ (depth step) but not by the depth $\mathfrak{a}_l$, as in LAS. The $LOLAS$ search updates the dictionary $Dict$ of G_{mc} by the stoichiometric vector function, $v_{\mu}(X(t))$ on state at level $\mathfrak{a}_l$ to explore the child nodes carrying states on levels $\mathfrak{a}_l + 1$, $\mathfrak{a}_l + 2$ ….. $\mathfrak{a}_l + l$, where $l = \{1,2 \ldots . \infty\}$ and then *retracts* to level $\mathfrak{a}_l$ at which new state exploration decisions can be made. In case of networks with *reversible* reactions, the ISP conditions will prevent the $LOLAS$ to return to the state it came from and prevent transitions containing cycles resulting in DAG with no repetition of any state whatsoever; however, the change in propensities $a_{i,j}$ is validated. Verifying the explored states in $\mathbf{X}_K$ in iterations ensures that the algorithm completes and deadlocks in the state transition cycles cannot occur.

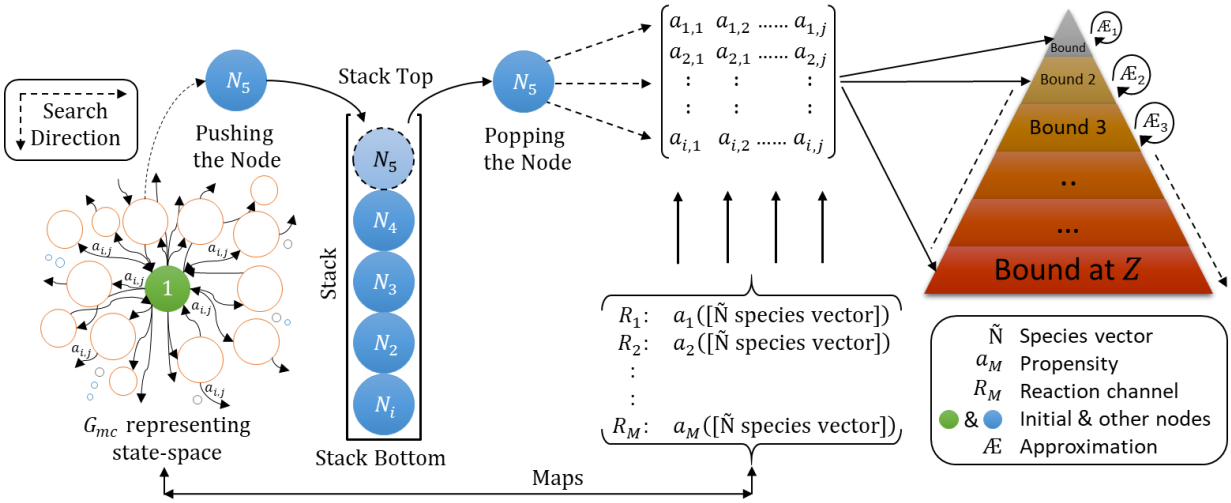


Figure 4.15. Infrastructure of the *Longitudinal Latitudinal Search* strategy, showing the G_{mc}, the *stack* and the domain.

In the absence of $б_{limit}$, the algorithm will not retract and will persist to explore longitudinally by tracking only one R_M reaction. In addition, the algorithm will not terminate with an optimal order domain carrying a maximum probability mass, leading to an increase in the approximation error. Instead, it will terminate when carrying only those set of states as a result of tracking for only a few R_M, creating an insufficient domain for approximation. Therefore, by default, the value of $б_{limit} \geq 1$ is kept for large systems and can be increased depending upon the dimension of the model and the availability of the random access memory (RAM) of the testing environment. The worst-case *time complexity* of $LOLAS$ depends on the average transitioning factor $ŧ$ and depth $đ_l$ is given by (see Appendix E for a detailed discussion):

$$(ŧ + 1)ŧ^0 + (ŧ)ŧ^1 + (ŧ - 1)ŧ^2 + \cdots \ldots + 3ŧ^{đ_l - 2} + 2ŧ^{đ_l - 1} + 1ŧ^{đ_l} = O(ŧ^{đ_l}). \quad (94)$$

$LOLAS$ only stores the transition path to the end state besides the neighbours of each relevant node in the exploration and then discards the node from the domain (explored) once all descendants are updated with the relevant propensities in the projection, ready for the approximation. $LOLAS$ first considers the R_1 reaction and the corresponding stoichiometric vector v_1 of the system, to explore all the neighbouring states up to bound limit $б_{limit}$, and then considers R_2, R_3,…..,R_M for same $б_{limit}$ and the corresponding v_2, v_3,….. , v_M to explore the states. For $count(б_{limit})$, $LOLAS$ retracts to R_1 reaction and explores the new neighbouring states longitudinally and then reconsiders R_2, R_3,….. , R_M to explore the other states in a similar fashion. Provided with this pattern of tracking the reactions, the $BLNP$ function alters this trend and guides this tracking by considering reactions in different order based on their propensities and number of probable states of the system.

If the system is ending in a set of state $\mathbf{X}_K$ carried by $\mathbf{n}_K$ at t_f, then $LOLAS$ will explore the states efficiently as long as $count(б_{limit}) \leq б_{limit}$, otherwise $count(б_{limit})$ is reset for further expansion. Choosing the appropriate $б_{limit}$ and $đ_{step}$ depends on the type of biochemical reaction network and the computing configuration.

Starting with depth $1 \longrightarrow б_{limit}$, $LOLAS$ explores all the states until they return $null$, and then it resets the $count(б_{limit})$ and *retracts* to explore again. In most cases, fewer states are positioned at the lower level and increase at a higher level when the number of active R_M reactions increases, so retracting gives the freedom to track all the reactions simulatenously. Due to the nature of the $LOLAS$ expansion, this finds more states at any time t compared to LAS and also at the deepest level of the graph. The states at depth $đ_l$ are explored once, the

states at depth $\mathfrak{d}_l - 1$ are explored twice, states at depth $\mathfrak{d}_l - 2$ are explored three times and so on, until it has explored all the states of the system. If the input τ_m is too small, the default value of $sqrt(eps)$ is automatically used by the algorithm. Where, $sqrt$ is the square root and eps is the default value of the epsilon on machine. The expansion of child nodes containing $state(N_i) = X_i$ stops if the condition of Eq. (82) is not satisfied. If criterion of *slow and fast* reaction is considered, then condition of Eq. (81) or (82) is used depending on number of $R_{M(sr)}$ and $R_{M(fs)}$. Table 4.7 shows the steps of the $LOLAS$ method with an embedded $BLNP$ function from step 4a to 5b.

Table 4.7. Steps of *ISP* Longitudinal Latitudinal Search (*LOLAS*) Algorithm.

Step 0:	**Inputs:** Initial node N_0, $\mathfrak{d}_{step}$, $\mathfrak{b}_{limit}$, a_μ, v_μ, tol τ_m, t_f, t_{step} **Initialise:** $Bound_{lower} = \mathbf{X}_K$, $b'_{N',N}(X - v_\mu)' = P^{(t)}(X_0)$, $A = [\,]$
Step 1:	Initialise $count(\mathfrak{b}_{limit})$ and start from parent node $N_i = (X_0, \mathfrak{d}_l) \longleftarrow$ Current State of the system at t_d,
Step 2:	Flag the current node as explored, update A and add the state X_i in the domain so that; *if* $1 - I^T exp(t.A_j).P^{(t)}(X_0) \geq \tau_m(leak)$ holds true go to *Step 3*; *else* stop the algorithm.
Step 3:	Sort $exp(t.A_j).P^{(t)}(X_0)$ and shift the set of states in $\mathbf{X}'_K$ at t' having smallest probabilities, if $P^{(t)}(\mathbf{X}_K) \geq \tau_m(leak) > P^{(t)}(\mathbf{X}'_K)$ and at t_d update $\mathbf{X}_K \longleftarrow \mathbf{X}_K - \mathbf{X}'_K$
Step 4a:	For $\mathfrak{d}_{step}$, extend the graph dictionary $Dict$ by $v_\mu(X_i(t))$ for $count(\mathfrak{b}_{limit})$ to check all the nodes $\mathbf{n}_J = \left(\mathbf{X}_J, \mathfrak{d}_l, \mathfrak{c}_{N_i,N'_i}(min)\right)$ adjacent to N_i: $Bound_{upper} \leftarrow R_M(Bound_{lower})$ reachable by exactly R_M reactions (from fast to slow) having $\mathfrak{c}_{N_i,N'_i}(min)$. If $\mathbf{n}_K = \left(\mathbf{X}_K, \mathfrak{d}_l, \mathfrak{c}_{N_i,N'_i}(min)\right)$ be the set of adjacent nodes such that $\mathbf{n}_K \in \mathbf{n}_J$, then go to the next *Step*,
Step 4b:	Compute the $BLNP$ function for $\mathbf{n}_K \in Bound_{upper}$: $b(N_{N1,..NM} \mid b'_{1,N....N',N}) = P_{N,1....N,N'}(\omega) * b'_{N',N}(X - v_i)'$
Step 5a:	If $\mathbf{n}_K = \left(\mathbf{X}_K, \mathfrak{d}_l, \mathfrak{c}_{N_i,N'_i}(min)\right) \in domain$, then *update* the values of the set of states $\mathbf{X}_K$ present in $domain$ and take $domain \longleftarrow domain_{previous} \cup domain$ and go back to *Step 1*; *else* If $\mathbf{n}_K = \left(\mathbf{X}_K, \mathfrak{d}_l, \mathfrak{c}_{N_i,N'_i}(min)\right) \notin domain$, then add it to the $stack$ in order, according to reachability and go to next *Step*,
Step 5b:	*sort* $b(N_{N1,..NM} \mid b'_{N1,..NM})$ in descending order and update $stack \longleftarrow (stack\ ;\ b(N_{N1,..NM} \mid b'_{1,N....N',N}))$
Step 6:	Pop of the top nodes $\mathbf{n}_K = \left(\mathbf{X}_K, \mathfrak{d}_l, \mathfrak{c}_{N_i,N'_i}(min)\right)$ from the $stack$ and add the set of states $\mathbf{X}_K$ in the domain as $domain \longleftarrow domain + \mathbf{X}_K$ and take $domain_{previous} \cup domain$, and go to next *Step*,
Step 7:	If $count(\mathfrak{b}_{limit}) = \mathfrak{b}_{limit}$ creates $Bound_{upper} = \{domain\}\ up\ to\ \mathfrak{b}_{limit}$ then label $Bound_{lower} \leftarrow Bound_{upper}$ and go back to *Step* 1; else *if* $count(\mathfrak{b}_{limit}) < \mathfrak{b}_{limit}$ creates $\{domain\}\ up\ to\ count(\mathfrak{b}_{limit})$ then go to next *Step*,
Step 8:	$count(\mathfrak{b}_{limit}) \leftarrow count(\mathfrak{b}_{limit}) + 1$ and go to *Step 4a*
	Output: $domain$ with probable states

To demonstrate the $ISP\ LOLAS$ algorithm, we assume the same toy model system as discussed in section 4.3. In section 4.4.1, we demonstrate the $LOLAS$ expansion and update strategy on the toy model.

4.4.1 Expansion and Update

Nodes $\mathbf{n}_J = \left(\mathbf{X}_K, \mathfrak{d}_l, \mathfrak{C}_{N_i,N_i'}(min)\right)$ carrying states are expanded, as shown by G_{mc} in number of stages ($\hat{\mathrm{S}}$), assuming $\mathfrak{d}_{step} = 1$, $\mathfrak{b}_{limit} = 2$. From Figs (A) to (F) in Figure 4.16 represent the systematic exploration of the nodes carrying states based on $LOLAS$ and as depicted up to six stages ($\hat{\mathrm{S}} = 6$), and walk representing R_M reactions with propensities $a_{i,j}$. This shows the change in state from $state(N_i) = X_i$ and $state(N_i') = X_{i'}$ of the system at time, t.

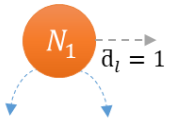


Fig (A). Stage 1, $\mathfrak{d}_l = 1$

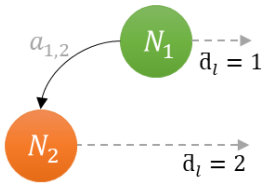


Fig (B). Stage 2, $\mathfrak{d}_l = 2$

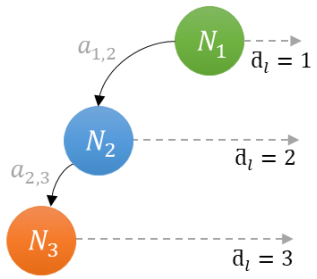


Fig (C). Stage 3, $\mathfrak{d}_l = 3$

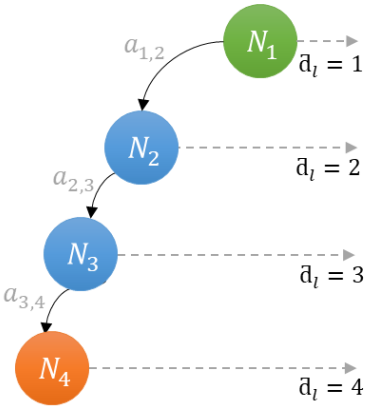


Fig (D). Stage 4, $\mathfrak{d}_l = 4$

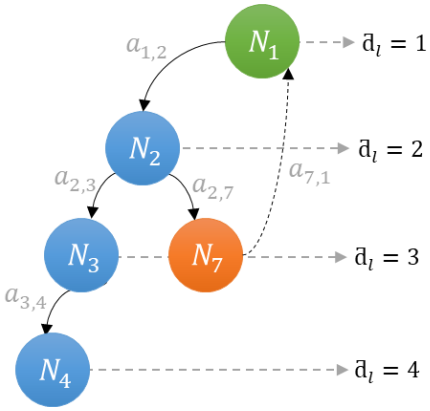


Fig (E). Stage 5, $\mathfrak{d}_l = 4$

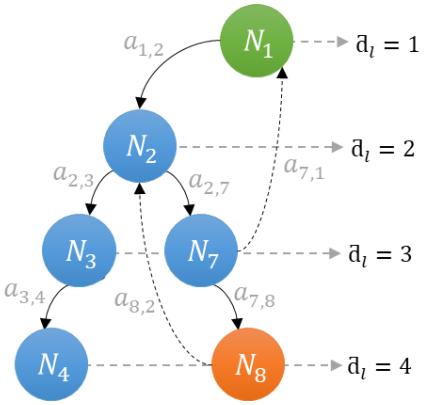


Fig (F). Stage 6, $\mathfrak{d}_l = 4$

Figure 4.16. Six stages of state exploration based on *LOLAS* search for the toy model, given average transitioning factor $\mathfrak{T} = 2$, assumed $\mathfrak{d}_{step} = 1$, $\mathfrak{b}_{limit} = 3$. Fig (A) is the first stage that denote the initial node carrying initial state of the system, Fig (B) is the second stage that denote all the nodes with new node at $\mathfrak{d}_l = 2$, Fig (C) is the third stage that denote all the nodes with new node at $\mathfrak{d}_l = 3$, Fig (D) is the fourth stage that denote all the nodes with new node at $\mathfrak{d}_l = 4$, Fig (E) is the fifth stage that denote all the nodes with new node at $\mathfrak{d}_l = 4$, Fig (F) is the sixth stage that denote all the nodes with new node at $\mathfrak{d}_l = 4$.

According to $LOLAS$ strategy in Table 4.7, for R_1, explores the states up to $\mathfrak{d}_l = 4$ as shown in Fig (D) of Figure 4.16 and retracts to initial node N_1 to track R_2. However, R_2 only occurs when the system is at N_2. Therefore, the system will consider the expansion from N_2 for R_2 and explore until $\mathfrak{d}_l = 4$. It retracts continuously for all the R_M reactions' up to $\mathfrak{b}_{limit} = 2$ and once all the states are explored up to $\mathfrak{d}_l = 4$ for all the reactions, it reconsider R_1 and other reactions to further explore and retract.

The toy model is solved for $t_f = 1.0\ sec$ by taking $t_{step} = 0.01sec$, $\tau_m = 1e-6$ with the initial state, $state(N_1) = X_0$, $\mathfrak{b}_{limit} = 3$. In *step 1*, the count of $\mathfrak{b}_{limit}$ is initialised and starts from the given initial state, $state(N_i) = X_i$. In *step 2*, matrix A is created by flagging the current node as visited and the state is added in the domain if $e_{rror} \geq \tau_m$; otherwise, it will stop the algorithm if it does not. In *step 3*, it *sorts* the probabilities of the expanded states and shifts the set of states in $\mathbf{X}'_K$ at t' having smallest probabilities, if $\tau_m(leak) > P^{(t)}(\mathbf{X}'_K)$ which leads to an optimal $\mathbf{X}_K \longleftarrow \mathbf{X}_K - \mathbf{X}'_K$ set.

In *step 4*, G_{mc} is extended for a count of $\mathfrak{b}_{limit}$ to check all the nodes $\mathbf{n}_J = \left(\mathbf{X}_J, \mathfrak{d}_l, \mathfrak{c}_{N_i,N'_i}(min)\right)$ adjacent to N_i reachable exactly by R_M reactions having $\mathfrak{c}_{N_i,N'_i}(min)$. In *step 5*, the values of the set of states $\mathbf{X}_K$ present in the $domain$ is updated if $\mathbf{n}_J \in domain$ and unique states was chosen from the $domain$ (current iteration) and $domain_{previous}$ (previous iteration); otherwise $\mathbf{n}_J$ is added to the $stack$ and moves to the next step. In *step 6*, the nodes are popped out from the $stack$ one-by-one and the corresponding set of states $\mathbf{X}_K$ are added to the domain, followed by creating a $domain$ comprising unique states. The nodes $\mathbf{n}_J = \left(\mathbf{X}_K, \mathfrak{d}_l, \mathfrak{c}_{N_i,N'_i}(min)\right)$ are then expanded as given in Table 4.3.

Table 4.8. *LOLAS* nodes expansion strategy for the toy model.

Iterations	Stack	Domain
1	$\{N_1\}$	[Empty]
2	$\{N_2\}$	$[N_1]$
3	$\{N_3, N_7\}$	$[N_2, N_1]$
4	$\{N_4, N_8, N_7\}$	$[N_3, N_2, N_1]$
5	$\{N_5, N_{10}, N_8, N_7\}$	$[N_4, N_3, N_2, N_1]$
6*	$\{N_9\}$	$[N_7, N_8, N_{10}, N_5, N_4, N_3, N_2, N_1]$
7*	$\{N_{13}, N_{11}\}$	$[N_9, N_7, N_8, N_{10}, N_5, N_4, N_3, N_2, N_1]$
8*	$\{N_{17}, N_{14}, N_{11}\}$	$[N_{13}, N_9, N_7, N_8, N_{10}, N_5, N_4, N_3, N_2, N_1]$
9*	$\{N_{18}, N_{14}, N_{11}\}$	$[N_{17}, N_{13}, N_9, N_7, N_8, N_{10}, N_5, N_4, N_3, N_2, N_1]$
10*	$\{N_{19}, N_{14}, N_{11}\}$	$[N_{18}, N_{17}, N_{13}, N_9, N_7, N_8, N_{10}, N_5, N_4, N_3, N_2, N_1]$
11*	$\{N_{20}, N_{14}, N_{11}\}$	$[N_{19}, N_{18}, N_{17}, N_{13}, N_9, N_7, N_8, N_{10}, N_5, N_4, N_3, N_2, N_1]$
12*	$\{N_{15}, N_{11}\}$	$[N_{14}, N_{20}, N_{19}, N_{18}, N_{17}, N_{13}, N_9, N_7, N_8, N_{10}, N_5, N_4, N_3, N_2, N_1]$
13*	$\{N_{16}, N_{11}\}$	$[N_{15}, N_{14}, N_{20}, N_{19}, N_{18}, N_{17}, N_{13}, N_9, N_7, N_8, N_{10}, N_5, N_4, N_3, N_2, N_1]$
14*	$\{N_{11}\}$	$[N_{16}, N_{15}, N_{14}, N_{20}, N_{19}, N_{18}, N_{17}, N_{13}, N_9, N_7, N_8, N_{10}, N_5, N_4, N_3, N_2, N_1]$
15*	$\{N_{12}\}$	$[N_{11}, N_{16}, N_{15}, N_{14}, N_{20}, N_{19}, N_{18}, N_{17}, N_{13}, N_9, N_7, N_8, N_{10}, N_5, N_4, N_3, N_2, N_1]$
16*	$\{N_{21}\}$	$[N_{12}, N_{11}, N_{16}, N_{15}, N_{14}, N_{20}, N_{19}, N_{18}, N_{17}, N_{13}, N_9, N_7, N_8, N_{10}, N_5, N_4, N_3, N_2, N_1]$
17	{Empty}	$[N_{21}, N_{12}, N_{11}, N_{16}, N_{15}, N_{14}, N_{20}, N_{19}, N_{18}, N_{17}, N_{13}, N_9, N_7, N_8, N_{10}, N_5, N_4, N_3, N_2, N_1]$

To avoid the repetition of states, a check is carried out at every level, as shown by * in Table 4.3. The algorithm rechecks the explored nodes and validates the propensities of the corresponding nodes, N_1, N_2, N_3, N_5, and then explores the new states through v_m to the same bound limit. In general, the algorithm prolongs the retraction until it explores all possible ways to update the states up to a certain $б_l$. If $₹_{tract}$ is the number of retraction, then condition $₹_{tract} \approx R_M$ holds true for $LOLAS$ for any biochemical system having R_M reactions. The states, $state(N_i) = X_i$ are updated in the domain in every iteration as given in Table 4.4, based on the $LOLAS$ update trend.

Table 4.9. *LOLAS* states update strategy for the toy model.

Iteration	Depth of exploration	Corresponding propensities $\Delta a_{i,j}$
1	1	Empty
2	1	$\Delta a_{1,2}$
3	1 ⟶ 2	$\Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
4	2 ⟶ 3	$\Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
5	2 ⟶ 4	$\Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
6*	4 ⟶ 5, 5 ⟶ 4, 4 ⟶ 3	$\Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{8,9}, \Delta a_{8,10}, \Delta a_{8,2}, \Delta a_{10,11}, \Delta a_{10,13}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
7*	3 ⟶ 5	$\Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{8,9}, \Delta a_{8,10}, \Delta a_{8,2}, \Delta a_{10,11}, \Delta a_{10,13}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
8*	5 ⟶ 6	$\Delta a_{13,14}, \Delta a_{13,17}, \Delta a_{13,4}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{8,9}, \Delta a_{8,10}, \Delta a_{8,2}, \Delta a_{10,11}, \Delta a_{10,13}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
9*	6 ⟶ 7	$\Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{13,14}, \Delta a_{13,17}, \Delta a_{13,4}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{8,9}, \Delta a_{8,10}, \Delta a_{8,2}, \Delta a_{10,11}, \Delta a_{10,13}, \Delta a_{10,3}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
10*	7 ⟶ 8	$\Delta a_{18,19}, \Delta a_{18,13}, \Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{13,14}, \Delta a_{13,17}, \Delta a_{13,4}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{8,9}, \Delta a_{8,10}, \Delta a_{8,2}, \Delta a_{10,11}, \Delta a_{10,13}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
11*	8 ⟶ 9	$\Delta a_{19,20}, \Delta a_{19,14}, \Delta a_{18,19}, \Delta a_{18,13}, \Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{13,14}, \Delta a_{13,17}, \Delta a_{13,4}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{8,9}, \Delta a_{8,10}, \Delta a_{8,2}, \Delta a_{10,11}, \Delta a_{10,13}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
12*	9 ⟶ 10, 10 ⟶ 7	$\Delta a_{14,15}, \Delta a_{14,18}, \Delta a_{14,10}, \Delta a_{20,21}, \Delta a_{20,15}, \Delta a_{19,20}, \Delta a_{19,14}, \Delta a_{18,19}, \Delta a_{18,13}, \Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{13,14}, \Delta a_{13,17}, \Delta a_{13,4}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{8,9}, \Delta a_{8,10}, \Delta a_{8,2}, \Delta a_{10,11}, \Delta a_{10,13}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
13*	7 ⟶ 8	$\Delta a_{15,16}, \Delta a_{15,19}, \Delta a_{15,11}, \Delta a_{14,15}, \Delta a_{14,18}, \Delta a_{14,10}, \Delta a_{20,21}, \Delta a_{20,15}, \Delta a_{19,20}, \Delta a_{19,14}, \Delta a_{18,19}, \Delta a_{18,13}, \Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{13,14}, \Delta a_{13,17}, \Delta a_{13,4}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{8,9}, \Delta a_{8,10}, \Delta a_{8,2}, \Delta a_{10,11}, \Delta a_{10,13}, \Delta a_{5,13}, \Delta a_{5,6}, \Delta a_{4,10}, \Delta a_{4,5}, \Delta a_{3,8}, \Delta a_{3,4}, \Delta a_{2,7}, \Delta a_{2,3}, \Delta a_{1,2}$
14*	8 ⟶ 9	$\Delta a_{16,20}, \Delta a_{16,12}, \Delta a_{15,16}, \Delta a_{15,19}, \Delta a_{15,11}, \Delta a_{14,15}, \Delta a_{14,18}, \Delta a_{14,10}, \Delta a_{20,21}, \Delta a_{20,15}, \Delta a_{19,20}, \Delta a_{19,14}, \Delta a_{18,19}, \Delta a_{18,13}, \Delta a_{17,18}, \Delta a_{17,5}, \Delta a_{13,14}, \Delta a_{13,17}, \Delta a_{13,4}, \Delta a_{9,11}, \Delta a_{9,7}, \Delta a_{7,8}, \Delta a_{7,1}, \Delta a_{8,9}, \Delta a_{8,10},$

		$\Delta a_{8,2}$, $\Delta a_{10,11}$, $\Delta a_{10,13}$, $\Delta a_{5,13}$, $\Delta a_{5,6}$, $\Delta a_{4,10}$, $\Delta a_{4,5}$, $\Delta a_{3,8}$, $\Delta a_{3,4}$, $\Delta a_{2,7}$, $\Delta a_{2,3}$, $\Delta a_{1,2}$
15*	9 ⟶ 6	$\Delta a_{11,12}$, $\Delta a_{11,14}$, $\Delta a_{11,8}$, $\Delta a_{16,20}$, $\Delta a_{16,12}$, $\Delta a_{15,16}$, $\Delta a_{15,19}$, $\Delta a_{15,11}$, $\Delta a_{14,15}$, $\Delta a_{14,18}$, $\Delta a_{14,10}$, $\Delta a_{20,21}$, $\Delta a_{20,15}$, $\Delta a_{19,20}$, $\Delta a_{19,14}$, $\Delta a_{18,19}$, $\Delta a_{18,13}$, $\Delta a_{17,18}$, $\Delta a_{17,5}$, $\Delta a_{13,14}$, $\Delta a_{13,17}$, $\Delta a_{13,4}$, $\Delta a_{9,11}$, $\Delta a_{9,7}$, $\Delta a_{7,8}$, $\Delta a_{7,1}$, $\Delta a_{8,9}$, $\Delta a_{8,10}$, $\Delta a_{8,2}$, $\Delta a_{10,11}$, $\Delta a_{10,13}$, $\Delta a_{5,13}$, $\Delta a_{5,6}$, $\Delta a_{4,10}$, $\Delta a_{4,5}$, $\Delta a_{3,8}$, $\Delta a_{3,4}$, $\Delta a_{2,7}$, $\Delta a_{2,3}$, $\Delta a_{1,2}$
16*	6 ⟶ 7	$\Delta a_{12,15}$, $\Delta a_{12,9}$, $\Delta a_{11,12}$, $\Delta a_{11,14}$, $\Delta a_{11,8}$, $\Delta a_{16,20}$, $\Delta a_{16,12}$, $\Delta a_{15,16}$, $\Delta a_{15,19}$, $\Delta a_{15,11}$, $\Delta a_{14,15}$, $\Delta a_{14,18}$, $\Delta a_{14,10}$, $\Delta a_{20,21}$, $\Delta a_{20,15}$, $\Delta a_{19,20}$, $\Delta a_{19,14}$, $\Delta a_{18,19}$, $\Delta a_{18,13}$, $\Delta a_{17,18}$, $\Delta a_{17,5}$, $\Delta a_{13,14}$, $\Delta a_{13,17}$, $\Delta a_{13,4}$, $\Delta a_{9,11}$, $\Delta a_{9,7}$, $\Delta a_{7,8}$, $\Delta a_{7,1}$, $\Delta a_{8,9}$, $\Delta a_{8,10}$, $\Delta a_{8,2}$, $\Delta a_{10,11}$, $\Delta a_{10,13}$, $\Delta a_{5,13}$, $\Delta a_{5,6}$, $\Delta a_{4,10}$, $\Delta a_{4,5}$, $\Delta a_{3,8}$, $\Delta a_{3,4}$, $\Delta a_{2,7}$, $\Delta a_{2,3}$, $\Delta a_{1,2}$
17	7 ⟶ 11	$\Delta a_{21,16}$, $\Delta a_{12,15}$, $\Delta a_{12,9}$, $\Delta a_{11,12}$, $\Delta a_{11,14}$, $\Delta a_{11,8}$, $\Delta a_{16,20}$, $\Delta a_{16,12}$, $\Delta a_{15,16}$, $\Delta a_{15,19}$, $\Delta a_{15,11}$, $\Delta a_{14,15}$, $\Delta a_{14,18}$, $\Delta a_{14,10}$, $\Delta a_{20,21}$, $\Delta a_{20,15}$, $\Delta a_{19,20}$, $\Delta a_{19,14}$, $\Delta a_{18,19}$, $\Delta a_{18,13}$, $\Delta a_{17,18}$, $\Delta a_{17,5}$, $\Delta a_{13,14}$, $\Delta a_{13,17}$, $\Delta a_{13,4}$, $\Delta a_{9,11}$, $\Delta a_{9,7}$, $\Delta a_{7,8}$, $\Delta a_{7,1}$, $\Delta a_{8,9}$, $\Delta a_{8,10}$, $\Delta a_{8,2}$, $\Delta a_{10,11}$, $\Delta a_{10,13}$, $\Delta a_{5,13}$, $\Delta a_{5,6}$, $\Delta a_{4,10}$, $\Delta a_{4,5}$, $\Delta a_{3,8}$, $\Delta a_{3,4}$, $\Delta a_{2,7}$, $\Delta a_{2,3}$, $\Delta a_{1,2}$

The response of $LOLAS$ for the state-space expansion is shown in Figure 4.17. It clearly shows how the size of the domain (as *2D pyramids*) quickly increases with the addition of new probable states as compared to LAS.

The probability bunked during the expansion and approximation of model is shown in Figure 4.18. There were no reversible reactions in the model and all R_M have similar kinetic parameters, so the error becomes a constant from $t = 0.56\ sec$ after losing $1.03e-05$ of probability in the approximation up until t_f.

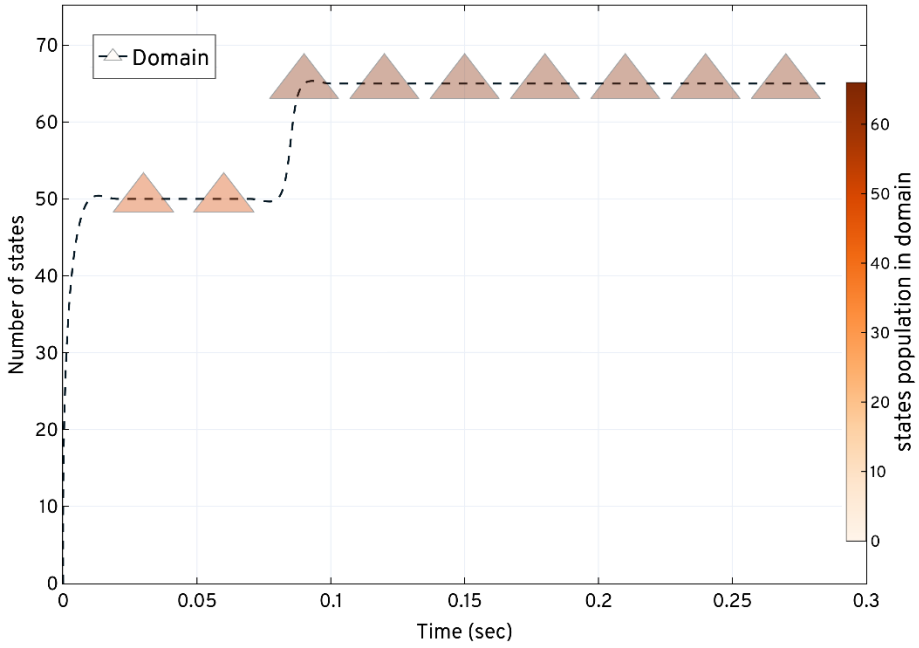


Figure 4.17. Based on *LOLAS* strategy, the response shows how the size of the domain increases with the addition of new states with time in the toy model.

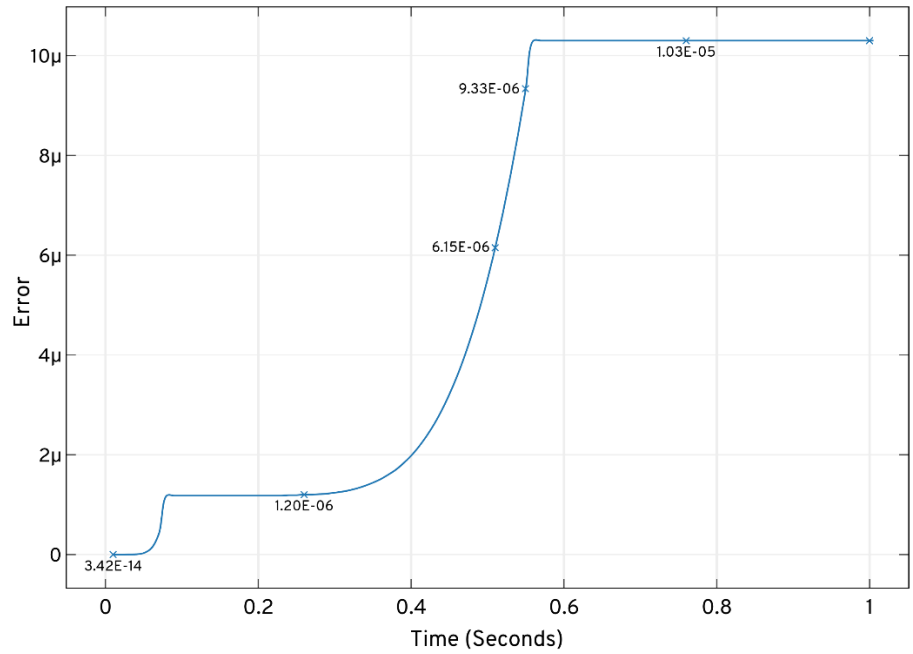


Figure 4.18. Based on *LOLAS* strategy, the response shows the total bunked probability during the approximation of the toy model.

$LOLAS$ successfully creates the domain of an optimum order with 66 states at t_f by introducing new states to the domain with time, as shown in Figure 4.19. The response of $LOLAS$ in Figure 4.19 is different from the LAS response in Figure 4.9, which shows that $LOLAS$ is much faster than LAS in finding states. Such behaviour of $LOLAS$ is more suitable for large models having large state-space. In addition, the ISP state-space patterns in Figure 4.9 and Figure 4.19 can be used as a *blueprint* of the model's state-space to compare it with other model *blueprints* for the characteristics and occurrence of reactions. Such a pattern is considered to predict the behaviour of large network state-space expansion when the set of occurrence of initial reactions are similar in different systems.

The computational expense estimation for exploring this system is given in Table 4.10 for different time points. Overall, the number of states explored by LAS and $LOLAS$; however, the time at which different states are explored changed the bunked probabilities of the states at time points. At $t = 0.05$ *sec*, $LOLAS$ explores 50 states while LAS explores 27 states taking 0.00882 *sec* and 0.02082 *sec* respectively. This shows that even though the number of states explored by both the variants do not change at t_f, the number of states present at the time points may change the solution of the CME, as seen for the system like the toy model.

Table 4.10. LAS expansion of the state-space and solution of the toy model at different time points.

Time (sec)	No. of states explored	Exploration performance time (sec)	Probability lost	Approximation (probability)
0.0	0	0.0	0	0
0.01	50	0.04473	3.42e-14	≈0.99
0.05	50	0.00882	2.89e-08	0.9999999711
0.1	66	0.01854	1.18e-06	0.99999882
1.0	66	0.00210	1.03e-05	0.9999897

Similarly, in the following section 4.4.2, we will apply the $ISP\ LOLAS$ on real biological model of a coupled enzymatic reaction system to study the expansion of state-space and its solution.

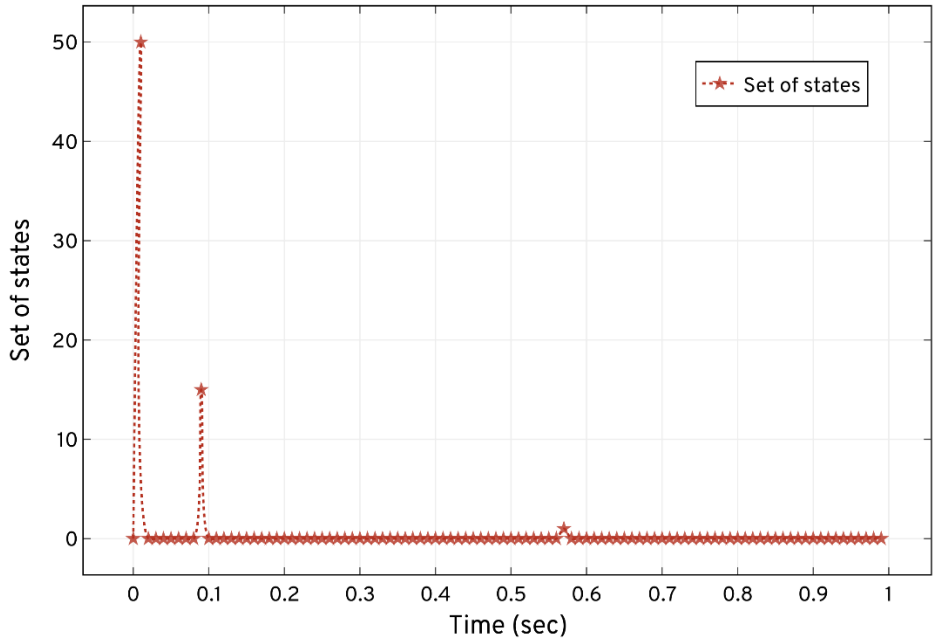


Figure 4.19. Shows the set of states explored for the toy model after every *LOLAS* iteration. Based on the reactions, *LOLAS* unfolds the state-space pattern to update states in the domain and expands 66 probable states in 1.0 *sec*. ★ shows the time point where new set of states is explored and updated in the domain.

4.4.2 Biological Example

Validation of the *ISP LOLAS* on biochemical model – coupled enzymatic reactions network is shown below. To demonstrate the *ISP LOLAS* algorithm, we first consider the model defined by the reactions,

$$S + E_1 \overset{k_1}{\rightarrow} C_1 \overset{k_2}{\rightarrow} S + E_1, \qquad C_1 \overset{k_3}{\rightarrow} P + E_1 \tag{95}$$

$$P + E_2 \overset{k_4}{\rightarrow} C_2 \overset{k_5}{\rightarrow} P + E_2, \qquad C_2 \overset{k_6}{\rightarrow} S + E_2 \tag{96}$$

depicted as a network in Figure 4.11 as:

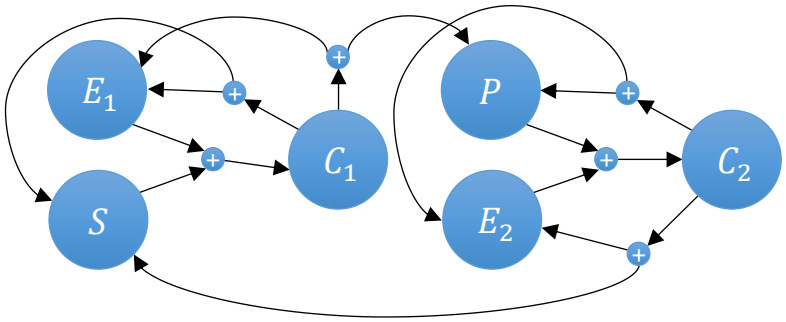


Figure 4.20. Coupled enzymatic reactions network. Showing six $\tilde{N} = 6$ species, S, E_1, C_1, P, E_2, C_2, in a network defining reactions, as given in Eqs. (95) and (96).

This biochemical system (dimension = 6) describes two sets of enzymatic reactions transforming species S into species P and transforming species P back into S. We rewrite this enzymatic reactions system as a network of six reactions:

$$\begin{gathered} \boldsymbol{R_1}\colon\ S + E_1 \overset{k_1}{\rightarrow} C_1, \quad \boldsymbol{R_2}\colon\ C_1 \overset{k_2}{\rightarrow} S + E_1 \\ \boldsymbol{R_3}\colon\ C_1 \overset{k_3}{\rightarrow} P + E_1, \quad \boldsymbol{R_4}\colon\ P + E_2 \overset{k_4}{\rightarrow} C_2 \\ \boldsymbol{R_5}\colon\ C_2 \overset{k_5}{\rightarrow} P + E_2, \quad \boldsymbol{R_6}\colon\ C_2 \overset{k_6}{\rightarrow} S + E_2 \end{gathered} \tag{97}$$

with initial copy counts $S = 50$, $E_1 = 20$, $E_2 = 10$, $C_1 = C_2 = P = 0$ and reaction rate parameters of $k_1 = k_4 = 4$, $k_2 = k_5 = 5$, $k_3 = k_6 = 1$. These species counts are used as a state-space to define the model and these copy counts are tracked as:

$$([S], [E_1], [C_1], [P], [E_2], [C_2]) \in \tilde{N} := (x_0, x_1, x_2, x_3, x_4, x_5).$$

In reaction R_1, the copy count of S and E_1 is reduced by 1, which increases the copy count of C_1 by 1. Reaction R_2 is the backward reaction of R_1; therefore, the copy count of C_1 is reduced by 1 and this increases the copy count of S and E_1 by 1. Then, reaction R_3 again reduces the count of C_1 and increase the counts of P and E_1 by 1. Similarly, in reaction R_4, the copy counts of P and E_2 are reduced by 1, which increases the copy count of C_2 by 1. Reaction R_5 is the backward reaction of R_4; therefore, the copy count of C_2 is reduced by 1 and this increases the copy count of P and E_2 by 1. Then, reaction R_6 again reduces the count of C_2 and increases the counts of S and E_2 by 1. We can now define the transitions associated with $R_1, R_2, R_3, R_4, R_5, R_6$ in stoichiometric vector V_M matrix as:

$$V_M = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix} = \begin{bmatrix} -1 & -1 & 1 & 0 & 0 & 0 \\ 1 & 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & 0 & 1 & 1 & -1 \\ 1 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}. \tag{98}$$

For the $LOLAS$ method for compatibility, the associated Markov chain of this model is converted to a Markov chain tree, and the states in terms of nodes with additional information, such as number of R_M reactions required to reach the state. In growing Markov chain tree, the transition between the nodes:

$$N_i \xrightarrow{v_\mu(X_0(t),X_1(t),\ldots,X_K(t))} N_{i+1}\,, \tag{99}$$

is defined in the typical form of the dictionary $Dict$. We express the propensity functions of the six reactions in terms of the states $([S],[E_1],[C_1],[P],[E_2],[C_2]) \in \tilde{\mathrm{N}}$ as:

$$\boldsymbol{R_1}: \quad a_1([S],[E_1],[C_1],[P],[E_2],[C_2]) = k_1([S][E_1]),$$

$$\boldsymbol{R_2}: \quad a_2([S],[E_1],[C_1],[P],[E_2],[C_2]) = k_2([C_1]),$$

$$\boldsymbol{R_3}: \quad a_3([S],[E_1],[C_1],[P],[E_2],[C_2]) = k_3([C_1]),$$

$$\boldsymbol{R_4}: \quad a_4([S],[E_1],[C_1],[P],[E_2],[C_2]) = k_4([P][E_2]),$$

$$\boldsymbol{R_5}: \quad a_5([S],[E_1],[C_1],[P],[E_2],[C_2]) = k_5([C_2]),$$

$$\boldsymbol{R_6}: \quad a_6([S],[E_1],[C_1],[P],[E_2],[C_2]) = k_6([C_2]),$$

Node $N_1 = (X_0, \mathrm{d}_1)$ carries the initial state X_0 of the system at initial depth level 1. Then $\mathbf{n}_J = (\mathbf{X}_K, \mathrm{d}_{1,2,\ldots\ldots})$ is further expanded and the states updated by following the order of $LOLAS$. The

corresponding propensities $\Delta a_{i,j}$ are updated in the $A_{i,j}$ matrix in every iteration based on the given $LOLAS$ updation trend (see Table 4.9). Initially, the system started with $S = 50$, $E_1 = 20$, $E_2 = 10$ and gradually all reactant species are transformed to products resulting in the system ending in $\mathbf{n}_J = \left(X_{1,2,\ldots\ldots 8296}, \mathfrak{a}_l\right)$.

Figure 4.21 shows the response of the $LOLAS$ method when solved with $\tau_m = 1e-6$ for $t_f = 2.0$ *sec*. Due to the nature of the model reaction rates, small steps $t_{step} = 0.01$ *sec* are taken to capture the moments based on non-negative non-zero states for the domain. $LOLAS$ successfully creates the domain of an optimum order with 8296 states at t_f by introducing the new states to the domain with time, as shown in Figure 4.21.

This pattern also depicts that the frequency (number of states at any time t) of expansion increases in depth when the number of active reactions increases in the system. With the addition of probable states, the domain contains enough probability mass to approximate the solution up to t_f. The states are updated in sets as seen in Figure 4.22, for the catalytic system after every iteration.

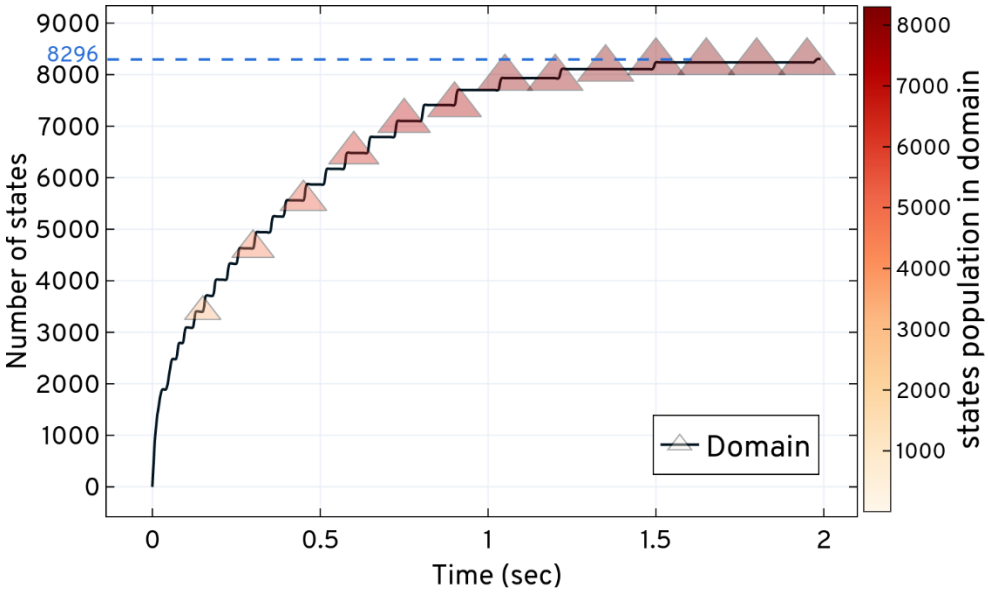


Figure 4.21. Expansion and updation of states for the dual enzymatic reaction network based on the *LOLAS* method. The state-space expansion increases the number of additions of new states in the domain. The size and colour of ▲ shows the increase in size of the domain with the states' population.

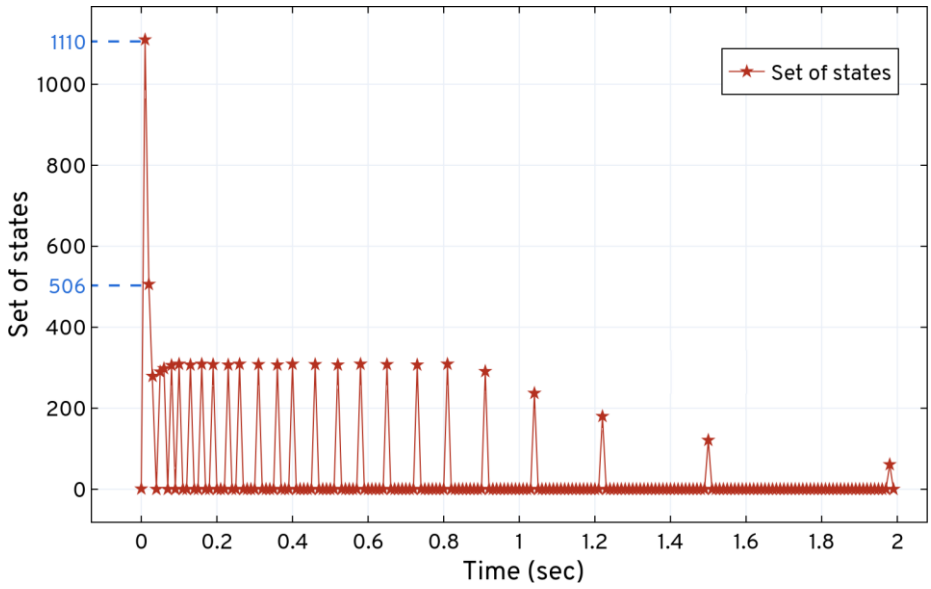


Figure 4.22. Shows the set of states explored for the dual enzymatic reaction network after every *LOLAS* iteration. Based on the network of reactions, *LOLAS* unfolds the state-space pattern to update states in the domain and expands 8296 probable states in 2.0 *sec*. ★ shows the time point where new set of states is explored and updated in the domain.

The state-space pattern in Figure 4.22 can be used as a blueprint of the dual enzymatic reaction network state-space to compare with other model blueprint for their characteristics and the occurrence of reactions. Such a pattern is considered to predict the behaviour of a large network state-space expansion when the set of occurrences of the initial reactions are similar in different systems. The solution of Eq. (9) up to t_f for the domain created by $LOLAS$ is shown in Table 4.6 and the system's conditional probabilities based on species is shown in Figure 4.13.

Table 4.11. $LOLAS$ expansion response and solution at t_f for the dual enzymatic reaction network.

$t_f = 2.0$, $t_{step} = 0.01$	Run-time (sec)	Domain	Expansion time (sec)	Error at t_f
ISP LOLAS	1614.22	8296	2.0	$5.953\text{e}-05$

In three test runs, the run time of $ISP\ LOLAS$ for the dual enzymatic reaction network was ≈1614 *secs* when solving the Eq. (9) with 14666 states. The probability of the species in Figure 4.23 shows the nature of the reactions affecting the counts of each species in the system. At t_f, the probabilities of E_2 and C_2 remain high compared to E_1 and C_1 at different molecular counts, which results in a low probability of P compared to S. We know that this network transforms species S into species P and then transforms the species P back into S. Therefore, based on the current probabilities of the species at t_f, the future probability of P will increase and, for S, it will remain same or decrease. With this change, the probabilities of E_2 and C_2 decreased compared to E_1 and C_1.

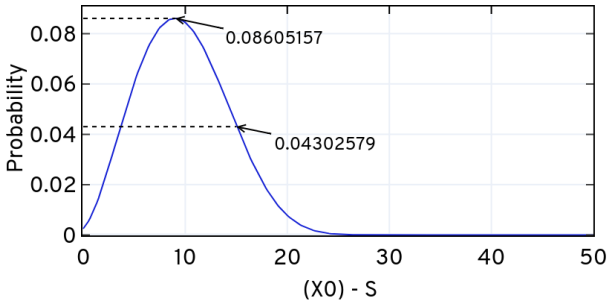


Fig (A). Probability of S over t_f

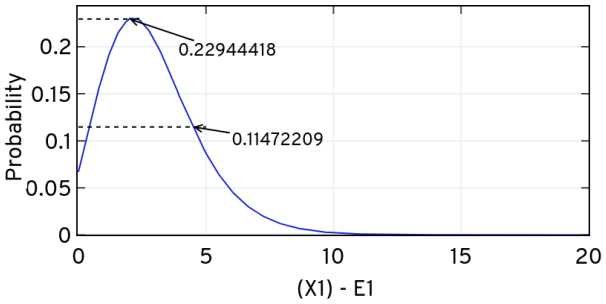


Fig (B). Probability of E_1 over t_f

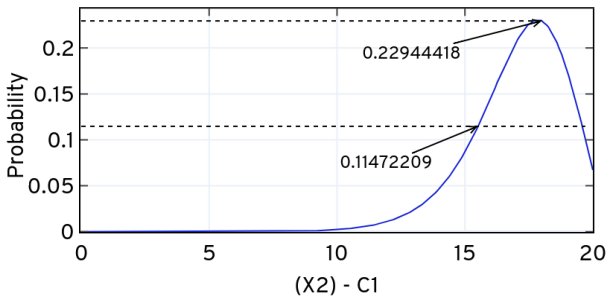


Fig (C). Probability of C_1 over t_f

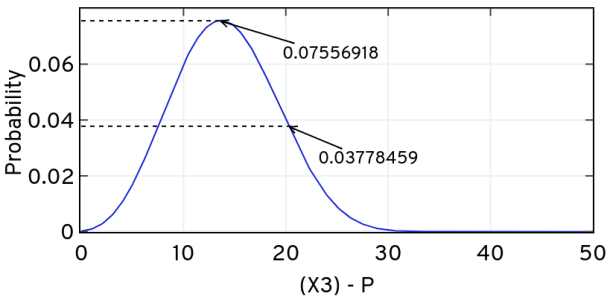


Fig (D). Probability of P over t_f

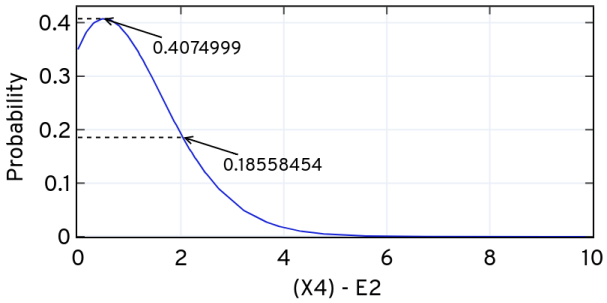


Fig (E). Probability of E_2 over t_f

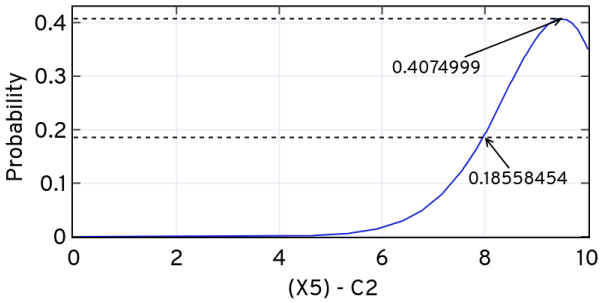


Fig (F). Probability of C_2 over t_f

Figure 4.23. Conditional probability of the dual enzymatic reactions system evaluated at $t_f = 2.0$ *sec*, $t_{step} = 0.01$ using *LOLAS*. Fig (A) is the probability of the species S over t_f, Fig (B) is the probability of the species B over t_f, Fig (C) is the probability of the species C over t_f, Fig (D) is the probability of the species P over t_f, Fig (E) is the probability of the species E over t_f.

Figure 4.24 shows the total probability bunked at t' while progressing with the expansion. The bunking produces an error (w.r.t approximation) with time when the number of states increases with the expansion and provided that, $LOLAS$ produces a minimal error of order, 10^{-5}, as given in Table 4.11.

A comparative study of $LOLAS$ with other methods, such as SSA and *r-step reachability* used in SW and $OFSP$, respectively, is discussed further in Chapter 5.

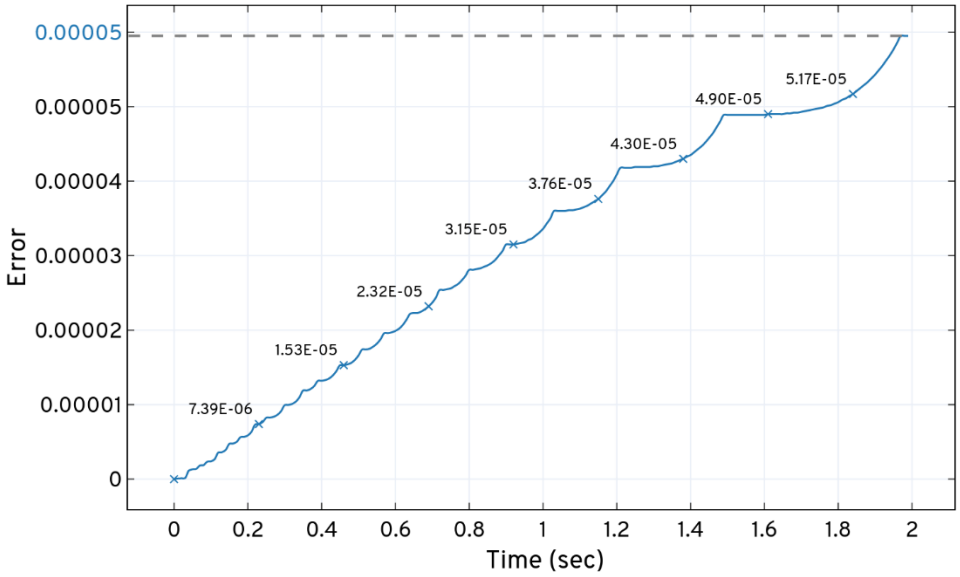


Figure 4.24. Total probability of states bunked at t' from the domain produced by dual enzymatic reactions system in *ISP LOLAS* iteration while expansion and solving the CME

4.5 Data Structure Complexity of Operations

As discussed in section 3.6, one of the effective ways to understand the complexity of the algorithm is to calculate the *tight bound* (average case) and *upper bound* (worst case) cases of the operations. General routines of the operation of ISP elements are given in Figure 4.25 which define the complexity of the operations independently.

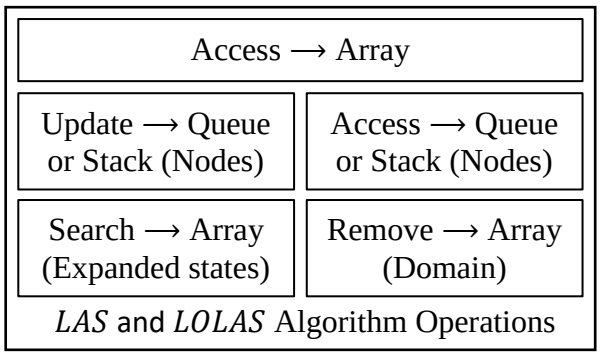


Figure 4.25. Routines of operation of ISP modules. The worst-case complexity is considered for LAS and $LOLAS$ algorithm for the upper limit of the execution.

A comparison of the $ISP\ LAS$ and $ISP\ LOLAS$ average-case Θ and worst-case O *Time-Space complexity* based on operations (*access, search, update, and remove*) on the elements (*array, queue, and tree*) is given in Appendix E.

The data operations performed in *steps* of LAS (Table 4.2) and $LOLAS$ (Table 4.7) have different complexities, as given in Tables E.1 and E.2 of Appendix E, respectively. For LAS and $LOLAS$, the worst-case O complexity of a tree is considered as we visualise the Markov chain of biochemical systems through graphs, whereas the individual complexities are considered for operations in algorithm steps. For LAS, $O\big(Algo(y)\big)$ the *Time* and *Space* complexity is $O(\mathrm{T}^{\mathrm{d}_l+1})$, as seen in Figure 4.8 when showing linearity. For $LOLAS$, $O\big(Algo(y)\big)$ the time and space complexity is $O(\mathrm{T}^{\mathrm{d}_l})$ and $O(\mathrm{T}\mathrm{d}_l)$, respectively.

4.6 Discussion and Conclusion

In this chapter, we first derived the expansion criterion and break-off point in the expansion of state-space by using a vector form (see Eq. (7)) of the CME. These criterion changes (as given by Eqs. (72), (80), (81), (82)) when *fast* and *slow* reactions are considered for any model for state-space expansion. Based on these conditions, we introduced an analytical method called the *Intelligent State Projection* (ISP) that integrates the reactions propensity, parameters describing the Markovian processes through nodes that govern the states. The purpose of using ISP was to understand and predict the dynamics of the state-space response in biochemical systems for the solution of the CME.

The ISP has two sub-routines; namely – LAS (see section 4.3) and $LOLAS$ (see section 4.4). They treat the Markov chain of a biochemical system as a Markov chain tree structure and state as objects of the class *node*. The search was performed in a latitudinal way using the $ISP\ LAS$ method. Whereas, a bidirectional (*Longitudinal-Latitudinal Search*) search was applied using $ISP\ LOLAS$, which quickly expanded the state-space up to a specified bound limit. To support the expansion strategy, the $BLNP$ function of Eq. (56) was joined with ISP variants to follow the events (firing of reactions) at any interval at the molecular population level (through propensities). $BLNP$ provided confidence to the expansion strategy by considering the weighted probability values on the occurrence of future reactions and by prioritising the direction of expansion. Both variants of the ISP embedding node projection inductively expanded the multiple states with time. They also defined the complexity of the system by predicting the pattern (called a *blueprint*) of the state-space updation and the formation of bounds at different time points.

Based on the complexity of the data structure, the $ISP\ LAS$ speed (rate of searching the states per time step) of expansion was low compared to $ISP\ LOLAS$; however, the computational time depends on the nature of the model and size of the time step. At any point, the amount of memory in use was directly proportional to the neighboring states reachable through a single R_M reaction. In addition, the $ISP\ LOLAS$ retraction to the initial node to track firing of new reaction was followed by revisiting the depth many times did not affect the computational time. The results of our application of ISP on toy models showed that both LAS and $LOLAS$ have different patterns of expansion but the number of states in the domain were same at t_f. However, in real biological models the number of states in a domain may not be same at t_f for LAS and $LOLAS$ and we have discussed this in Chapter 5 through application of LAS and $LOLAS$ on the same biological reaction system.

The application on real biological models in sections 4.3.2 and 4.4.2 highlight the effectiveness of the ISP and the differences between two variants. Upon implementation, the performance of the ISP method was compared with other efficient methods discussed in the literature. Therefore, in Chapter 5, we have compared the ISP variants with the *r-step rechability* (used in $OFSP$) and SSA (used in SW) by applying it to biological models to ascertain the number of states in the domain at t_f, the computational time and the accuracy of the solution.

Chapter 5
Comparative Study And Analysis

In this chapter, we investigate and compare the performance of the numerical $ISP\ LAS$, $ISP\ LOLAS, OFSP, SSA$ expansion methods based on biochemical reaction networks – the catalytic reaction network (as discussed in section 4.3.2) and dual enzymatic reaction network (as discussed in section 4.4.2). We compared ISP with the *r-step reachability* strategy used by several methods (see section 2.3), such as FSP, $OFSP$, and the SSA used by SW for the expansion of the state-space on the basis of computational time, domain size and the accuracy of the solution at t_f. The ISP harnesses the functions that are building blocks of the fast numerical computation used in Intel® and AMD® microprocessors. They also use the advanced vector extensions, universal functions, compilers and the vector math library to provide benefit in making the routines efficient. The dependencies and workloads employed in the performance test were implemented on Amazon® Web Services with distribution by the Python library (see Appendix F**).**

In this chapter, in section 5.1, we first give an overview of the comparative study that we will conduct in following sections. Sections 5.2 and 5.3, present a comparison of ISP with other methods based on biological models. In section 5.4, we will give a brief discussion and the chapter's conclusions.

5.1 Study Overview

One is generally interested in the evolution of the probability density over time (as seen in Figure 4.14, Figure 4.24) and this requires the solution of Eq. (9) at t_f (as seen in Table 4.6, Table 4.11). To compare the solution of the $ISP\ LAS$, $ISP\ LOLAS$, $OFSP$, SSA methods, we computed the biochemical models:

(a) The catalytic reaction network up to $t_f = 0.5$ *sec* with the initial condition point density at (50, 0, 0, 80, 0), and the propensities of the reactions, as given in Eq. (89), (90), (91) and the network as in Figure 4.10. The time step for the ISP and $OFSP$ methods was set to $t_{step} = 0.01$, and the compression step for $OFSP$ was set to 10.

(b) The dual enzymatic reaction network of up to $t_f = 2.0$ *sec* with the initial condition point density at (50, 20, 0, 0, 10, 0), and the propensities of the reactions as given in Eq. (97)

and the network, as in Figure 4.11. The time step for the ISP and $OFSP$ methods was set to $t_{step} = 0.01$, and the compression step for the $OFSP$ was set to 1.

We performed these experiments on the carbon-neutral platform of Amazon® Web Service Elastic Computing (EC2) instance type large (m5a) running on HVM (hardware virtual environment) virtualisation with variable ECUs, multicore environment 16vCPU @ 2.2GHz, AMD EPYC 7571 running Ubuntu 16.04.1 with the relevant dependencies, 64GB memory with 8GB Elastic Block Storage (EBS) type General Purpose SSD (GP2) formatted with Elastic File System (EFS). The performance mode was set to General Purpose with input-outputs per second (IOPS = 100/3000) and the throughput mode of type bursting was set. (also see Appendix F and Appendix G).

5.2 Comparison Based on Catalytic Reaction System

An approximation of 10^{-5} is considered to find the approximate number of realisations required by the SSA for 10^{-4} global error. Realisations were computed until the difference is less than 10^{-4} between the known distribution and the empirical distribution. Therefore, approximately 10^6 runs were required to obtain the right distribution for this model. In Table 5.1, we observe that both versions of ISP are faster than the $OFSP$ of *r-step reachability* and the SSA of sliding windows, and the improvement was attributed to the $LOLAS$ having fewer states and less computational time than the $OFSP$ method and with better accuracy at t_f. Similary, the ISP was much faster than the SSA as the total number of realisations required to have an empirical distribution while the error at t_f was ≈10 times more than the domain produced by the ISP.

Table 5.1. Comparison of the solution of the catalytic reaction system based on ISP, $OFSP$ and SSA.

$t_f = 0.5$, $t_{step} = 0.01$	ISP		$OFSP$	SSA
	LAS	$LOLAS$		
Run-time (sec)	4677	2706	8767	17428
Domain at t_f	14666	13089	14665	10^6 Runs
Expansion time	0.5	0.5	0.5	-
Error at t_f	$1.865e-05$	$1.532e-05$	$1.917e-05$	$\approx 9.81 \times 10^{-3}$

We also compared the error at t_f to note the efficiency of the solution. As seen in the results the increase in step error in $OFSP$ affected the solution at t_f. Figure 5.1, shows a comparison

of the ISP (LAS and $LOLAS$) with $OFSP$ on the basis of the approximation error at t during the expansion of the catalytic reaction system. Addressing the step error in ISP and the selection of the probable states results in an efficient solution at t_f as compared to $OFSP$.

The typical firing nature of reactions in catalytic system makes them stiff and; therefore, the selection of states becomes difficult for approximation. This is due to some species in the system tending to increase abruptly in the population while others do so very slowly because the kinetic parameters ($k_1 = 1$, $k_2 = 1000$, $k_3 = 100$) have large differences and this triggers the reactions at different rates. Reaction R_1, is categorised as a *slow* reaction in the network and that affects the fast reaction, R_2. In the computation results of Table 5.1, the ISP identified that only 13089 probable states were required to solve the system up to t_f, this saves computational time (see Figure 5.2) compared to $OFSP$ and SSA, as well as improves the accuracy of the solution. In $OFSP$, applying the compression at every step or after a few steps is still computationally expensive for small models (like the catalytic reaction system), as seen in Table 5.1 and Figure 5.1. This also leads to the justification that the total computation effort required at every step when compressing the number of states up to t_f is approximately equal to the total computation effort required when the compression is applied in the gaps in some steps on a set of states up to t_f. Moreover, the state-space will remain the same, at t_f regardless of when the compression is applied.

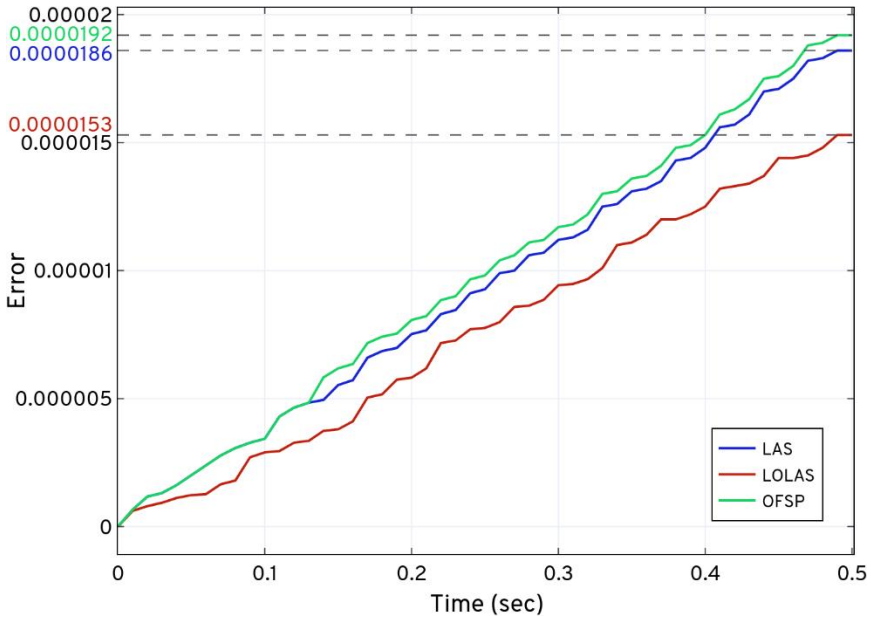


Figure 5.1. The comparison of *ISP* (*LAS* and *LOLAS*) with *OFSP* based on the solution of the catalytic reaction system.

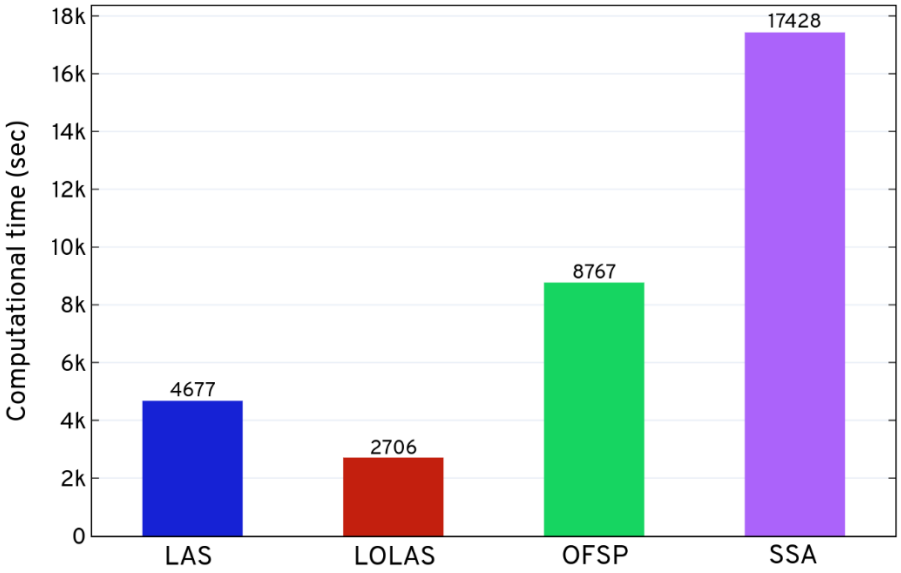


Figure 5.2. The comparison of *ISP* (*LAS* and *LOLAS*) with *OFSP* and *SSA* by computational time. All methods were applied to the catalytic reaction system that was previously integrated in section 4.3.2.

A comparison of the computational times in Table 5.1 shows that both versions of ISP are significantly faster than the other methods. Figure 5.3 shows the CPU utilisation (%) of $LOLAS$ and $OFSP$ with respect to run-time (minutes). The dedicated throughput (see Appendix F) between EC2 and EBS was not used for solving the model. The average CPU exertion is about 60%, which is a considerable workload for such a configuration for a given model. The expansion and approximation, started when CPU use was at ≈1.6422% at $t = 0$ *sec* and increases up to 60.0% and then goes down to zero at t_f.

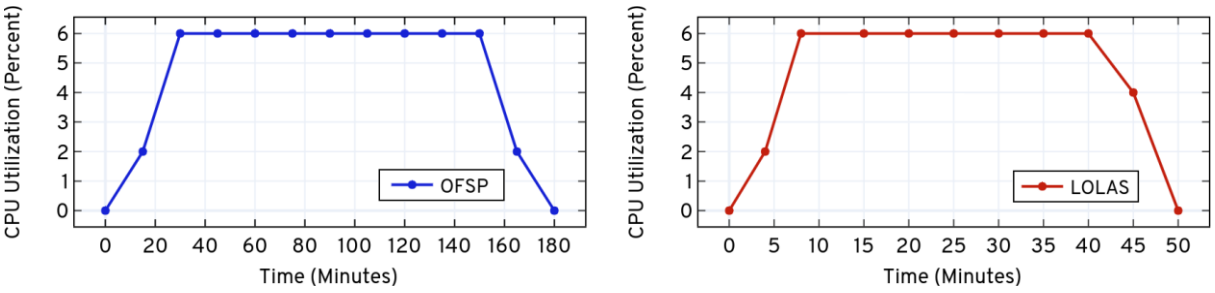


Figure 5.3. AWS® CPU utilisation percentage, when the catalytic reaction system is solved up to $t_f = 0.5$ *sec* using $OFSP$ and $LOLAS$. The performance analysis was carried out using CloudWatch® (Statistic: Average, Time Range: Hour, Period: 5 Minutes).

5.3 Comparsion Based on the Dual Enzymatic Reaction Network

An approximation of 10^{-5} is considered to find the approximate number of realisations required by the SSA for a 10^{-4} global error. Realisations were computed until the difference was less than 10^{-4} between the known distribution and the empirical distribution. Therefore, approximately 10^5 runs were required to obtain the correct distribution for this model. In Table 5.2, we observe that both versions of ISP are faster than $OFSP$ of *r-step reachability* and the SSA of sliding windows; the improvement is attributed to both the variants of ISP having an efficient domain with a small approximation error and less computational time than that of $OFSP$ method and with better accuracy at t_f. Similarly, both variants of ISP were much faster than SSA as the total number of realisations required to have an empirical distribution with the error at t_f is ≈12 times more than the domain produced by ISP.

Table 5.2. Comparison of the solutions of the dual enzymatic reaction system based on ISP, $OFSP$ and SSA

$t_f = 2.0$, $t_{step} = 0.01$	ISP		$OFSP$	SSA
	LAS	$LOLAS$		
Run-time (sec)	2386	1614	2804	6374
Domain at t_f	8282	8296	8266	10^5 Runs
Expansion time	2.0	2.0	2.0	-
Error at t_f	$7.470\text{e}-05$	$5.953\text{e}-05$	$1.060\text{e}-04$	$\approx 9.94 \text{ x } 10^{-3}$

We also compared the error at t_f to note the efficiency of the solution. As seen in the results, the increase in the step error in $OFSP$ affects the solution at t_f. Figure 5.4, shows the comparison of the ISP (LAS and $LOLAS$) with $OFSP$ on the basis of the approximation error at t during the expansion of the dual enzymatic reaction system. Addressing the step error in ISP and the selection of the probable states results in an efficient solution at t_f compared to $OFSP$.

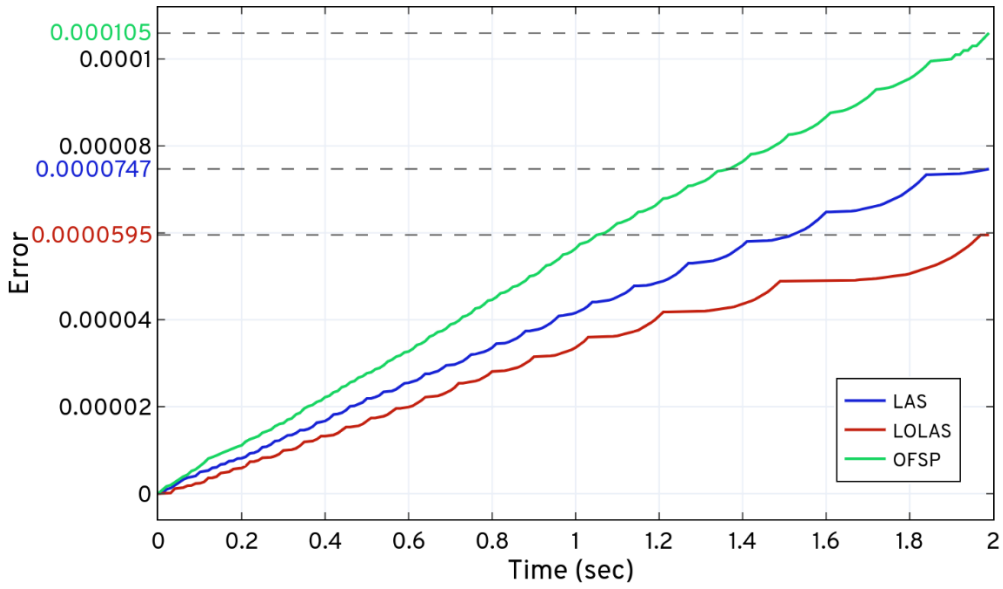


Figure 5.4. Comparison of the *ISP* (*LAS* and *LOLAS*) with *OFSP* based on the solution of the dual enzymatic reaction system.

The network consists of two enzymatic reaction system interlinked that transforms species S and P into each other via the other species, making the system stiff in nature and; therefore, the selection of states becomes difficult for approximation. This is due to some species (S and E_1) in the system tending to increase in population abruptly while others very slowly, because some of the kinetic parameters ($k_1 = k_4 = 4$, $k_2 = k_5 = 5$) have large differences from other kinetic parameters ($k_3 = k_6 = 1$) and this triggers the reactions at different rates.

Reaction R_1, categorised as the fastest reaction of the network that affects species S, C_1 and E_1 is followed by other reactions involving other species. From the computation results in Table 5.2, the $ISP\ LAS$ identified that only 8282 probable states and $ISP\ LOLAS$ identified that only 8296 probable states are required to solve the system up to t_f which saves the computational time (see Figure 5.5), compared to $OFSP$ and SSA, as well as improves the accuracy of the solution. In $OFSP$, applying the compression at a defined step or after a few steps is still computationally expensive for models like the dual enzymatic reaction system, as seen in Table 5.2 and Figure 5.4. This also leads to the justification that the total computation effort required at each step in compressing the number of states up to t_f is approximately equal to the total computation effort required when compression is applied in the gaps between some of the steps on the set of states up to t_f. Moreover, the state-space will remain the same at t_f regardless of when the compression is applied.

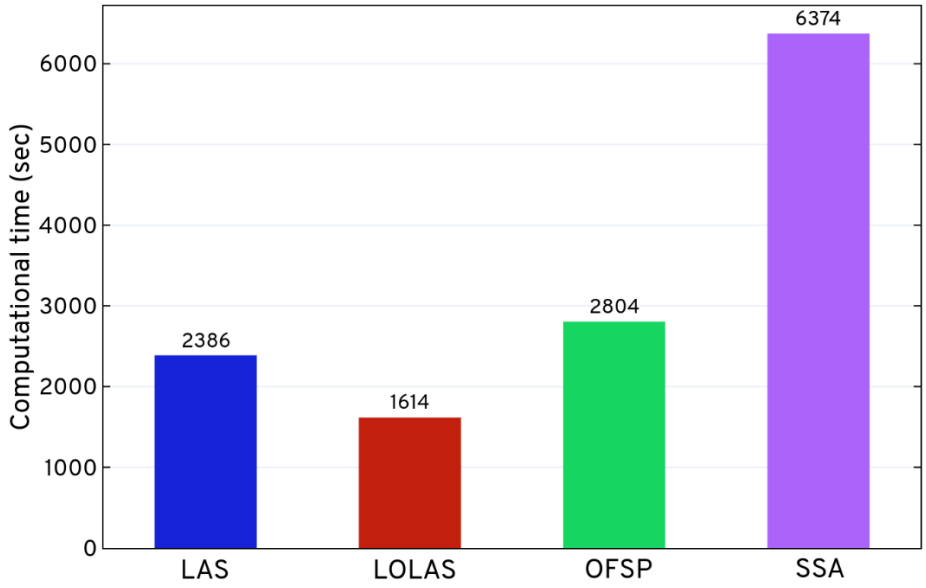


Figure 5.5. The comparison of *ISP* (*LAS* and *LOLAS*) with *OFSP* and *SSA* by computational time. All the methods were applied to the dual enzymatic reaction system previously integrated in section 4.4.2.

Comparison of the computational times in Table 5.2 shows that both versions of ISP were significantly faster than the other methods. Figure 5.3 shows the CPU utilisation (%) of $LOLAS$ and $OFSP$ with respect to run-time (minutes). The dedicated throughput (see Appendix F) between EC2 and EBS has not been used for solving the model. The average CPU exertion is about 60%, which is a considerable workload for such a configuration for the given model. The expansion and approximation started when CPU use was at ≈1.23% at $t = 0$ *sec* and increased up to 60.0% before going down to zero at t_f.

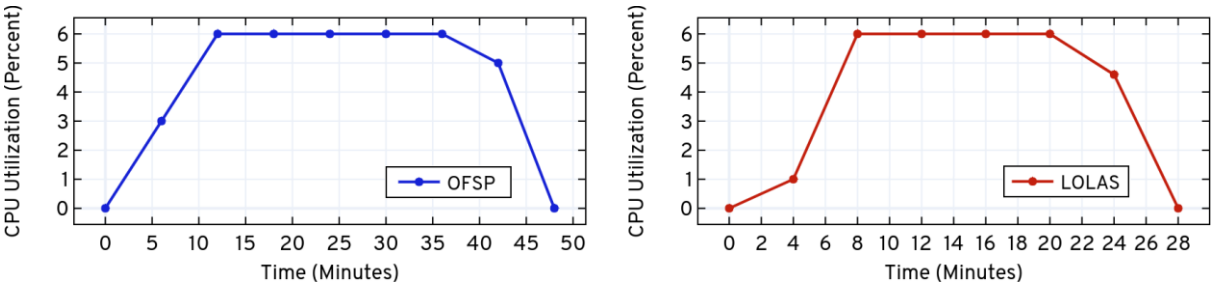


Figure 5.6. AWS® CPU utilisation percentage, when dual enzymatic reaction system is solved up to $t_f = 2.0$ *sec* using $OFSP$ and $LOLAS$. The performance analysis is done using CloudWatch® (Statistic: Average, Time Range: Hour, Period: 5 Minutes).

5.4 Discussion and Conclusion

In this chapter, we have compared $ISP\ LAS$, $ISP\ LOLAS$ with $OFSP$ and SSA based on the number of states present in the domain at t_f, error at t_f and their computational times (run-time). All the methods were applied to the biological model of a catalytic reaction system (in section 5.1), and the dual enzymatic reaction system (in section 5.3), to expand the state-space and solve the CME up to the desired t_f and to check the precision of the domain formed. During method settings, we tested their ability to reproduce the model to measure the dynamics of the system's key components, as discussed in section 4.3.2 and 4.4.2. We found that the ISP method is a novel easy-to-integrate and use technique to model and expand the state-space of biochemical systems, which also show several improvements in modelling and computational efficiency.

The results of the comparative study in sections 5.1 and 5.3 leads to important conclusions based on the performance and accuracy of the ISP methods compared to the other methods. The results in Table 5.1 show that the $ISP\ LAS$ is much faster than $OFSP$ and SSA. Although, the number of states (14666 states) present in the domain created by $ISP\ LAS$ is almost the same as that of the $OFSP$ domain (14665 states); however, the accuracy shows that the domain created by $ISP\ LAS$ has more probable states compared to the domain created by $OFSP$ and the number of SSA realisations. Similarly, $ISP\ LOLAS$ is much faster and the domain (13089 states) produced by $ISP\ LOLAS$ is the smallest compared to the $OFSP$ domain and the number of SSA realisations.

The results in Table 5.2 also show that the $ISP\ LAS$ is faster than $OFSP$ and SSA. Although, the number of states (8282 states) present in the domain created by $ISP\ LAS$ is more similar to that of the $OFSP$ domain (8266 states); however, the accuracy shows that the domain created by $ISP\ LAS$ has more probable states compared to the domain created by $OFSP$ and the number of realisations of SSA. Similarly, $ISP\ LOLAS$ is much faster and has better accuracy than other methods.

The domain formed in the initial steps acts as training information for the ISP methods and, gradually, when the training improves with the addition of new states in the domain and state-space blueprint, it enchances the solution by providing better accuracy with time while keeping the domain (with most probable states) of optimal order as possible. The experiments discussed here show the improvement in the expansion methods through ISP and how the performance of it can be analysed. The computational superiority of both the ISP techniques

over methods that use *r-step reachability* and SSA for expansion of state-space, is evident. The dynamic nature of biological systems and their domain size demonstrates the importance of chosing the probable states for improving the solution of the CME.

Although, $OFSP$ successfully creates the optimum support for some experiments and makes the problem smaller to compute; however, the selection of states and truncation (compression of domain) of state-space in every few set of steps (compression after every 10 steps is chosen in the catalytic reaction system) or for every step (compression in every step is chosen in the dual enzymatic reaction system) does not ensure that the domain created will have the most probable states and nor does it guarantee the computational efficiency, as seen in sections 5.1 and section 5.3 in the results. This is an important guideline that states the selection for domain should be strategic and adaptive to achieve better accuracy in the solution for the CME. For replication of this experiment, refer to G.3 in Appendix G.

Chapter 6

Case study 1 - G1/S checkpoint involving the DNA-damage signal transduction pathway

The main objective of this chapter is to investigate the performance of the *ISP LOLAS* algorithm on a large biochemical system. We analyse the state-space and probabilities of the species by implementing and integrating the mathematical model of the G1/S cell-cycle checkpoint involving the DNA-damage signal transduction pathway. The mathematical model is based on chemical kinetic principles for simulating pathways to show the dynamic behaviour of a cell cycle checkpoint pathway having reactive components. The cell cycle checkpoint pathways involve interactions between different enzymes and proteins in linked reactions. Most models developed for cell cycle checkpoints are quantitative, notably the G1/S checkpoint, which involves interactions between different proteins. They provide important information about the internal mechanisms and complex behaviour of cell cycle checkpoints.

In this chapter, we will briefly introduce the robustness of critical proteins of G1/S checkpoint involving the DNA-damage signal transduction pathway model in section 6.1. In section 6.2, we will prepare and integrate the model to our *ISP LOLAS* algorithm and, in section 6.3, we analyse the state-space of the model and the performance of the *ISP LOLAS* algorithm using computational experiments. In section 6.4, we provide a brief discussion and summary of the study and its results.

6.1 Introduction

In the biological system of G1/S checkpoint transitions, maintaining the stability of the cell cycle is governed by the DNA-damage signal transduction pathway (Iwamoto et al., 2008; Ling et al., 2010). Therefore, interpreting the association between the G1/S checkpoint and DNA-damage signal transduction is a important issue that affects support for life sciences (H. Ling et al., 2010). It is complex to understand the network of the G1/S checkpoint with DNA-damage signal only by using experiments (*in vitro*); therefore, studying a mechanistic mathematical model of the network and performing computational simulations for the different peturbation levels using the chemical kinetics of the model is followed by an analysis of the results.

When the DNA-damage signal transduction and cell cycle checkpoint pathways manage the cell cycle, the genomic DNA (Dasika et al., 1999) components are directly or indirectly damaged by various endogenous and exogenous elements (Ling et al., 2011). DNA damage occurs when regular cells undergo stress (physical or chemical) and, as soon as the DNA damage signal transduction pathway senses this it activates a "damage" signal to the cell cycle. The cell cycle checkpoints provide adequate time for the DNA repair process by blocking the progression of the cell cycle (Geva-Zatorsky et al., 2006; Iwamoto et al., 2008). The complete process is divided into five steps: (1) Mdm2 cannot supervise the stability of protein p53 strictly, so the DNA-damage signal propagates p53 protein; (2) p53 advocates Mdm2 and p21 activity sequentially; (3) p21, a CDK [CDK4/6, CDK2] (cyclin-dependent kinase) inhibitor binds to Cyc/CDK (cyclin/cyclin-dependent kinase), which controls cell cycle progression and has a vital role in the cell cycle checkpoint pathway; (4) cell cycle progression is blocked when complex p21/Cyc/CDK restrains the activity of Cyc/CDK; and (5) the DNA-damage repair pathway can also be activated by p53 triggering its cellular mechanisms (Ling et al. 2011). The cells' abnormal and faulty genomes pass through a progression of cell cycles to future cellular generations when the cell cycle checkpoint loses the capability to detect the DNA-damage signal. This inability makes the genome unstable and normal cells prone to malicious modifications, which result in invasive diseases like cancers.

Cell cycle progression is controlled by distinct mechanisms of different cell cycle checkpoints; for example, G1/S, G2/M and the $G0^5$/G1 phase transition. The cell moves to the G1 phase after staying in the G0 phase for a long time where it first gathers growth factors, nutrients and then communicates the signal (proliferation) for the development of the cell (Iliakis et al. 2003). Once they sense the signal, cells enter the G1 phase and initiate the process of DNA replication. Losing the capability to detect the damage-signal and the interruption of the G1/S checkpoint leads to further problems in cell cycle progression. In the absence of the DNA-damage signal, the transition begins through E2F (Helin, 1998) and that contributes to the production of vital proteins that enable DNA replication (Dyson, 1998; Ikeda et al., 1996; Leone et al., 1999). In the case of DNA damage, this activity ceases until the damage is repaired. In following sub-sections we discuss more about DNA-damage signals.

6.1.1 What Happens in Normal Conditions?

When there is no signal for DNA damage, cells remain in the G0 phase for a long time, as mentioned earlier in section 6.1. When a cell enters the G1 phase it senses the signal (proliferation) that activates CycD synthesis and increases its concentration (Ling et al., 2010). The CycD produced forms the complexes CycD/CDK6, CycD/CDK4 by combining with CDK6 or CDK4, which keeps them activated. According to a study discussed in (Hong Ling, 2011), CycD/CDK6 and CycD/CDK4 have similar physiological functions; therefore, CycD/CDK4/CDK6 represents these two complex proteins. One of the task of CycD/CDK4/CDK6 is to instigate Rb phosphorylation (related to E2F) to obtain Rb-PP/E2F (hypophosphorylated form) and another task is to keep CycE/CDK2 activated. Further, hypophosphorylation of Rb-PP/E2F is carried out by activated CycE/CDK2 releasing E2F by the dissociation of Rb-PPPP and E2F. The excessive level of free E2F activates CycE synthesis in the G1 phase, which promotes the association of CycE and CDK2 to form the CycE/CDK2 complex. This task of E2F then forms a positive feedback loop between CycE and E2F, and this allows the cell to enter the S phase transition (Hiebert et al., 1992).

In the mid-phase of G1, CycD degrades with the release of p27 bound to CycD/CDK4/6 and p27 is then further passed to new complexes (CycA/CDK2 or CycE/CDK2). While p27 hampers the activity of these new complexes, the increased numbers of CycE/CDK2 can trigger the degradation of p27 by phosphorylation (p27 binds to CycE/CDK2) (Coqueret, 2003). This collection of CycE/CDK2 in cells results in p27 degradation. At the G1/S transition, CycA expression is supported by E2F with a noteworthy increase in the S phase. CycA is the principal protein in transition through the S phase and DNA replication. Once CycA is sythesised, it combines with CDK2 to form the CycA/CDK2 complex in the S phase. This CycA/CDK2 complex manages the negative feedback loop to phosphorylate E2F for degradation and also hampers its activities. When the cell settles in the G1/S transition progression, E2F recombines with Rb when Rb-PPPP is dephosphorylated to restrict the activity of E2F in the S phase and maintain the inactive form of Rb/E2F.

6.1.2 What Happens in the Presence of a DNA-damage signal?

In this section, we see how cells regulate in the presence of the DNA-damage signal. p53 is responsible for preserving cell genome fidelity and has an important part in response to the DNA-damage signal (as mentioned in section 6.1). In the absence of the DNA-damage signal, p53 remains at low concentrations due to a negative feedback loop with Mdm2, which keeps p53 in a stable steady-state where its activity is restrained and this improves its degradation rate (Barak et al., 1993; Kubbutat et al., 1997). DNA-damage occurs (Ling et al., 2010) when this negative feedback loop fails to strictly control the p53 level. In such a situation, p53 accumulation leads the nucleus to hamper the cell cycle progression, where DNA damage repair is impossible. The activity of p53 in the cell cycle (Geva-Zatorsky et al., 2006; Lahav et al., 2004; Li & Ho, 1998; Hong Ling, 2011) is different depending on level of the DNA_damage. The DNA-damage signal (low) supports p53 activation and p53 supports p21 sythesis and transcription. When the level of p21 increases, it forms two complexes, p21/CycA/CDK2-P and p21/CycE/CDK2-P, that then combine with CycA/CDK2-P and CycE/CDK2-P to impede their activation. The loss of both these complexes impedes Rb-PP/E2F hypophosphorylation and E2F release and, as a result, this initiates the synthesis of CycA and CycE, which are necessary for progression to the S phase (Ling et al., 2010; Hong Ling, 2011). This step temporarily suspends cell cycle progression and initiates the cell cycle repair pathway to support cells to repair the DNA damage. As soon as the repair is complete, p53 returns to a low level and the negative feedback loop between Mdm2 and p53 is re-established. The level of p21 is reduced with the decrease in p53 and the cell cycle returns to its normal condition when the CycA/CDK2-P and CycE/CDK2-P complexes are released. When the DNA damage is high, the apoptosis pathway is triggered when the DNA-damage signal activates p53.

In following section 6.2, we will combine the situations discussed in section 6.1.1 and section 6.1.2 to obtain the complex structure of G1/S checkpoint that embeds the DNA-damage signal transduction pathway to create the compatible model for the ISP algorithm. We also discuss the modelling and integration of the model for $ISP\ LOLAS$ (as discussed in section 4.4), as well as the hypothesis and assumptions underlying this method.

6.2 Model Integration

To demonstrate the numerical $ISP\ LOLAS$ algorithm on a large model, we consider the G1/S checkpoint involving the DNA-damage signal transduction pathway, as discussed section 6.1 concentrate on the dynamic behaviour of the model. The system is a network of 28 species (see Table 6.1) reacting based on linked reactions (see Table 6.2) with 75 kinetic parameters ($k_{M=75}$) (see Table 6.3). These parameters and 28 mass balance ODEs charaterise the robustness and dynamic behaviour of the network.

The ISP method is initialised and parameterised using initial conditions (given in Table 6.1, Table 6.2, Table 6.3) of the model and integrated as a function. Due to large number of maths operations and equations, simultaneous parameter predictions with limited numbers of experimental values at any instance are often complicated for dynamic systems. Therefore, consistency with the available experimental data and assumptions was ensured at each step of $ISP\ LOLAS$, as this method has led to the successful development of several functions that integrate large numbers of processes supporting the extensive expansion of the state-space.

Considering all the protein species denoted by $\tilde{\mathrm{N}} = x_0, x_1, \ldots\ldots\ldots x_{26}, x_{27}$ then, for $ISP\ LOLAS$, the network (Iwamoto et al., 2008; Ling et al., 2010; Hong Ling, 2011) was redesigned using model variable as Figure 6.1.

The presence of agitation in the DNA damage signal and kinetic parameters affects various critical proteins species, as mentioned in section 6.1. Therefore, we will track the most critical proteins (see Table 6.1) of this network. The dynamic behaviour of the system depends on how these proteins and their probabilities change with time and increase the domain size over time as that will give us information about the robustness of the network. It is sufficient to track the number of copies of the proteins in order, as given in Table 6.1.

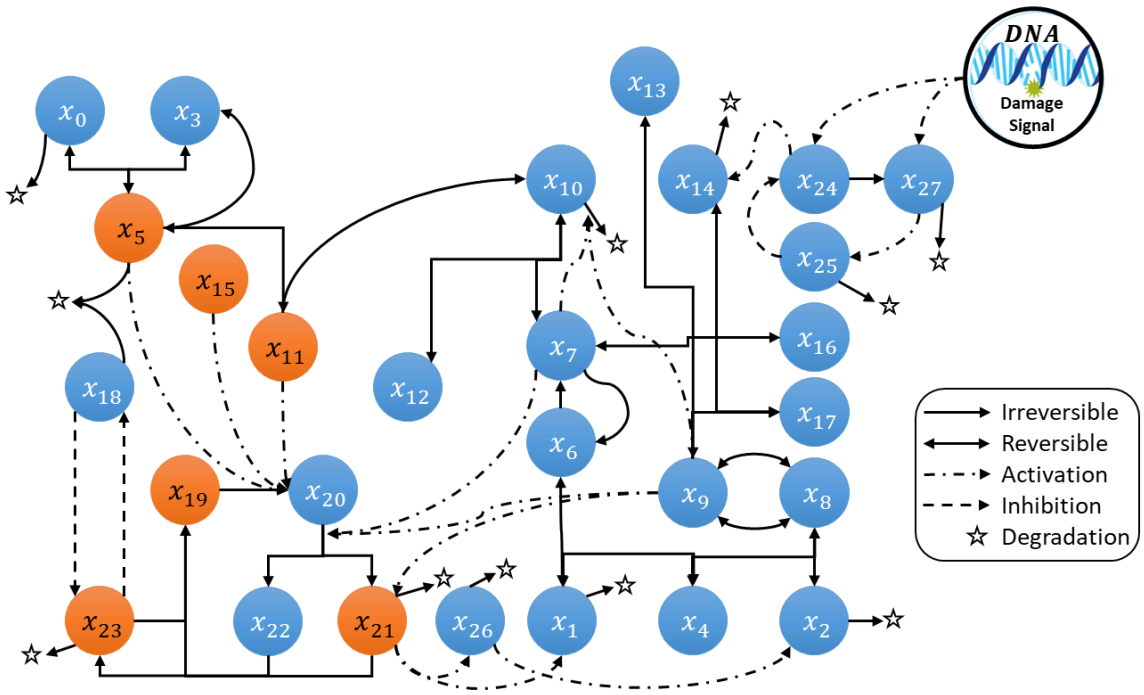


Figure 6.1. 2D structure of the G1/S checkpoint model involving the DNA damage signal transduction pathway using model variables for *ISP LOLAS*.

Table 6.1. Description of model variables of the G1/S checkpoint involving proteins to create the state-space for *ISP LOLAS*.

Biochemical species	Initial condition	Model variable
CycD/CDK4/6	2	x_5
p27/CycD/CDK4/6	1×10^{-3}	x_{11}
E2f	0	x_{21}
Rb	$5 \times e^{-2}$	x_{23}
Rb/E2f	1.95	x_{19}
p21/CycD/CDK4/6	0	x_{15}
CycD	3×10^{-2}	x_0
CDK4/6	5	x_3
p21	0	x_{14}
p27	6.3	x_{10}
p16	1×10^{-3}	x_{18}
Rb − PP/E2f	1×10^{-3}	x_{20}
CycE	1×10^{-3}	x_1
CycE/CDK2 − P	1×10^{-3}	x_7
CycA/CDK2 − P	1×10^{-4}	x_9
′X′	1×10^{-4}	x_{26}
Rb − PPPP	1×10^{-2}	x_{22}
p21/CycE/CDK2 − P	0	x_{16}
p21/CycA/CDK2 − P	0	x_{17}
p27/CycE/CDK2 − P	1	x_{12}
p27/CycA/CDK2 − P	1×10^{-4}	x_{13}
CDK2	13.5	x_4
CycE/CDK2	1×10^{-3}	x_6
p53	2.65×10^{-2}	x_{24}
CycA/CDK2	4×10^{-4}	x_8

Table 6.2. Reactions in the G1/S model involving the DNA-damage signal transduction pathway

$\boldsymbol{R_1}$: $\star \xrightarrow{k_1} x_0$	$\boldsymbol{R_{26}}$: $x_7 + x_4 \xrightarrow{k_{26}} x_{16}$	$\boldsymbol{R_{51}}$: $x_{21} \xrightarrow{k_{51}} x_{21}$
$\boldsymbol{R_2}$: $x_0 \xrightarrow{k_2} \star$	$\boldsymbol{R_{27}}$: $x_{16} \xrightarrow{k_{27}} x_{14} + x_7$	$\boldsymbol{R_{52}}$: $\star \xrightarrow{k_{52}} x_{21}$
$\boldsymbol{R_3}$: $x_0 + x_3 \xrightarrow{k_3} x_5$	$\boldsymbol{R_{28}}$: $x_8 + x_9 \xrightarrow{k_{28}} x_9$	$\boldsymbol{R_{53}}$: $x_{21} \xrightarrow{k_{53}} \star$
$\boldsymbol{R_4}$: $x_5 \xrightarrow{k_4} x_0 + x_3$	$\boldsymbol{R_{29}}$: $x_9 \xrightarrow{k_{29}} x_8$	$\boldsymbol{R_{54}}$: $x_{21} + x_9 \xrightarrow{k_{54}} \star$
$\boldsymbol{R_5}$: $x_{21} \xrightarrow{k_5} x_1$	$\boldsymbol{R_{30}}$: $x_9 + x_{10} \xrightarrow{k_{30}} x_{13}$	$\boldsymbol{R_{55}}$: $x_{22} \xrightarrow{k_{55}} x_{23}$
$\boldsymbol{R_6}$: $x_1 \xrightarrow{k_6} \star$	$\boldsymbol{R_{31}}$: $x_{13} \xrightarrow{k_{31}} x_{10} + x_9$	$\boldsymbol{R_{56}}$: $\star \xrightarrow{k_{56}} x_{23}$
$\boldsymbol{R_7}$: $x_1 + x_4 \xrightarrow{k_7} x_6$	$\boldsymbol{R_{32}}$: $x_9 + x_{14} \xrightarrow{k_{32}} x_{17}$	$\boldsymbol{R_{57}}$: $x_{23} \xrightarrow{k_{57}} \star$
$\boldsymbol{R_8}$: $x_6 \xrightarrow{k_8} x_1 + x_4$	$\boldsymbol{R_{33}}$: $x_{17} \xrightarrow{k_{33}} x_{14} + x_9$	$\boldsymbol{R_{59}}$: $x_{18} \xrightarrow{k_{59}} x_{23}$
$\boldsymbol{R_9}$: $x_{26} \xrightarrow{k_9} x_2$	$\boldsymbol{R_{34}}$: $\star \xrightarrow{k_{34}} x_{10}$	$\boldsymbol{R_{60}}$: $\star \xrightarrow{k_{60}} x_{24}$
$\boldsymbol{R_{10}}$: $x_2 \xrightarrow{k_{10}} \star$	$\boldsymbol{R_{35}}$: $x_7 + x_{10} \xrightarrow{k_{35}} \star$	$\boldsymbol{R_{61}}$: $Signal \xrightarrow{k_{61}} x_{24}$
$\boldsymbol{R_{11}}$: $x_2 + x_4 \xrightarrow{k_{11}} x_8$	$\boldsymbol{R_{36}}$: $x_9 + x_{10} \xrightarrow{k_{36}} \star$	$\boldsymbol{R_{62}}$: $x_{24} \xrightarrow{k_{62}} \star$
$\boldsymbol{R_{12}}$: $x_8 \xrightarrow{k_{12}} x_2 + x_4$	$\boldsymbol{R_{37}}$: $\star \xrightarrow{k_{37}} x_{14}$	$\boldsymbol{R_{63}}$: $\star \xrightarrow{k_{63}} x_{25}$
$\boldsymbol{R_{13}}$: $x_5 \xrightarrow{k_{13}} x_3$	$\boldsymbol{R_{38}}$: $x_{24} \xrightarrow{k_{38}} x_{14}$	$\boldsymbol{R_{64}}$: $x_{25} \xrightarrow{k_{64}} \star$
$\boldsymbol{R_{14}}$: $x_9 \xrightarrow{k_{14}} x_4$	$\boldsymbol{R_{39}}$: $x_{14} \xrightarrow{k_{39}} \star$	$\boldsymbol{R_{65}}$: $x_{27}^9 \xrightarrow{k_{65}} x_{25}$
$\boldsymbol{R_{15}}$: $x_8 \xrightarrow{k_{15}} x_4$	$\boldsymbol{R_{40}}$: $\star \xrightarrow{k_{40}} x_{18}$	$\boldsymbol{R_{66}}$: $x_{27}^9 \xrightarrow{k_{66}} x_{25}$
$\boldsymbol{R_{16}}$: $x_6 \xrightarrow{k_{16}} x_4$	$\boldsymbol{R_{42}}$: $x_{23} \xrightarrow{k_{42}} x_{18}$	$\boldsymbol{R_{67}}$: $x_{27} \xrightarrow{k_{67}} \star$
$\boldsymbol{R_{17}}$: $x_7.x_7 \xrightarrow{k_{17}} x_4$	$\boldsymbol{R_{43}}$: $x_{18} \xrightarrow{k_{43}} \star$	$\boldsymbol{R_{68}}$: $x_{21} \xrightarrow{k_{68}} x_{26}$
$\boldsymbol{R_{18}}$: $x_5 + x_{14} \xrightarrow{k_{18}} x_{15}$	$\boldsymbol{R_{44}}$: $x_{18} + x_5 \xrightarrow{k_{44}} \star$	$\boldsymbol{R_{69}}$: $x_{26} \xrightarrow{k_{69}} \star$
$\boldsymbol{R_{19}}$: $x_{15} \xrightarrow{k_{19}} x_{14} + x_5$	$\boldsymbol{R_{45}}$: $x_{21} + x_{23} \xrightarrow{k_{45}} x_{19}$	$\boldsymbol{R_{70}}$: $x_{24} + Signal \xrightarrow{k_{70}} x_{27}$
$\boldsymbol{R_{20}}$: $x_5 + x_{10} \xrightarrow{k_{20}} x_{11}$	$\boldsymbol{R_{46}}$: $x_{19} + x_5 \xrightarrow{k_{46}} x_{20}$	$\boldsymbol{R_{71}}$: $x_{24} + x_{25} \xrightarrow{k_{71}} x_{27}$
$\boldsymbol{R_{21}}$: $x_{11} \xrightarrow{k_{21}} x_{10} + x_5$	$\boldsymbol{R_{47}}$: $x_{11} + x_{19} \xrightarrow{k_{47}} x_{20}$	$\boldsymbol{R_{72}}$: $DDS \xrightarrow{exp(-k_{72}.t)} Signal$
$\boldsymbol{R_{22}}$: $x_6 + x_7 \xrightarrow{k_{22}} x_7$	$\boldsymbol{R_{48}}$: $x_{19} + x_{15} \xrightarrow{k_{48}} x_{20}$	$\boldsymbol{R_{73}}$: $x_{24} + x_{25} \xrightarrow{k_{73}} \star$
$\boldsymbol{R_{23}}$: $x_7 \xrightarrow{k_{23}} x_6$	$\boldsymbol{R_{49}}$: $x_7 + x_{20} \xrightarrow{k_{49}} x_{21}$	$\boldsymbol{R_{74}}$: $x_{24} \xrightarrow{k_{74}} \star$
$\boldsymbol{R_{24}}$: $x_7 + x_{10} \xrightarrow{k_{24}} x_{12}$	$\boldsymbol{R_{50}}$: $x_9 + x_{20} \xrightarrow{k_{50}} x_{21}$	$\boldsymbol{R_{75}}$: $x_{24} + DDS_{initial} \xrightarrow{k_{75}} \star$
$\boldsymbol{R_{25}}$: $x_{12} \xrightarrow{k_{25}} x_{10} + x_7$		

Table 6.3. Kinetic parameter values for reactions involved in the G1/S model

Parameter	Value	Parameter	Value	Parameter	Value
k_1	5×10^{-3}	k_{26}	2.25×10^{-2}	k_{51}	5×10^{-8}
k_2	5×10^{-4}	k_{27}	1.75×10^{-4}	k_{52}	5×10^{-7}
k_3	5×10^{-3}	k_{28}	1.9×10^{-2}	k_{53}	5×10^{-5}
k_4	2.5×10^{-3}	k_{29}	5×10^{-4}	k_{54}	1×10^{-2}
k_5	7.5×10^{-2}	k_{30}	2.5×10^{-3}	k_{55}	5×10^{-8}
k_6	2.5×10^{-3}	k_{31}	1.75×10^{-4}	k_{56}	5×10^{-5}
k_7	1.25×10^{-3}	k_{32}	2.5×10^{-3}	k_{57}	5×10^{-3}
k_8	2.5×10^{-4}	k_{33}	1.75×10^{-4}	k_{58}	5×10^{-5}
k_9	8×10^{-4}	k_{34}	5×10^{-8}	k_{59}	5×10^{-4}
k_{10}	5×10^{-4}	k_{35}	1×10^{-2}	k_{60}	1×10^{-4}
k_{11}	1×10^{-3}	k_{36}	1.5×10^{-3}	k_{61}	1.5
k_{12}	2×10^{-4}	k_{37}	5×10^{-5}	k_{62}	1×10^{-3}
k_{13}	5×10^{-4}	k_{38}	1×10^{-2}	k_{63}	9.4×10^{-4}
k_{14}	5×10^{-4}	k_{39}	5×10^{-3}	k_{64}	2×10^{-2}
k_{15}	5×10^{-4}	k_{40}	2×10^{-3}	k_{65}	9.5
k_{16}	5×10^{-4}	k_{41}	5×10^{-5}	k_{66}	10
k_{17}	2×10^{-3}	k_{42}	1×10^{-4}	k_{67}	5×10^{-3}
k_{18}	5×10^{-4}	k_{43}	5×10^{-4}	k_{68}	5×10^{-2}
k_{19}	5×10^{-3}	k_{44}	5×10^{-4}	k_{69}	8×10^{-4}
k_{20}	5×10^{-4}	k_{45}	5×10^{-5}	k_{70}	6
k_{21}	5×10^{-3}	k_{46}	2.5×10^{-3}	k_{71}	4×10^{-3}
k_{22}	2.5×10^{-2}	k_{47}	2.5×10^{-3}	k_{72}	1×10^{-8}
k_{23}	1.75×10^{-3}	k_{48}	2.5×10^{-3}	k_{73}	7.72×10^{-1}
k_{24}	2.25×10^{-2}	k_{49}	4×10^{-2}	k_{74}	5.56×10^{-2}
k_{25}	1.75×10^{-4}	k_{50}	2.5×10^{-3}	k_{75}	2×10^{-2}

These protein species counts and reactions (see Table 6.2) propensities will be used to define the state-space of the model for $ISP\ LOLAS$ and they are also responsible for changes in the number of states. Therefore, we track the copy counts of these species as:

$$([Chemical\ species]) \in \tilde{\mathrm{N}} \equiv (Model\ variables). \tag{100}$$

In reaction R_3, the copy count of x_5 is increased by the rate of association of CycD and $\mathrm{CDk4/6}$; however, the dissociation rate will decrease the copy count of x_5 through R_4. In reaction R_{13}, x_5 is decreased by 1, producing $\mathrm{CDk4/6}$, which increases the copy counts of x_3. The copy counts of x_{15} are increased by the rate of association of $\mathrm{p21}$ and $\mathrm{CyCD/CDk4/6}$, which decreases x_5 by 1, through R_{18}, and in the reversible reaction, R_{19}, it disassociates both proteins at the rate of 5×10^{-3} while increasing the copy counts of x_5, which are 10 times faster compared to the forward reaction. Similarly, R_{20} and R_{21} are a set of forward and backward reactions resulting in changes in the copy counts of x_5 and x_{11}. In reaction R_{51} the synthesis of x_{21} is promoted by itself at the rate of 5×10^{-8} in R_{50} and R_{51}; whereas, in reaction R_{52} the synthesis x_{21} is at the rate of 5×10^{-5}.

Similarly, by noting the changes in the counts of all the proteins, as given in Table 6.1, we can now define the transitions associated with R_M (as given in Table 6.2) in a stoichiometric vector V_M matrix for all $\tilde{\mathrm{N}}$ species involved in the reactions. Based on interactions of the proteins in Figure 6.1 (Iwamoto et al., 2008; Ling et al., 2010), we have R_M channels {3, 4, 13, 18, 19, 20, 21, 44, 45, 49, 50, 51, 52, 53, 54, 55, 56, 57, 59, 46, 47, 48, 1, 2, 37, 39, 27, 26, 33, 32, 38, 34, 24, 25, 30, 31, 35, 36, 40, 42, 43, 5, 6, 7, 8, 22, 23, 27, 26, 17, 28, 14, 29, 68, 69} and $\tilde{\mathrm{N}}$ species affecting the proteins - $\mathrm{CycD/CDK4/6}$, Rb, E2f, $\mathrm{CycE/CDK2}$, $\mathrm{p53}$ and $\mathrm{p21}$ - through a given number of reaction channels. By tracking the changes in the species, we will now define the transitions associated with $R_{M=3,4,13,18,19,20,21,\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2,23,27,26,17,28,14,29,68,69}$ and $\tilde{\mathrm{N}}$ in the stoichiometric vector matrix as:

$$V_M = \begin{pmatrix}
(1, 0, 0, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(-1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(-1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(-1, 0, 0, 0, 0, 1, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(1, 0, 0, 0, 0, -1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(-1, 1, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(1, -1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(-1, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, -1, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 1, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(-1, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, -1, 0, 0, -1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, -1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, -1, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, -1, 0, 0, 0, 1, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, -1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, -1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, -1, 1, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 1, 0, 0, 0, -1, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1) \\
(0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0)
\end{pmatrix}$$

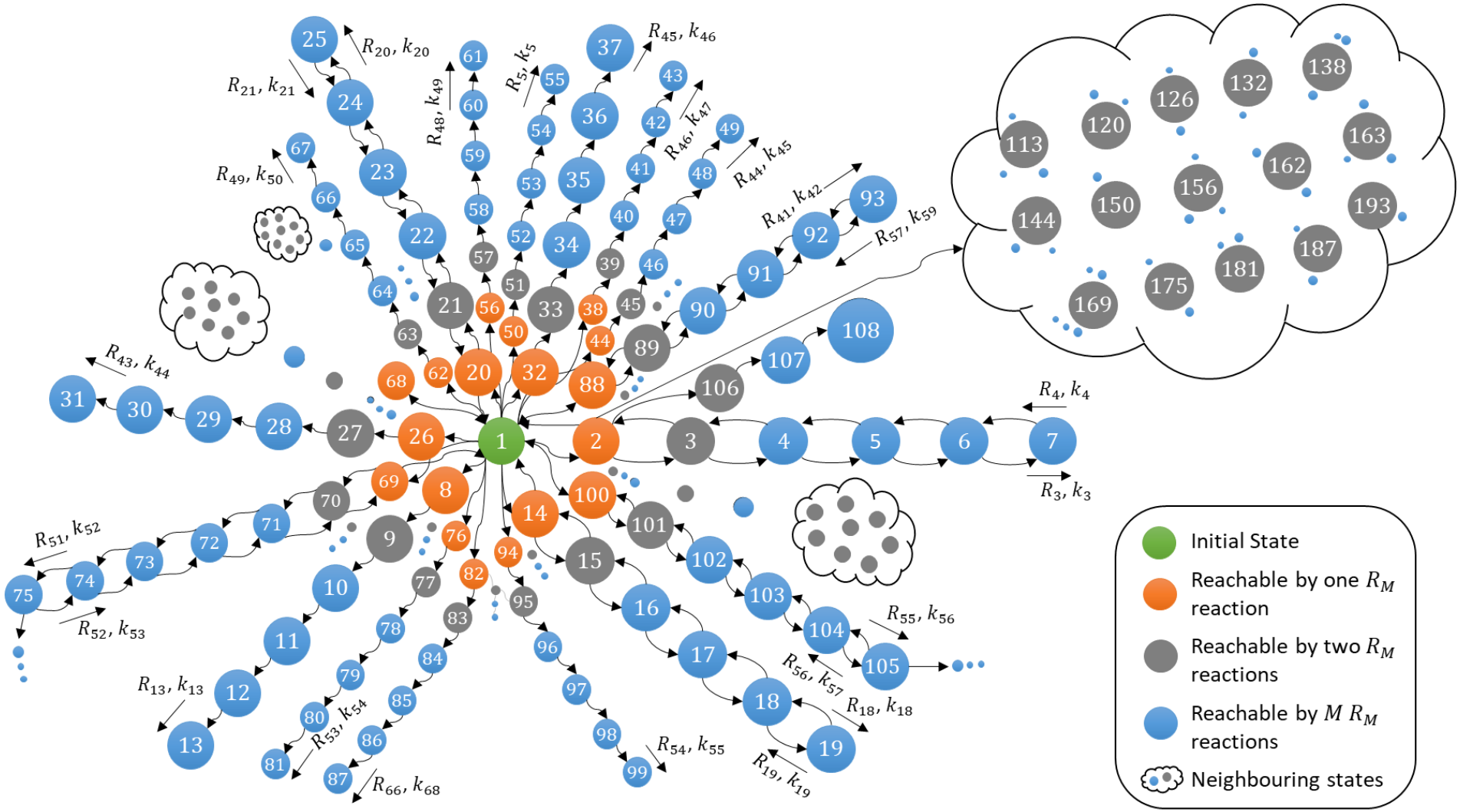


Figure 6.2. Markov chain graph given ● as the initial node of the G1/S checkpoint involving the DNA damage signal transduction pathway model. The ● nodes are directly reachable nodes from the initial state when exactly one R_M occurs. The ● nodes are reachable from the initial state when exactly two R_M occurs. When R_M occurs {3,4…….} times the system jumps to the ● nodes, respectively.

For method compatibility, the associated Markov chain of the model is treated as a Markov chain graph, as given in Figure 6.2, and stated in terms of the nodes with additional information, such as the number of R_M reactions required to reach the state. If Figure 6.2 is equivalent to the growing graph tree of the model, the transition between the nodes is defined by the typical form of the growing dictionary for this model, as given, below:

$$Dict = \left(\begin{array}{c} [1 \longrightarrow 2, 8, 14, 20, 26, 32, 38, 44, 50, 56, 62, 68, 69, 76, 82, 88, 94, 100], \\ [2 \longrightarrow 1, 106, 113, 120, 126, 132, 138, 144, 150, 156, 162, 163, 169, 175, 181, 187, 193], \\ [8 \longrightarrow 199, 205, 211, 217, 223, 229, 235, 241, 247, 253, 259, 260, 266, 272, 278, 284, 290], \\ [14 \longrightarrow 1, 296, 302, 308, 314, 320, 326, 332, 338, 344, 350, 351, 357, 363, 369, 375, 381, 387], \\ [20 \longrightarrow 1, 393, 399, 405, 411, 417, 423, 429, 435, 441, 447, 453, 454, 460, 466, 472, 478, 484], \\ [26 \longrightarrow 495, 501, 507, 513, 519, 525, 531, 537, 543, 549, 555, 556, 562, 568, 574, 580, 586, 592], \\ [32 \longrightarrow 599, 605, 611, 617, 623, 629, 635, 641, 647, 653, 659, 660, 666, 672, 678, 684, 690, 696, 702], \\ [38 \longrightarrow 709, 715, 721, 727, 733, 739, 745, 751, 757, 763, 769, 770, 776, 782, 788, 794, 800, 806], \\ [44 \longrightarrow 813, 819, 825, 831, 837, 843, 849, 855, 861, 867, 873, 874, 880, 886, 892, 898, 904, 910], \\ [50 \longrightarrow 917, 923, 929, 935, 941, 947, 953, 959, 965, 971, 977, 978, 984, 990, 996, 1002, 1008, 1014], \\ [56 \longrightarrow 1021, 1027, 1033, 1039, 1045, 1051, 1057, 1063, 1069, 1075, 1081, 1082, 1088, 1094, 1100, \\ 1106, 1112, 1118], [62 \longrightarrow 1125, 1131, 1137, 1143, 1149, 1155, 1161, 1167, 1173, 1179, 1185, \\ 1186, 1192, 1198, 1204, 1210, 1216, 1222], [68 \longrightarrow 1, 1229, 1235, 1241, 1247, 1253, 1259, 1265, \\ 1271, 1277, 1283, 1289, 1290, 1296, 1302, 1308, 1314, 1320], [69 \longrightarrow 1, 1327, 1333, 1339, 1345, \\ 1351, 1357, 1363, 1369, 1375, 1381, 1387, 1388, 1394, 1400, 1406, 1412, 1418], [76 \longrightarrow 1425, \\ 1431, 1437, 1443, 1449, 1455, 1461, 1467, 1473, 1479, 1485, 1486, 1492, 1498, 1504, 1510, 1516], \\ [82 \longrightarrow 1523, 1529, 1535, 1541, 1547, 1553, 1559, 1565, 1571, 1577, 1583, 1584, 1590, 1596, \\ 1602, 1608, 1614], [88 \longrightarrow 1, 1621, 1627, 1633, 1639, 1645, 1651, 1657, 1663, 1669, 1675, 1681, \\ 1682, 1688, 1694, 1700, 1706, 1712], [94 \longrightarrow 1719, 1725, 1731, 1737, 1743, 1749, 1755, 1761, \\ 1767, 1773, 1779, 1780, 1786, 1792, 1798, 1804, 1810, 1816], [100 \longrightarrow 1, 1823, 1829, 1835, \\ 1841, 1847, 1853, 1859, 1865, 1871, 1877, 1883, 1884, 1890, 1896, 1902, 1908, 1914], \ldots\ldots\ldots \end{array} \right)$$

Further, we express the propensity functions of all the R_M reactions (see Table 6.2) involved in the model for selective M as follows:

$$R_M: \quad a_M([Chemical\ species\ vector]) = k_M([Product\ of\ reactant\ species]) \qquad (101)$$

Figure 6.2 is used to visualise the Markov process of the model with $\mathbf{n}_J = (\mathbf{X}_K, \mathfrak{d}_l)$ number of states stored in all the nodes that are directly reachable from the initial node, $N_1 = (X_0, \mathfrak{d}_{l=1})$ via R_M reactions. In the next section, we will discuss the results of the computational experiments and visualise the state-space and conditional probabilities of the species.

6.3 Computational Experiments

In this section, we tested the ability of $ISP\ LOLAS$ to produce a large model to measure the dynamics of the key components, as discussed in sections 6.1 and 6.2. We performed the experiment on the carbon-neutral platform of Amazon® Web Service Elastic Computing (EC2) instance type large (m5a) running on HVM (hardware virtual environment) virtualisation with variable ECUs, a multicore environment 16vCPU @ 2.2GHz, AMD EPYC 7571 running Ubuntu 16.04.1 with its relevant dependencies, a 64GB memory with 8GB Elastic Block Storage (EBS) type General Purpose SSD (GP2) formatted with the Elastic File System (EFS). The performance mode was set to General Purpose with input-outputs per second (IOPS = 100/3000) and a throughput mode of type bursting is set (see Appendix F).

After $ISP\ LOLAS$ simulation, we collected the outputs from the events, as follows:

(1) Dimension of the G1/S checkpoint involving the DNA-damage signal transduction pathway. This is the indicating factor for knowing the difficulty level for solving the CME.

(2) Time taken by $ISP\ LOLAS$ to run the G1/S checkpoint involving the DNA-damage signal transduction pathway for CME approximation of up to t_f. This is a determining factor for the run-time performance and, subsequently, the total time required to solve the CME.

(3) Number of states at the time steps. Once the states are expanded, the domain size shows persistent increase in spite of bunking some states. This is critical for algorithm performance and maintenance of the accuracy of the solution, which requires a much longer time course in comparison with the expansion.

(4) Adaptive approximation error when states were simultaneously expanded and bunked.

(5) Conditional probabilities of the protein species at t_f and an approximate solution of the CME for the system in terms of probability.

The G1/S model (dimension = 28) with a DNA-damage signal transduction pathway is considered to be very stiff in nature, so molecular counts of certain proteins increase very quickly and some do so slowly (as discussed in section 6.1), which makes it tough to solve, even for a short time. The model is solved for $t_f = 1.5\ sec$ with $\mathfrak{b}_{limit} = 1$, $\tau_m = 1e-6$, $t_{step} = 0.1$. The systematic exploration of nodes carrying probable states are undertaken in a similar way, as discussed previously in Table 4.8 and depicted in Figure 4.16 in six stages (denoted as $\hat{S}$), representing R_M reactions with propensity, a_μ, with the arcs as transitions.

The nodes are expanded up to t_f for the identifying the reaction channels responsible for variations in the proteins. It is observed from the transitioning factor of the *2nd-tier* that every node has an average of at least ≈97 possible child nodes carrying states and; further, $Dict$ is expanded for *n*-tiers of child nodes to add more states to the domain. Further, $\mathbf{n}_J = \left(\mathbf{X}_K, \mathrm{a}_{l=1,2,\ldots\ldots}\right)$ is expanded and updated as per the trend of $ISP\ LOLAS$ (see section 4.4).

The repsonse from the $ISP\ LOLAS$ method between a number of states in the domain and time, t, is shown in Figure 6.3. The initial response suggests that only a few reactions were active until $t = 0.4$ *sec* and after that more reactions triggered that explosively take the exploration above 0.5 million states in 0.5 *sec*. For such a large model this combination of explosion states was expected because proteins undergo a number of excursions due to the number of reactions in fractions of time, t. The second explosion of states occurs after 1.0 *sec* when almost all the reactions (involving the species given in Table 6.1) become active in the network. The size and colour of the *2D pyramid* in Figure 6.3 shows the increase in size of the domain with the state explosions.

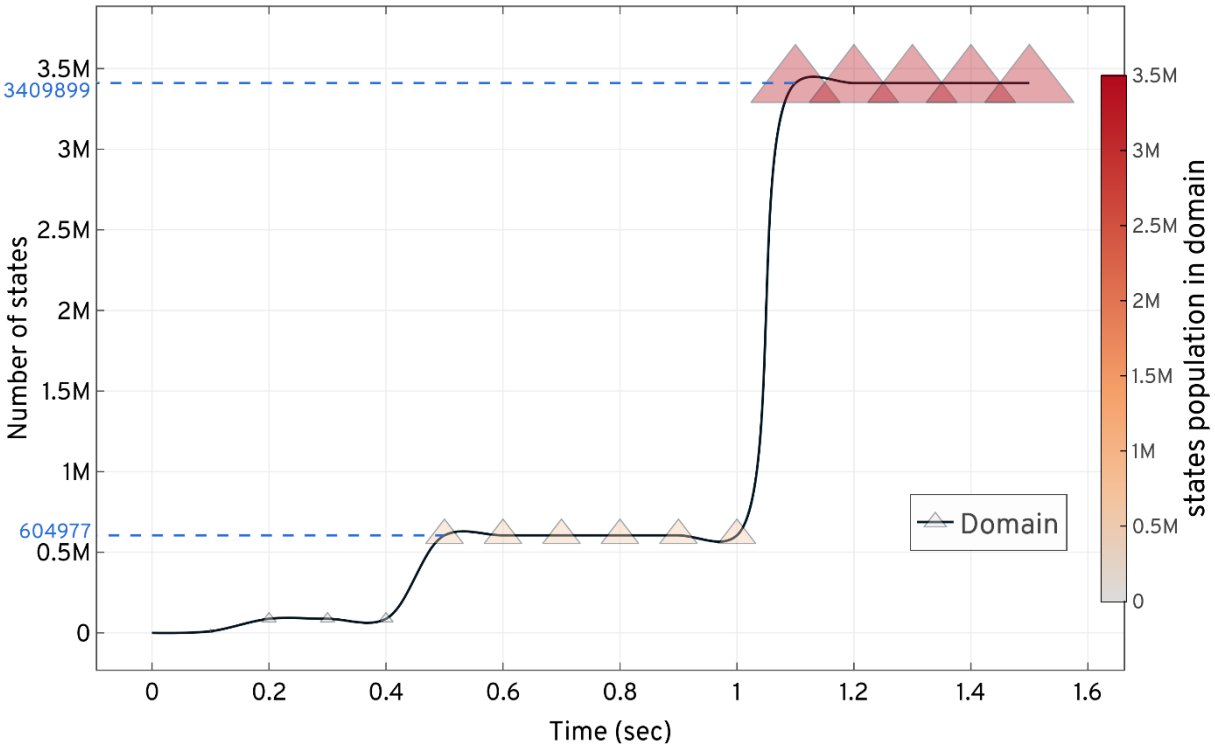


Figure 6.3. Showing the expansion and updating of the states for the G1/S model based on the *ISP LOLAS* method. The state-space expansion increases the number of additions of new states in the domain. *ISP LOLAS* quickly expands the state-space up to ≈3.5 million states in 1.5 *sec*.

The corresponding propensities, $\Delta a_{i,j}$, are updated in the $A_{i,j}$ matrix in every iteration, based on the $ISP\ LOLAS$ update trend (for example see Table 4.9). The system started with the initial state of the protein species and gradually, when protein level changes in the system, it exploits the copy counts that shift the system to a new state. We have set the checkpoint to examine the initial state probability over time. The response in Figure 6.4 indicate that the probability of the system to remain in the initial (normal) state decrease with indicates that the probability of the system remaining in the initial (normal) state decreases with time in the presence of DNA damage, which triggers the change in protein levels.

The change in protein levels causes the system to shift to new states, which manifest the Markov process of the system. The $ISP\ LOLAS$ captures this process and defines several bounds of the domain at different time intervals, as defined by Figure 4.2 pyramid. To investigage the expansion of states closely, the order of bounds at different time intervals, and the number of states present in the bounds, are given in Table 6.4. The size of bound created in each duration reveals that, for every step, the growth of the domain is eight-to-ten times the previous size of the domain.

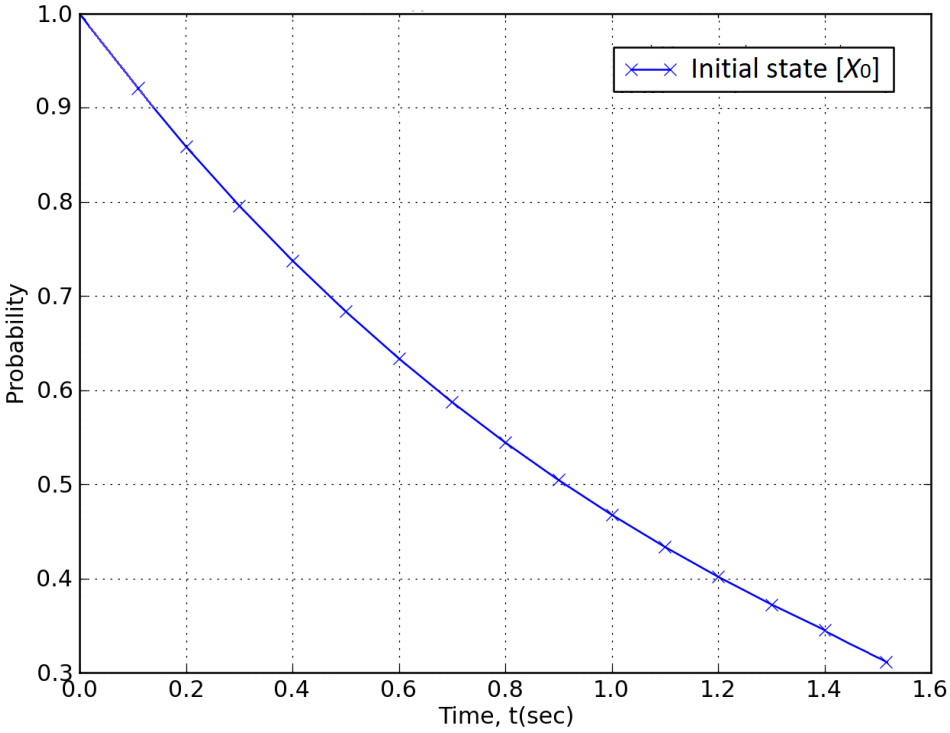


Figure 6.4. Response of the *ISP LOLAS* checkpoint for examining the G1/S checkpoint involving the DNA-damage signal transduction pathway's initial state probability over time.

Table 6.4. Lower and upper Bounds of the domain for G1/S model given by *ISP LOLAS* trend based on bound limit $б_{limit}$

Z	$Bound(Z)_{lower}$	$Bound(Z)_{upper}$	States	Duration
1	$Bound(1)_{lower} = \{X_0\}$ formed at $t = 0.0$ *sec* $Approximation = 1$	$Bound(1)_{upper} = \{X_{0,1,2.....9808}\}$ formed at $t = 0.1$ *sec* $Approximation = 0.999999867$	9808	0.0 – 0.1 *sec*
	$б_{limit} = 1$, $count(б_{limit}) = 0,1$,			
2	$Bound(2)_{lower} = Bound(1)_{upper}$ formed at $t = 0.1$ *sec* $Approximation = 0.999999847$	$Bound(2)_{upper} = \{X_{0,1,2.....87393}\}$ formed at $t = 0.2$ *sec* $Approximation = 0.999999173$	87393	0.1 – 0.2 *sec*
	$б_{limit} = 1$, $count(б_{limit}) = 0,1$			
3	$Bound(3)_{lower} = Bound(2)_{upper}$ formed at $t = 0.4$ *sec* $Approximation = 0.999999157$	$Bound(3)_{upper} = \{X_{0,1,2.....604677}\}$ formed at $t = 0.5$ *sec* $Approximation = 0.99999701$	604677	0.4 – 0.5 *sec*
	$б_{limit} = 1$, $count(б_{limit}) = 0,1$			
4	$Bound(4)_{lower} = Bound(3)_{upper}$ formed at $t = 1.1$ *sec* $Approximation = 0.99999699$	$Bound(4)_{upper} = \{X_{0,1,2.....3409899}\}$ formed at $t = 1.5$ *sec* $Approximation = 0.99999648$	3409899	1.1 – 1.5 *sec*
	$б_{limit} = 1$, $count(б_{limit}) = 0,1$			

The number of set of states that create the bounds at t are shown in Figure 6.5. In Figure 6.5, with the exploration of the set of 517584 states, the $Bound(3)_{upper} = \{X_{0,1,2.....604677}\}$ is formed at 0.5 *sec* carrying 604677 states. Some states were bunked at 0.5 *sec* resulting in approximation errors that reach $2.42e-06$ at 0.6 *sec*. At t_f, the *LOLAS* ends up with a domain defined by $Bound(4)_{upper} = \{X_{0,1,2.....3409899}\}$ carrying 3409899 states with $3.52e-06$ approximation errors.

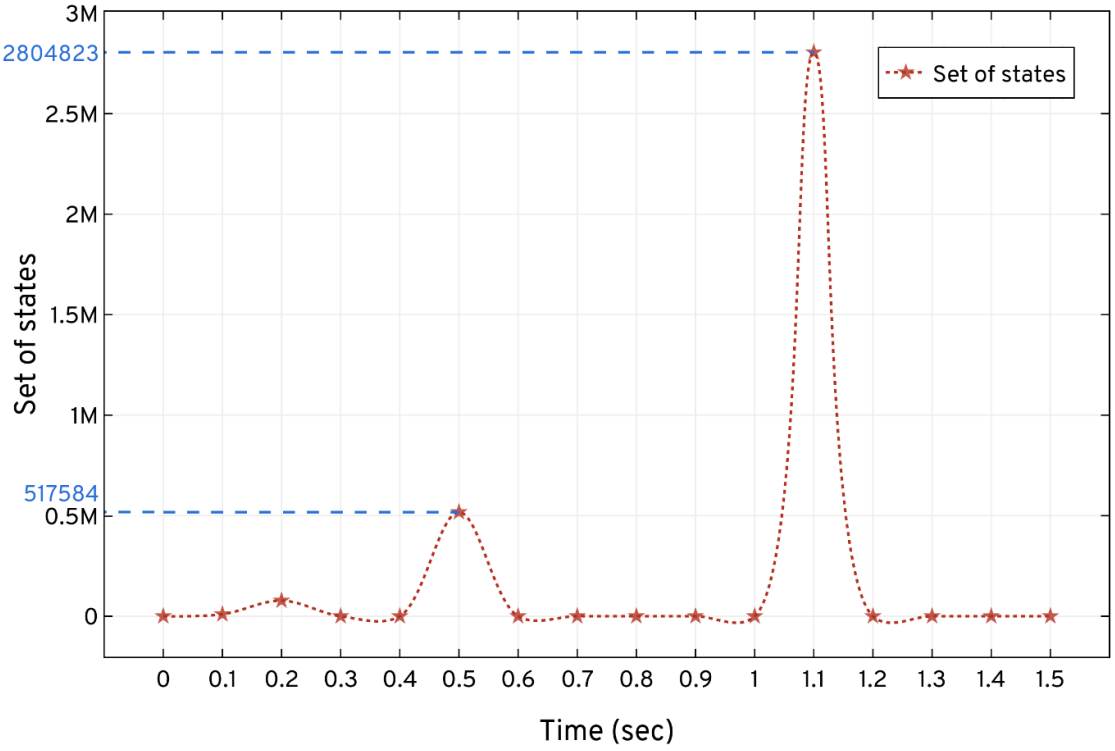


Figure 6.5. The set of states explored for G1/S model using the *ISP LOLAS* method. Based on network of reactions, *ISP LOLAS* unfolds the state-space pattern to update states in the domain and expands 3409899 states up to t_f. ★ shows the time point where new set of states is explored and updated in the domain.

The set of nodes $N_1, N_2, \ldots . N_{3409900}$ carries unique states representing the set of $state(n_{3409900}) = (X_{0,1,2,\ldots 3409899})$ that forms the state-space of the model. It is important to note that some proteins are synthesised and promoted by the network itself, as evidenced by some reactions of the pathway, which increase the frequency of the repeated states. However, *ISP LOLAS* validation does not consider them for the domain. The solution of Eq. (9) up to t_f for the domain created by *ISP LOLAS* is shown in Table 6.5.

Table 6.5. *ISP LOLAS* expansion response and solution at t_f for the G1/S model.

$t_f = 1.5$ *sec*, $t_{step} = 0.1$	Run-time (sec)	Domain	Expansion time (sec)	Error at t_f
ISP LOLAS	1372	3409899	1.5	$3.52\text{e}-06$

Over three test runs, the run time of *ISP LOLAS* for the G1/S model was 1372 *secs* in solving the Eq. (9) with the optimal domain having 3409899 states. The response of *ISP LOLAS* given in Figure 6.6, shows the system's probabilities bunked at t' while expansion (w.r.t approximation) when the number of states increases with the expansion and provided that, *ISP LOLAS* produces minimal errors of the order of 10^{-6}, as given in Table 6.5 and Figure 6.6.

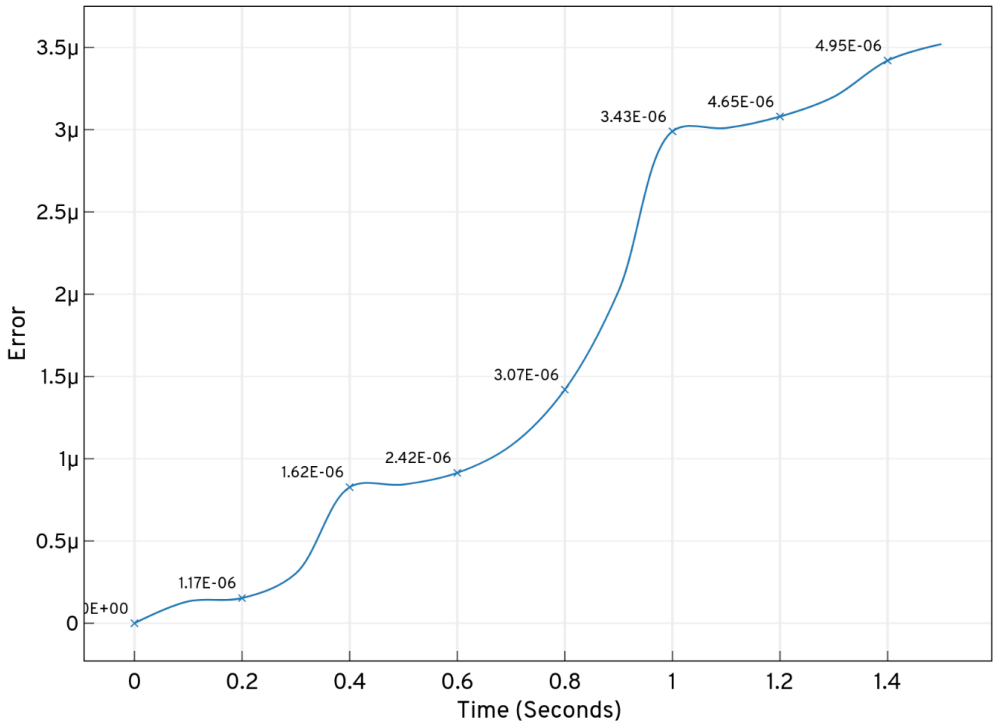


Figure 6.6. Total probability bunked at t' from the domain in *ISP LOLAS* iteration while expansion and solving the CME for G1/S model.

The conditional probabilities of the species' systems are given in Figure 6.7. In the case of the DNA-damage situation, large numbers of the most notable parameters (as mentioned in section 6.1) increase compared with the normal condition (in cell cycle progression). The increase is predominantly related to x_{14} (p21) having a high initial probability, see Fig(14) in Figure 6.7, and the feedback (negative) of x_{24} (p53) increases its probability, see Fig(24) in Figure 6.7, such as association rate of x_{16} (p21/CycE/CDK2 − P), rate of synthesis of x_{14} (p21) by x_{24} (p53), rate of degradation of x_{14} (p21), rate of synthesis of x_{24} (p53) by DNA-damage signal. The conditional probabilities of the two key proteins, x_{10} (p27) and x_1 (CycE), are affected by the change in the response of cells to the level of the DNA-damage signal, see Fig (10) and Fig (3) in Figure 6.7. The parameters related to x_{10} (p27) as well as x_1 (CycE) greatly affect the probability of x_{21} (E2f) with time, see Fig (21) in Figure 6.7. The impact of x_1 (CycE) involves additional parameters related to CycA because the release of supplementary x_{21} (E2f) depends on x_{20} (Rb − PP/E2f) hyperphosphorylation by the activation x_7 (CycE/CDK2 − P), which affects the probability of x_{21} (E2f).

When the release of x_{21} (E2f) is affected, the probability of x_1 (CycE) increases, see Fig (3) in Figure 6.7, that leads to progression to the S-phase followed by the temporary suspension of cell cycle progression. The increase in probability of x_{24} (p53) shows support to cells to repair the DNA damage.

The parameters and its probabilities relating to x_{14} (p21) and x_{24} (p53) become important in the case of DNA damage. When combined, the conditional probability of these parameters indicates the involvement of the DNA-damage signal in the transition of G1/S.

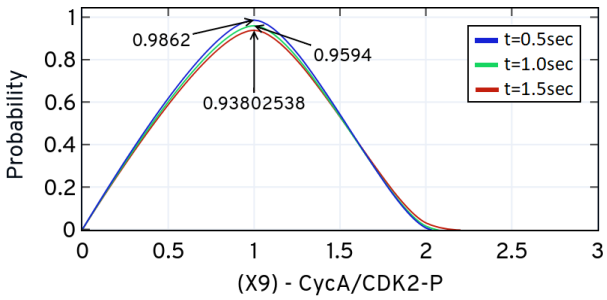


Fig (1). Probability of CycA/CDK − P over t_f

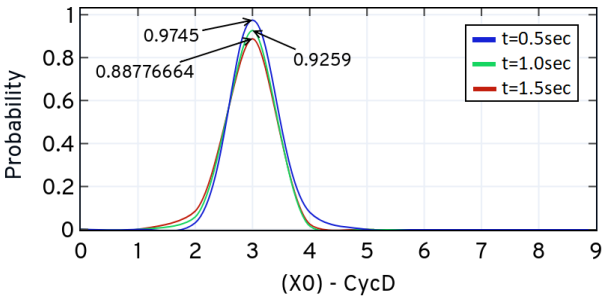


Fig (2). Probability of CycD over t_f

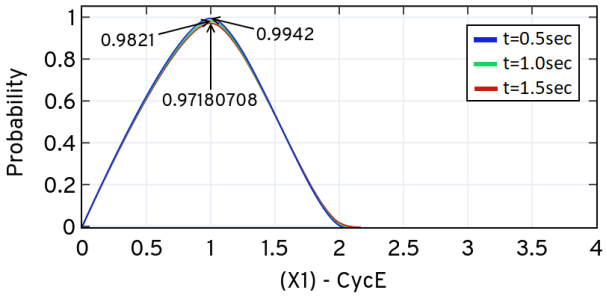


Fig (3). Probability of CycE over t_f

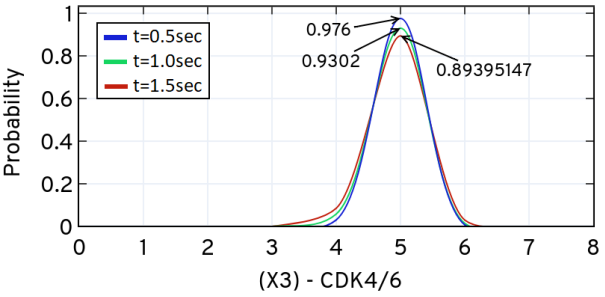


Fig (4). Probability of CDK4/6 over t_f

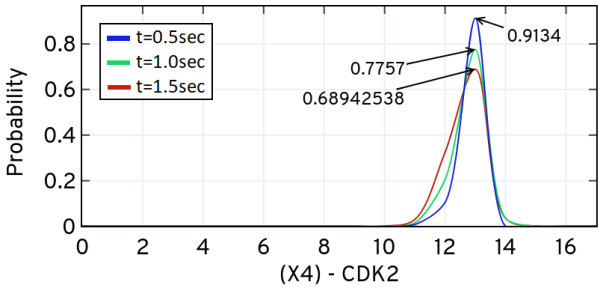


Fig (5). Probability of CDK2 over t_f

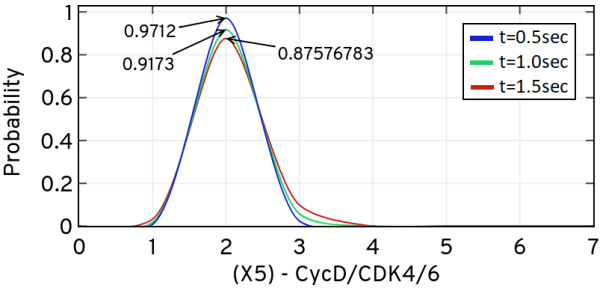


Fig (6). Probability of CycD/CDK4/6 over t_f

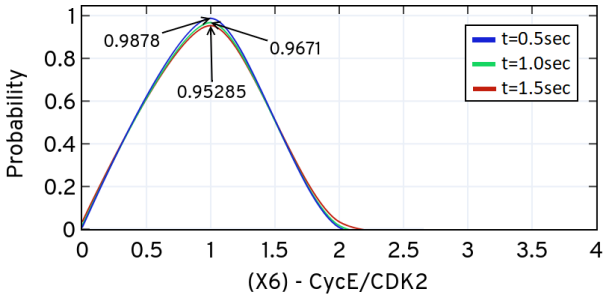


Fig (7). Probability of CycE/CDK2 over t_f

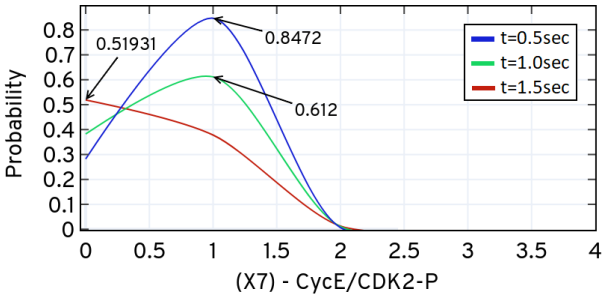


Fig (8). Probability of CycE/CDK2 − P over t_f

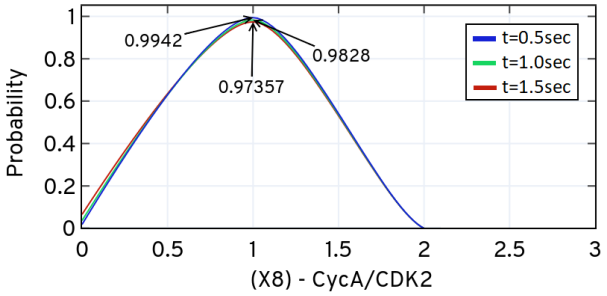


Fig (9). Probability of CycA/CDK2 over t_f

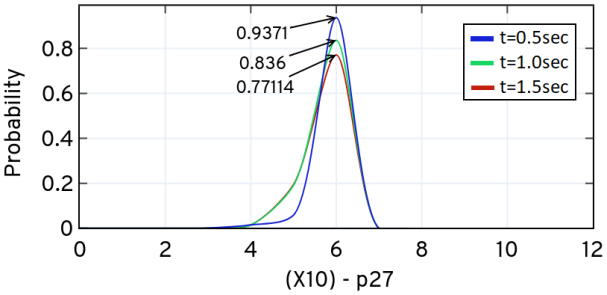


Fig (10). Probability of p27 over t_f

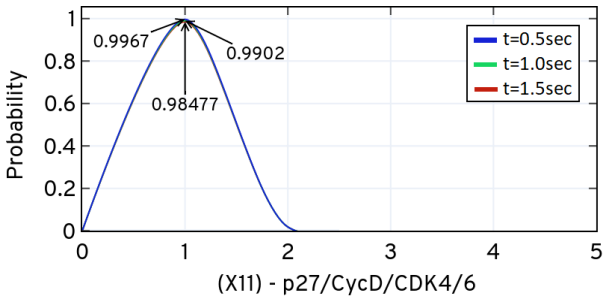


Fig (11). Probability of p27/CycD/CDK4/6 over t_f

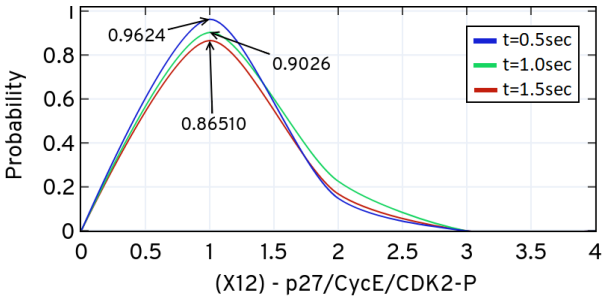


Fig (12). Probability of p27/CycE/CDK2 − P over t_f

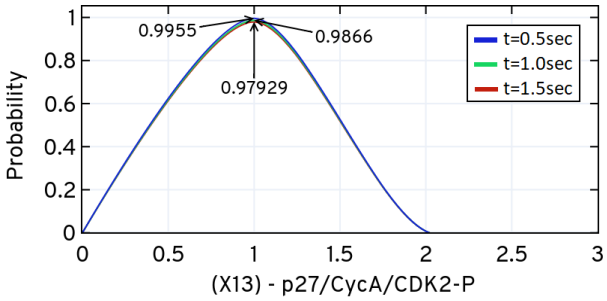


Fig (13). Probability of p27/CycA/CDK2 − P over t_f

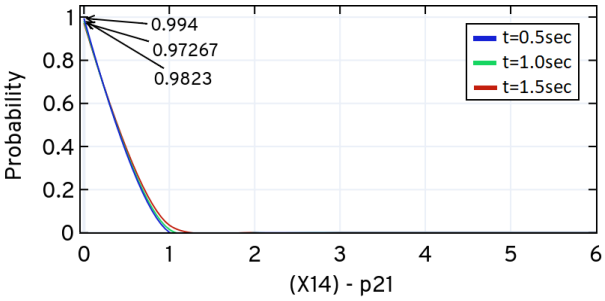


Fig (14). Probability of p21 over t_f

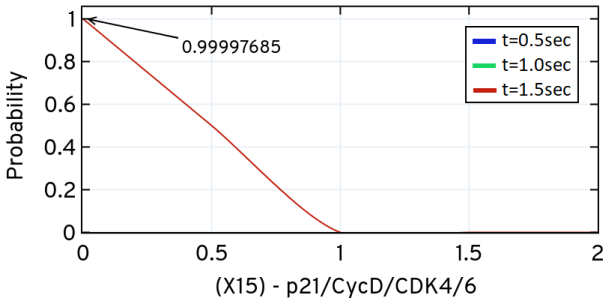


Fig (15). Probability of p21/CycD/CDK4/6 over t_f

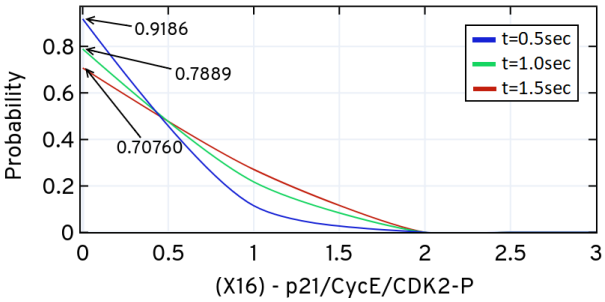


Fig (16). Probability of p21/CycE/CDK2 − P over t_f

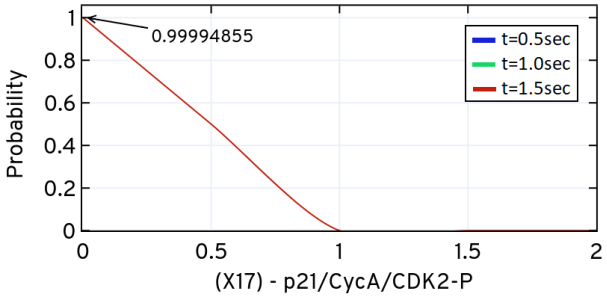


Fig (17). Probability of p21/CycA/CDK2 − P over t_f

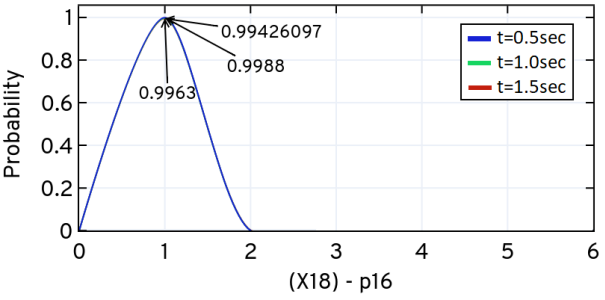


Fig (18). Probability of p16 over t_f

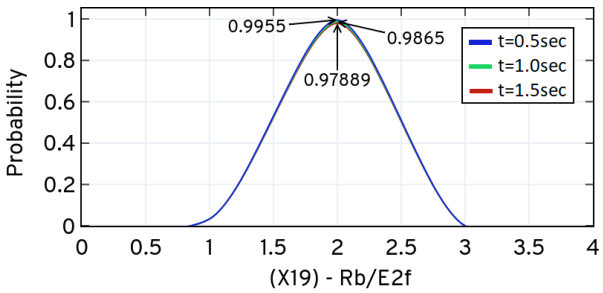


Fig (19). Probability of Rb/E2f over t_f

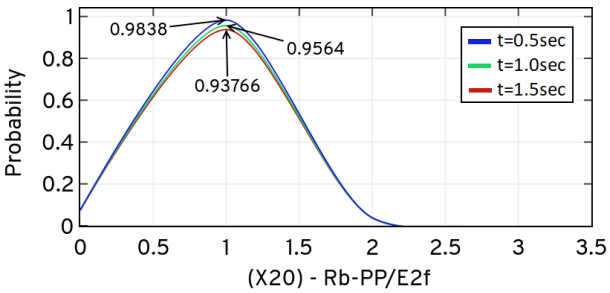


Fig (20). Probability of Rb − PP/E2f over t_f

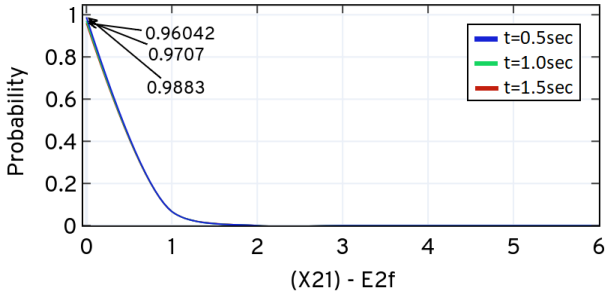


Fig (21). Probability of E2f over t_f

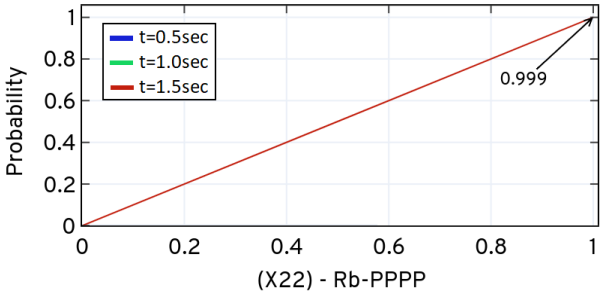


Fig (22). Probability of Rb − PPPP over t_f

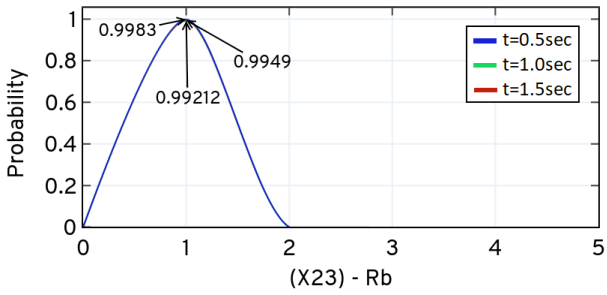


Fig (23). Probability of Rb over t_f

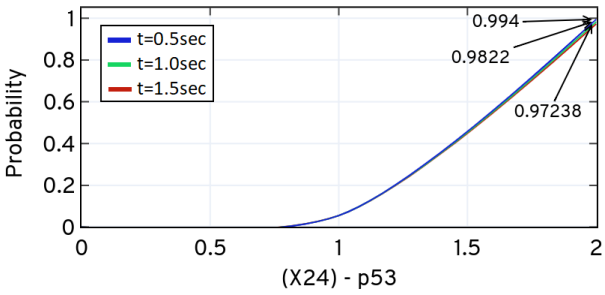


Fig (24). Probability of p53 over t_f

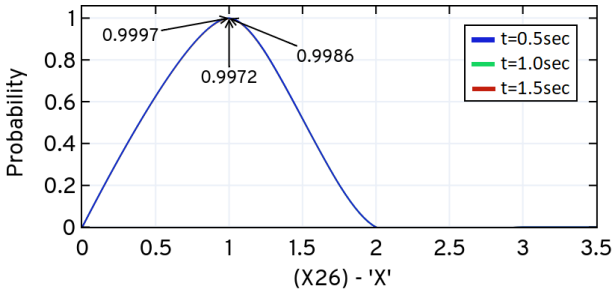


Fig (25). Probability of 'X' over t_f

Figure 6.7. Conditional probability of the G1/S model evaluated at $t_f = 1.5\ sec$, $t_{step} = 0.1$ using *ISP LOLAS*.

6.4 Discussion and Summary

In this chapter, we extended the application of our ISP method to a large model, compared to application presented on models in the previous chapter, to investigate the performance of $ISP\ LOLAS$ for the expansion of the state-space and an approximate solution of the CME based on produced bounds. We simulate the G1/S model under the condition of the DNA-damage signal to study the number of states at different time points. ISP also predicts the conditional probabilities of the protein species (see Figs (1) to (25) of Figure 6.7) based on events leading to the formation of different complexes in the system.

The simulation results of the expansion of states show that the explosion of states does not take place at every time step but with gaps of several time steps. The first explosion took the system to 0.5 million states and the second the explosion to 3.5 million states (see Figure 6.3 and Figure 6.5). This sudden explosion is due to changes in the levels of proteins in the presence of DNA-damage signals, which are useful to gaining insights into the different complexes probabilities over time. Both the explosions were also captured in Figure 6.5, which shows the time points where the new set of states is explored and updated in the domain.

This also helps us to better understand the potential outcomes of the domain in terms of bounds defined at different time durations. Although the number of states increased with the rate of 8-10 times per bound, the accuracy of the approximate solution was not greatly affected (see Table 6.4). The solution in Table 6.5 clearly reflect the pros of $ISP\ LOLAS$ in terms of performance (run-time), number of states (domain size) and the error at t_f for a large model. In addition, the checkpoint response of ISP in Figure 6.4 shows the decline in the initial state probability over time. In Figure 6.6, the adaptive approximation error marked by $ISP\ LOLAS$ shows a major change after $t = 0.4$ *sec* and $t = 1.0$ *sec* at the first and second explosion of the states, respectively. This suggests that both explosions bring states with lower probabilities that were bunked from the domain for approximation.

Overall, these results provide a clearer picture about the performance of the $ISP\ LOLAS$ on a large model and the expansion of the state-space of the G1/S model in the presence of DNA-damage and how the probabilities of different protein species change over time. Collectively, the behaviour of these protein species defines the dynamic nature of the G1/S model. For replication of this experiment, refer to Appendix F and Appendix G.

Page intentionally left blank

Chapter 7

Case study 2 – Oxidative stress adaptation in the Fungal Pathogen *Candida albicans*

The main objective of this chapter is to investigate the performance of the numerical *ISP LAS* algorithm on a very large biochemical system. We analyse the state-space, domain formation and probabilities of important species by implementing and integrating the mathematical model of oxidative stress adaptation in the fungal pathogen, *Candida albicans*.

With the advances in healthcare systems, as the number of immunocompromised patients has increased, they are vulnerable to various infections caused by pathogenic microbes. *Candida albicans* is well-known among fungal pathogens. The infection caused by *Candida albicans* is called candidiasis, which is categorised based on the extreme effects of the disease (Kabir et al., 2012). Cardiovascular and bloodstream infections are caused by *Candida* species but bloodstream infections are the most persistent of the two. Because of this, *Candida albicans* has gained significance as a pathogen in humans. *Candida albicans* has a vigorous feedback to oxidative stress that is important in its ability to infect (Komalapriya et al., 2015). This response prevents the reactive oxygen species produced by the hosts' immune cells attempting to kill the fungal infection. To understand the interactions between host and fungus, it is important to have good knowledge of the dynamic processes that initiate *Candida albicans* oxidative stress. Oxidation that occurs in human bodies is a conventional and essential process. Stress occurs when there is a disparity between its activities and radicals cannot help fight off pathogens, which leads to infections. When more radicals are present, this creates an imbalance between antioxidants and this damages the tissues causing oxidative stress.

In this chapter, in section 7.1, we will briefly introduce the dynamic nature of an integrative model of oxidative stress adaptation in the fungal pathogen *Candida albicans*. In section 7.2, we will prepare and integrate the model with our *ISP LAS* algorithm and, in section 7.3, we will analyse the state-space of the model and the performance of the *ISP LAS* algorithm using computational experiments. In section 7.4, we give a brief discussion and summary of the study and its results.

7.1 Introduction

The mathematical model of oxidative stress adaptation in the fungal pathogen, *Candida albicans,* as proposed in (Komalapriya et al., 2015), is a combination of several pathways and systems. It has $Cap1$ and $Hog1$ *pathways* (oxidative stress signalling pathways) and three antioxidant systems - catalase, thioredoxin and glutathione. It also has a pentose phosphate pathway (PPP), which is important for oxidative stress adaption in reducing equivalents. *C. albicans* exists in about 70% of individuals in their urogenital and gastrointestinal tracts, oral cavities and as commensals on skin. There are also about 5% patients who have recurring vaginal and oral infections caused by *C. albicans.* However, in the case of immunocompromised patients, *C. albicans* infections can be life-threatening (Schulze et al., 2009) and the mortality rates related to candidiasis in transplant and chemotherapy patients are between 45% - 75%.

C. albicans survives in human hosts when they outlast phagocytes attacks or successfully avoid recognition by the immune system. Phagocytic attacks kill the microorganisms with the help of toxic chemicals, together with H_2O_2 (reactive oxygen species), which has toxic as well as cytostatic effects (Brown, 2011). The potential of *C. albicans* to survive submitting to reactive oxygen species is noteworthy in terms of pathogen-host interactions because oxidative stress response inactivation reduces the effect of the severity from fungus attack and its ability to act against phagocytic killing (Arana et al., 2007; Brown et al., 2012). In fungal cells low level reactive oxygen species act as messengers to promote development and growth (Rinnerthaler et al., 2012) and, at higher levels, apply cytotoxic effects by damaging proteins, DNA, lipids and altering cellular redox homeostasis.

Similar to other yeasts, *C. albicans* responds by triggering the reactive oxygen species detoxification procedure, introducing antioxidant systems (thioredoxin and glutathione systems) and restoring the oxidative damage done by reactive oxygen species to groups of protein-thiols (Brown et al., 2012). In *C. albicans*, the detoxification of H_2O_2 is moderated by these antioxidants. Catalase acts as a catalyst for the dismutation of H_2O_2 and the detoxification of H_2O_2 is moderated by the glutathione system. $GSSG$ (glutathione disulphide) is formed when GSH (glutathione) is oxidised, which is then reduced through $Glr1$ (glutathione reductase) using $NADPH$. The protein-thiols, which are damaged oxidatively, are repaired by the glutathione system via $Ttr1, Grx3, orf19.4150$ and Grx1 glutaredoxins. The reductant Trx1 (thioredoxin) is used by Tsa1 (peroxiredoxin) when H_2O_2 is detoxified by the thioredoxin system. Further, $NADPH$ reduces the oxidised Trx1 through Trr1 (thioredoxin

reductase) action. *C. albicans* introduces the main components of the thioredoxin, catalase and glutathione antioxidant systems, including the production of $NADPH$ through PPP (pentose phosphate pathway) when exposed to H_2O_2. This action is contributed to by $Cap1$ and partially supported by $Hog1$ signalling (Enjalbert et al., 2006).

While significant efforts have been made to understand the oxidative stress response at the molecular level, the dynamics of antioxidant systems and some signalling systems, such as *hog1* and c*ap1,* are yet be to explored. The collection of these systems, pathways and their components would help in understanding the dynamics and mechanisms of the system in better ways as well as their individual contributions to the system. The comprehensive model constructed by (Komalapriya et al., 2015) is the first model for medically-related microorganisms that unites $NADPH$ production through PPP, antioxidant systems (thioredoxin, glutathione and catalase) and signalling ($Hog1$ and $Cap1$).

7.1.1 Integrated Model Overview

The comprehensive model is a combination of several modules that includes a: (i) transporter module; (ii) antioxidant module; (iii) protein-thiol module; and (iv) signalling and gene expression module, as shown in Table 7.1. This model can be used to: (i) anticipate the presence of extra types of the key oxidative pressure controllers ($Hog1$ and $Cap1$); (ii) spotlight the duties of antioxidant systems in the maintenance of damage repair and cellular redox potentials; (iii) evaluate the impacts of key mutations on the capacity of *C. albicans* cells to react to oxidative stress; (iv) feature the part of catalase in the principle detoxification system under the conditions inspected; and (v) explain the aggregate time-and-portion determined responses of oxidative stress response system under various stress situations.

Table 7.1. Modules and components of comprehensive network of the integrated model.

Module	**Description**	**Components**
Transporter	Based on diffusion of H_2O_2 (external and internal) it represents the dynamics of peroxide.	$H_2O_2{}^{Ex}$, $H_2O_2{}^{In}$
Antioxidant	Through enzyme catalase it specifies the H_2O_2 detoxification pathway. Under different conditions (stress and non-stress) it keeps the track of the maintenance of NADPH. During oxidative stress, it is responsible for repairing protein mono-thiols and protein di-thiols. In the elimination of H_2O_2, it demonstrates the role of the thioredoxin and glutathione pathways in maintaining cellular redox potential.	$Cat1$, GSH, $GSSG$, Gpx, $Glr1$, $Ttr1^{Red}$, $Ttr1^{Ox}$, $Tsa1^{Red}$, $Tsa1^{Ox}$, $Trx1^{Red}$, $Trx1^{Ox}$, $Trr1^{Red}$, $Trr1^{Ox}$, $NADPH$, $NADP$
Protein-thiol	During oxidative stress, it tracks the damage and repair the protein mono-thiols	$Pr.SH$, $Pr.SOH$, $Pr.SSG$, $Pr.(SH)_2$, $Pr.SS$
Signalling and gene expression	During oxidative stress, it describes the dynamics of the protein kinase pathway and activation of factors like AP-1 transcription. Describes antioxidant gene regulation under different conditions.	$Cap1^N$, $Cap1^A$, $Cap1^I$, $Ssk2$, $Ssk2.P$, $Pbs2$, $Pbs2.$PP, $Hog1^N$, $Hog1^I$, $Hog1^N.PP$, $Hog1^I.PP$, $CAT1$, $CAP1$, GPX, $GLR1$, $TTR1$, $TSA1$, $TRR1$, $TRX1$, GSH, $NADPH$, $SSK2$, $PBS2$, $HOG1$

Through the *C. albicans* plasma membrane, the chemical kinetics of diffusion of H_2O_2 and chemical degradation are described by the *transporter* module. Here, catalase degrades the externally introduced H_2O_2 that enters the cell through diffusion (Bienert et al., 2006; Branco et al., 2004). In addition, the cells of *C. albicans* produce the base level of external H_2O_2 and, in spite of the absence of externally introduced H_2O_2, $Cat1$ removal results in the deposition of intracellular reactive oxidant species (Komalapriya et al., 2015).

The repairing of damage and elimination of H_2O_2 is tracked in antioxidant system through its components (see Table 7.1). This also includes PPP (pentose phosphate pathway), which is the source of NADPH. The $Cat1$ (catalase) represents the H_2O_2 degradation to form H_2O and O_2. According to a study by (Komalapriya et al., 2015), the $Cat1$ plays vital role in the elimination of H_2O_2 because the single locus of $Cat1$ negatively affects the decay rate of extracelluar H_2O_2. Under experimental conditions (Kaloriti et al., 2012), the rate of degradation of extracelluar H_2O_2 remains the same even after Trx1 or Glr1 inactivation but that does not diminish the significance of the roles of the thioredoxin and glutathione systems in the pathway. GSH (glutathione) signifies the place of abundant tripeptides in the response; however, Gpx (single glutathione peroxidase) entity was examined in this model. Through Glr1 (glutathione reductase), oxidised GSSG is recycled to form GSH relying upon NADPH (Dickinson et al., 2002).

Thioredoxin is responsible for repairing protein damage and decribing H_2O_2 detoxification. Thioredoxin is a class of thiol disulphide oxidoreductases having low molecular weights. In *C. albicans*, out of the two genes for thioredoxin, only TRX1 has a role in the stress response. Hence, we are only considering Trx1 for its role in stress response. With H_2O_2 disclosure, the model also defines Tsa1 (peroxiredoxin) oxidation through peroxide. Further, $Tsa1^{Ox}$ is decreased by $Trx1^{Red}$ to produce $Trx1^{Ox}$, and the action of Trr1 (thioredoxin reductase), together with reductant NADPH, reconverts $Trx1^{Ox}$ to its native form. The PPP (pentose phosphate pathway) depicts the NADPH production that happens when exposed to oxidative stress to assist the detoxification of H_2O_2 through major antioxidant modules (thioredoxin and glutathione systems). PPP gene expression increases the production of NADPH via Yap1-based induction (Chechik et al., 2008; Godon et al., 1998; Lee et al., 1999). In addition, the shift of the metabolic flux via the PPP oxidative stream increases the production of NADPH. The PPP sub-module in our model does not detail the pathway in full, but describes these two phases in an abstract fashion.

Protein mono- and di-thiol reactions during oxidative stress are described by the protein thiol

module (Komalapriya et al., 2015). The mono- and di-thiols manage the repair of protein sulfenic acid and protein disulphides, respectively. The last module in the model of (Komalapriya et al., 2015) accounts for $Cap1$ and $Hog1$ signalling pathways as well as the procedure for genes under different conditions (normal and oxidative stress). In this module, the mRNA of genes were changed under normal condition at base rates and then influence the response to oxidative stress. Gene expression is modelled through the Hill function and the this increase is moderated by the $Cap1$ and $Hog1$ pathways. The increase in the levels of protein are used as inputs for the antioxidant modules and protein repair. However, during oxidative stress response, the production rates of $Hog1$, $Pbs2$ and $Ssk2$ remain constant.

In section 7.2, we will consider the systems and pathways discussed in section 7.1 to understand the complex structure of the integrative model to create a compatible model for the ISP algorithm. We also discuss the modelling and integration of the model for $ISP\ LAS$, as discussed in section 4.3, as well as the hypothesis and assumptions underlying this method.

7.2 Model Integration

To demonstrate the numerical $ISP\ LAS$ algorithm on a large model, we considered the reactions involved in the systems and pathways (see section 7.1.1) to understand the dynamic behaviour of the model through state-space and species probabilities. The system is a network of 45 chemical and biochemical species (see Table 7.2) reacting based on 121 biochemical reactions (see Table 7.4) with 121 kinetic parameters ($k_{M=121}$) (see Table 7.5), 3 auxiliary variables (see Table 7.3). These parameters, variables and 45 mass balance ODEs characterise the robustness and dynamic behaviour of the network.

The ISP method is initialised and had its parameters set using the initial conditions (given in Table 7.2, Table 7.3, Table 7.4 and, Table 7.5) of the model and integrated as a function. Due to the large number of mathematical operations and equations in framing the pathway, simultaneous parameter predictions with limited number of sub-module experimental values at any instance is often complicated for dynamic systems. Therefore, the consistency of the available experimental data and assumptions was ensured at each step of the $ISP\ LAS$. This method has led to successful development of several functions for each module that integrates large numbers of processes supporting the extensive expansion of the state-space.

The presence of stress signals and kinetic parameters affects various critical species as discussed in section 7.1. We will track these critical species based on different modules in the integrated model of this network. The dynamic behaviour of the whole system depends on how these species and their probabilities change with time and the increasing size of the domain over time that will provide information about the robustness of the network. We will now assign model variables to all the chemical and, biochemical species present in the system, as given in the Table 7.2, which also shows the initial condition values (in *M*) (Komalapriya et al., 2015) in terms of the concentration of the species present in the system.

Table 7.2. The description of model variables of the oxidative stress response in *C. albicans* pathway involving chemical and biochemical species to create the state-space for *ISP LAS*

Species	Initial condition (in *M*)	Model variable	Species	Initial condition (in *M*)	Model variable
$H_2O_2{}^{Ex}$	0	x_0	$Ssk2.P$	0	x_{25}
$H_2O_2{}^{In}$	1×10^{-9}	x_1	$Pbs2$	9.2870×10^{-8}	x_{26}
$Cat1$	$6.4493\ x\ 10^{-6}$	x_2	$Pbs2.PP$	0	x_{27}
GSH	$4.708\ x\ 10^{-3}$	x_3	$Hog1^N$	2.9151×10^{-7}	x_{28}
$GSSG$	$2.485\ x\ 10^{-3}$	x_4	$Hog1^N.PP$	0	x_{29}
Gpx	1.7284×10^{-7}	x_5	$Hog1^I$	0	x_{30}
$Glr1$	3.2676×10^{-7}	x_6	$Hog1^I.PP$	0	x_{31}
$Ttr1^{Red}$	2.5578×10^{-6}	x_7	$CAP1$	4.9367×10^{-11}	x_{32}
$Ttr1^{Ox}$	0	x_8	$CAT1$	3.5894×10^{-10}	x_{33}
$Pr.SH$	1.22×10^{-4}	x_9	GPX	2.7250×10^{-10}	x_{34}
$Pr.SOH$	0	x_{10}	$GLR1$	3.3723×10^{-10}	x_{35}
$Pr.SSG$	0	x_{11}	$TTR1$	8.1088×10^{-10}	x_{36}
$Pr.(SH)_2$	1×10^{-3}	x_{12}	$TSA1$	4.0247×10^{-9}	x_{37}
$Pr.SS$	0	x_{13}	$TRX1$	1.6183×10^{-9}	x_{38}
$Tsa1^{Red}$	1.7429×10^{-5}	x_{14}	$TRR1$	3.7187×10^{-10}	x_{39}
$Tsa1^{Ox}$	0	x_{15}	$GSH.mRNA$	3.5894×10^{-10}	x_{40}
$Trr1^{Red}$	1.2572×10^{-5}	x_{16}	$NADPH.mRNA$	3.5894×10^{-10}	x_{41}
$Trr1^{Ox}$	0	x_{17}	$SSK2$	8.3999×10^{-11}	x_{42}
$Trx1^{Red}$	1.1518×10^{-6}	x_{18}	$PBS1$	4.8373×10^{-11}	x_{43}
$Trx1^{Ox}$	0	x_{19}	$HOG1$	3.1587×10^{-10}	x_{44}
$NADPH$	150×10^{-6}	x_{20}	H_2O	0	x_{48}
$Cap1^N$	6.9652×10^{-8}	x_{21}	H^+	1	x_{49}
$Cap1^A$	0	x_{22}	O_2	0	x_{50}
$Cap1^I$	0	x_{23}	$NADP^+$	0	x_{51}
$Ssk2$	1.1738×10^{-8}	x_{24}			

Further, we will assign model variables to all the auxiliary variables involved in the system, as given in Table 7.3, with the definition and conditional values of the variable. The stress signal $S(t)$, intracellular oxidative stress XS^{In} and critical threshold value XS^{C} present in the system varies based on conditions shown in Table 7.3.

Table 7.3. Auxiliary variables involved in the oxidative stress response in the *C. albicans* pathway

Variable	Definition and conditional values	Details
Stress signal	$S(t) = \begin{cases} S_1, & if\ t = t_1 \\ 0, & if\ else \end{cases}$	Stress signal at time t_1. Model variable as x_{45}
XS^{In}	$XS^{In}(t) = \begin{cases} {H_2O_2}^{In}(t) - 10^{-9}, & if\ > 0 \\ 0, & if\ else \end{cases}$	$10^{-9}M$ is the steady state value of ${H_2O_2}^{In}$. Model variable as x_{46}.
XS^{C}	$XS^{*}(t) = \begin{cases} {H_2O_2}^{In}(t) - 3.2 \text{ x } 10^{-5}, if\ > 0 \\ 0, & if\ else \end{cases}$	$3.2 \text{ x } 10^{-5}M$ is the ${H_2O_2}^{In}$ threshold value. Model variable as x_{47}.

Now consider all the species denoted by $\tilde{N} = x_0, x_1, x_2, \ldots\ldots x_{50}, x_{51}$, then for $ISP\ LAS$ the integrated model of (Komalapriya et al., 2015) is redesigned as network of reactions using model variable as shown in Figure 7.1, where the auxiliary variables (see Table 7.3) are denoted by x_{45}, x_{46}, x_{47} and the others by x_{48}, x_{49}, x_{50}, x_{51} (see Table 7.2).

The presence of the stress signal S, (denoted by green circle in Figure 7.1) triggers the following reactions (see Table 7.4) in the network. The types of reaction (*slow* and *fast*) triggered are defined by the kinetic parameters (see Table 7.5), which are responsible for changes in the concentration of the biochemical species. There are 76-biochemical reactions assigned to the modules (see Table 7.1) and the remaining 45 reactions describe the decay rates of the 45 biochemical species. In addition to the reactions shown in Figure 7.1, all the species go through first order decay.

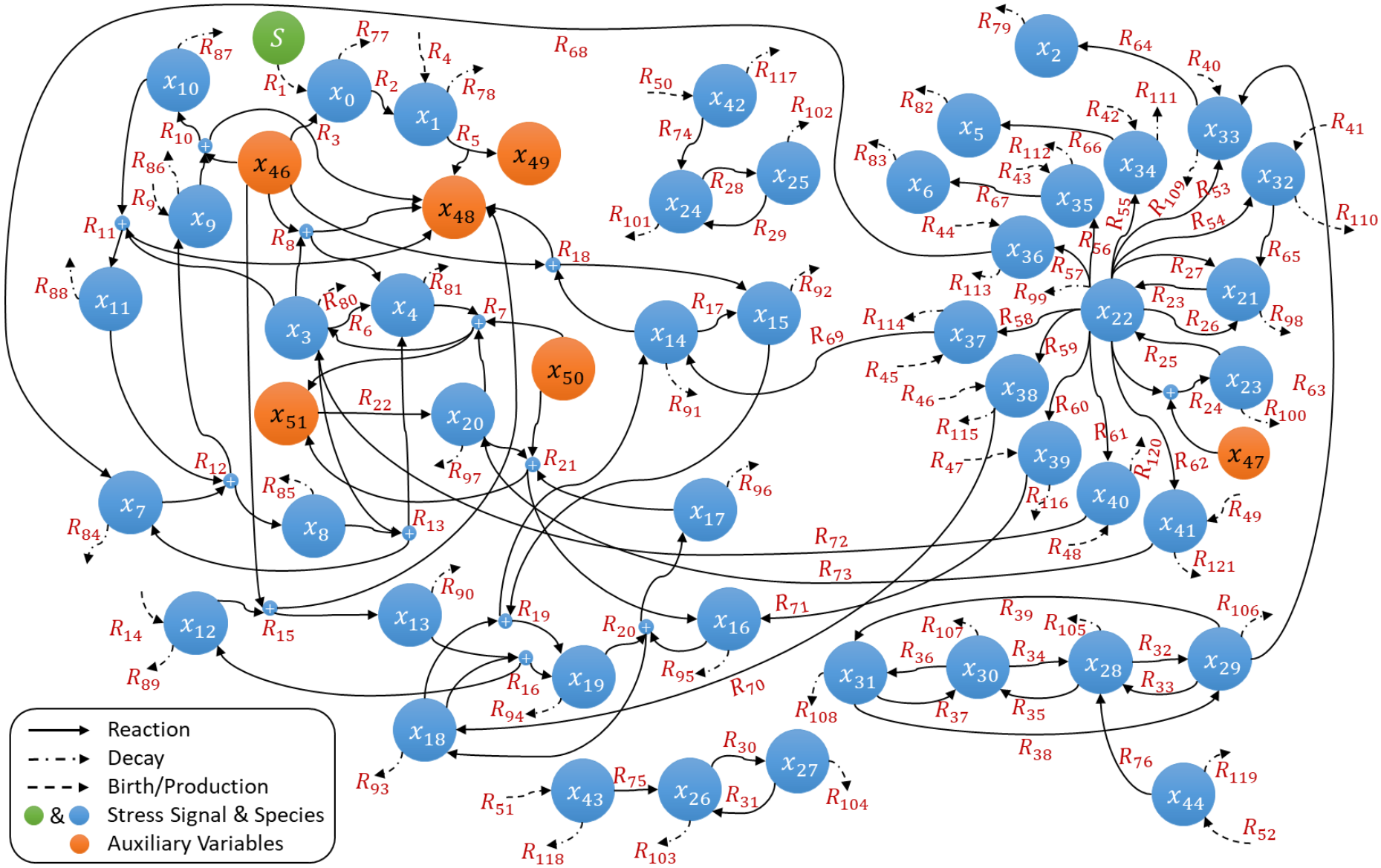


Figure 7.1. Network of reactions involving 45 species, 6 auxiliary variables of the model of the adaptation to oxidative stress response in *C. albicans* (fungal pathogen) pathway using model variables for *ISP LAS*

Table 7.4. Biochemical reactions involved in oxidative stress response in the *C. albicans* pathway

$\boldsymbol{R_1}$: $\star \xrightarrow{k_1} x_0$	$\boldsymbol{R_{61}}$: $x_{22} \xrightarrow{k_{61}} x_{40}$
$\boldsymbol{R_2}$: $x_0 \xrightarrow{k_2} x_1$	$\boldsymbol{R_{62}}$: $x_{22} \xrightarrow{k_{62}} x_{41}$
$\boldsymbol{R_3}$: $x_{46} \xrightarrow{k_3} x_0$	$\boldsymbol{R_{63}}$: $x_{29} \xrightarrow{k_{63}} x_{33}$
$\boldsymbol{R_4}$: $\star \xrightarrow{k_4} x_1$	$\boldsymbol{R_{64}}$: $x_{33} \xrightarrow{k_{64}} x_2$
$\boldsymbol{R_5}$: $2x_1 \xrightarrow{k_5} 2x_{48} + x_{49}$	$\boldsymbol{R_{65}}$: $x_{32} \xrightarrow{k_{65}} x_{21}$
$\boldsymbol{R_6}$: $2x_3 \xrightarrow{k_6} x_4$	$\boldsymbol{R_{66}}$: $x_{34} \xrightarrow{k_{66}} x_5$
$\boldsymbol{R_7}$: $x_4 + x_{20} + x_{50} \xrightarrow{k_7} 2x_3 + x_{51}$	$\boldsymbol{R_{67}}$: $x_{35} \xrightarrow{k_{67}} x_6$
$\boldsymbol{R_8}$: $x_{46} + 2x_3 \xrightarrow{k_8} x_{48} + x_4$	$\boldsymbol{R_{68}}$: $x_{36} \xrightarrow{k_{68}} x_7$
$\boldsymbol{R_9}$: $\star \xrightarrow{k_9} x_9$	$\boldsymbol{R_{69}}$: $x_{37} \xrightarrow{k_{69}} x_{14}$
$\boldsymbol{R_{10}}$: $x_9 + x_{46} \xrightarrow{k_{10}} x_{10} + x_{48}$	$\boldsymbol{R_{70}}$: $x_{38} \xrightarrow{k_{70}} x_{18}$
$\boldsymbol{R_{11}}$: $x_{10} + x_3 \xrightarrow{k_{11}} x_{11} + x_{48}$	$\boldsymbol{R_{71}}$: $x_{39} \xrightarrow{k_{71}} x_{16}$
$\boldsymbol{R_{12}}$: $x_{11} + x_7 \xrightarrow{k_{12}} x_9 + x_8$	$\boldsymbol{R_{72}}$: $x_{40} \xrightarrow{k_{72}} x_3$
$\boldsymbol{R_{13}}$: $x_8 + x_3 \xrightarrow{k_{13}} x_7 + x_4$	$\boldsymbol{R_{73}}$: $x_{41} \xrightarrow{k_{73}} x_{20}$
$\boldsymbol{R_{14}}$: $\star \xrightarrow{k_{14}} x_{12}$	$\boldsymbol{R_{74}}$: $x_{42} \xrightarrow{k_{74}} x_{24}$
$\boldsymbol{R_{15}}$: $x_{12} + x_{46} \xrightarrow{k_{15}} x_{13} + 2x_{48}$	$\boldsymbol{R_{75}}$: $x_{43} \xrightarrow{k_{75}} x_{26}$
$\boldsymbol{R_{16}}$: $x_{13} + x_{18} \xrightarrow{k_{16}} x_{12} + x_{19}$	$\boldsymbol{R_{76}}$: $x_{44} \xrightarrow{k_{76}} x_{28}$
$\boldsymbol{R_{17}}$: $x_{14} \xrightarrow{k_{17}} x_{15}$	$\boldsymbol{R_{77}}$: $x_0 \xrightarrow{k_{77}} \phi$
$\boldsymbol{R_{18}}$: $x_{46} + x_{14} \xrightarrow{k_{18}} x_{15} + x_{48}$	$\boldsymbol{R_{78}}$: $x_1 \xrightarrow{k_{78}} \phi$
$\boldsymbol{R_{19}}$: $x_{15} + x_{18} \xrightarrow{k_{19}} x_{14} + x_{19}$	$\boldsymbol{R_{79}}$: $x_2 \xrightarrow{k_{79}} \phi$
$\boldsymbol{R_{20}}$: $x_{19} + x_{16} \xrightarrow{k_{20}} x_{18} + x_{17}$	$\boldsymbol{R_{80}}$: $x_3 \xrightarrow{k_{80}} \phi$
$\boldsymbol{R_{21}}$: $x_{17} + x_{20} + x_{50} \xrightarrow{k_{21}} x_{16} + x_{51}$	$\boldsymbol{R_{81}}$: $x_4 \xrightarrow{k_{81}} \phi$
$\boldsymbol{R_{22}}$: $x_{51} \xrightarrow{k_{22}} x_{20}$	$\boldsymbol{R_{82}}$: $x_5 \xrightarrow{k_{82}} \phi$
$\boldsymbol{R_{23}}$: $x_{21} \xrightarrow{k_{23}} x_{22}$	$\boldsymbol{R_{83}}$: $x_6 \xrightarrow{k_{83}} \phi$
$\boldsymbol{R_{24}}$: $x_{22} \xrightarrow{k_{24}} x_{23}$	$\boldsymbol{R_{84}}$: $x_7 \xrightarrow{k_{84}} \phi$
$\boldsymbol{R_{25}}$: $x_{23} \xrightarrow{k_{25}} x_{22}$	$\boldsymbol{R_{85}}$: $x_8 \xrightarrow{k_{85}} \phi$
$\boldsymbol{R_{26}}$: $x_{22} \xrightarrow{k_{26}} x_{21}$	$\boldsymbol{R_{86}}$: $x_9 \xrightarrow{k_{86}} \phi$
$\boldsymbol{R_{27}}$: $x_{22} \xrightarrow{k_{27}} x_{21}$	$\boldsymbol{R_{87}}$: $x_{10} \xrightarrow{k_{87}} \phi$
$\boldsymbol{R_{28}}$: $x_{24} \xrightarrow{k_{28}} x_{25}$	$\boldsymbol{R_{88}}$: $x_{11} \xrightarrow{k_{88}} \phi$
$\boldsymbol{R_{29}}$: $x_{25} \xrightarrow{k_{29}} x_{24}$	$\boldsymbol{R_{89}}$: $x_{12} \xrightarrow{k_{89}} \phi$
$\boldsymbol{R_{30}}$: $x_{26} \xrightarrow{k_{30}} x_{27}$	$\boldsymbol{R_{90}}$: $x_{13} \xrightarrow{k_{90}} \phi$
$\boldsymbol{R_{31}}$: $x_{27} \xrightarrow{k_{31}} x_{26}$	$\boldsymbol{R_{91}}$: $x_{14} \xrightarrow{k_{91}} \phi$
$\boldsymbol{R_{32}}$: $x_{28} \xrightarrow{k_{32}} x_{29}$	$\boldsymbol{R_{92}}$: $x_{15} \xrightarrow{k_{92}} \phi$
$\boldsymbol{R_{33}}$: $x_{29} \xrightarrow{k_{33}} x_{28}$	$\boldsymbol{R_{93}}$: $x_{18} \xrightarrow{k_{93}} \phi$

$\boldsymbol{R_{34}}: x_{30} \xrightarrow{k_{34}} x_{28}$	$\boldsymbol{R_{94}}: x_{19} \xrightarrow{k_{94}} \phi$
$\boldsymbol{R_{35}}: x_{28} \xrightarrow{k_{35}} x_{30}$	$\boldsymbol{R_{95}}: x_{16} \xrightarrow{k_{95}} \phi$
$\boldsymbol{R_{36}}: x_{30} \xrightarrow{k_{36}} x_{31}$	$\boldsymbol{R_{96}}: x_{17} \xrightarrow{k_{96}} \phi$
$\boldsymbol{R_{37}}: x_{31} \xrightarrow{k_{37}} x_{30}$	$\boldsymbol{R_{97}}: x_{20} \xrightarrow{k_{97}} \phi$
$\boldsymbol{R_{38}}: x_{31} \xrightarrow{k_{38}} x_{29}$	$\boldsymbol{R_{98}}: x_{21} \xrightarrow{k_{98}} \phi$
$\boldsymbol{R_{39}}: x_{29} \xrightarrow{k_{39}} x_{31}$	$\boldsymbol{R_{99}}: x_{22} \xrightarrow{k_{99}} \phi$
$\boldsymbol{R_{40}}: \star \xrightarrow{k_{40}} x_{33}$	$\boldsymbol{R_{100}}: x_{23} \xrightarrow{k_{100}} \phi$
$\boldsymbol{R_{41}}: \star \xrightarrow{k_{41}} x_{32}$	$\boldsymbol{R_{101}}: x_{24} \xrightarrow{k_{101}} \phi$
$\boldsymbol{R_{42}}: \star \xrightarrow{k_{42}} x_{34}$	$\boldsymbol{R_{102}}: x_{25} \xrightarrow{k_{102}} \phi$
$\boldsymbol{R_{43}}: \star \xrightarrow{k_{43}} x_{35}$	$\boldsymbol{R_{103}}: x_{26} \xrightarrow{k_{103}} \phi$
$\boldsymbol{R_{44}}: \star \xrightarrow{k_{44}} x_{36}$	$\boldsymbol{R_{104}}: x_{27} \xrightarrow{k_{104}} \phi$
$\boldsymbol{R_{45}}: \star \xrightarrow{k_{45}} x_{37}$	$\boldsymbol{R_{105}}: x_{28} \xrightarrow{k_{105}} \phi$
$\boldsymbol{R_{46}}: \star \xrightarrow{k_{46}} x_{38}$	$\boldsymbol{R_{106}}: x_{29} \xrightarrow{k_{106}} \phi$
$\boldsymbol{R_{47}}: \star \xrightarrow{k_{47}} x_{39}$	$\boldsymbol{R_{107}}: x_{30} \xrightarrow{k_{107}} \phi$
$\boldsymbol{R_{48}}: \star \xrightarrow{k_{48}} x_{40}$	$\boldsymbol{R_{108}}: x_{31} \xrightarrow{k_{108}} \phi$
$\boldsymbol{R_{49}}: \star \xrightarrow{k_{49}} x_{41}$	$\boldsymbol{R_{109}}: x_{33} \xrightarrow{k_{109}} \phi$
$\boldsymbol{R_{50}}: \star \xrightarrow{k_{50}} x_{42}$	$\boldsymbol{R_{110}}: x_{32} \xrightarrow{k_{110}} \phi$
$\boldsymbol{R_{51}}: \star \xrightarrow{k_{51}} x_{43}$	$\boldsymbol{R_{111}}: x_{34} \xrightarrow{k_{111}} \phi$
$\boldsymbol{R_{52}}: \star \xrightarrow{k_{52}} x_{44}$	$\boldsymbol{R_{112}}: x_{35} \xrightarrow{k_{112}} \phi$
$\boldsymbol{R_{53}}: x_{22} \xrightarrow{k_{53}} x_{33}$	$\boldsymbol{R_{113}}: x_{36} \xrightarrow{k_{113}} \phi$
$\boldsymbol{R_{54}}: x_{22} \xrightarrow{k_{54}} x_{32}$	$\boldsymbol{R_{114}}: x_{37} \xrightarrow{k_{114}} \phi$
$\boldsymbol{R_{55}}: x_{22} \xrightarrow{k_{55}} x_{34}$	$\boldsymbol{R_{115}}: x_{38} \xrightarrow{k_{115}} \phi$
$\boldsymbol{R_{56}}: x_{22} \xrightarrow{k_{56}} x_{35}$	$\boldsymbol{R_{116}}: x_{39} \xrightarrow{k_{116}} \phi$
$\boldsymbol{R_{57}}: x_{22} \xrightarrow{k_{57}} x_{36}$	$\boldsymbol{R_{117}}: x_{42} \xrightarrow{k_{117}} \phi$
$\boldsymbol{R_{58}}: x_{22} \xrightarrow{k_{58}} x_{37}$	$\boldsymbol{R_{118}}: x_{43} \xrightarrow{k_{118}} \phi$
$\boldsymbol{R_{59}}: x_{22} \xrightarrow{k_{59}} x_{38}$	$\boldsymbol{R_{119}}: x_{44} \xrightarrow{k_{119}} \phi$
$\boldsymbol{R_{60}}: x_{22} \xrightarrow{k_{60}} x_{39}$	$\boldsymbol{R_{120}}: x_{40} \xrightarrow{k_{120}} \phi$
	$\boldsymbol{R_{121}}: x_{41} \xrightarrow{k_{121}} \phi$

Table 7.5. Kinetic parameter values for the biochemical reactions involved in the oxidative stress response in the *C. albicans* pathway.

Parameter	Value	Parameter	Value	Parameter	Value
k_1	Stress signal	k_{41}	5.3601×10^{-14}	k_{81}	1.2836×10^{-4}
k_2	3.5×10^{-5}	k_{42}	2.3805×10^{-13}	k_{82}	1.4991×10^{-4}
k_3	3.5×10^{-5}	k_{43}	1.0792×10^{-13}	k_{83}	3.3007×10^{-4}
k_4	2.1928×10^{-7}	k_{44}	4.4390×10^{-13}	k_{84}	1.0394×10^{-4}
k_5	3.4×10^{7}	k_{45}	1.8694×10^{-12}	k_{85}	1.0394×10^{-4}
k_6	2.4266×10^{-2}	k_{46}	5.4929×10^{-13}	k_{86}	2.6866×10^{-4}
k_7	9×10^{-1}	k_{47}	3.0006×10^{-13}	k_{87}	2.6866×10^{-4}
k_8	578	k_{48}	2.9619×10^{-13}	k_{88}	2.6866×10^{-4}
k_9	3.2777×10^{-8}	k_{49}	2.9619×10^{-13}	k_{89}	2.6866×10^{-4}
k_{10}	1×10^{4}	k_{50}	1.0161×10^{-13}	k_{90}	2.6866×10^{-4}
k_{11}	1.2×10^{5}	k_{51}	8.2790×10^{-14}	k_{91}	9.8793×10^{-5}
k_{12}	9.1×10^{4}	k_{52}	1.7012×10^{-13}	k_{92}	9.8793×10^{-5}
k_{13}	3.7×10^{4}	k_{53}	4.2398×10^{-11}	k_{93}	2.0111×10^{-4}
k_{14}	2.6866×10^{-7}	k_{54}	2.9009×10^{-12}	k_{94}	2.0111×10^{-4}
k_{15}	5×10^{1}	k_{55}	5.1062×10^{-11}	k_{95}	3.1484×10^{-4}
k_{16}	1×10^{5}	k_{56}	2.0353×10^{-11}	k_{96}	3.1484×10^{-4}
k_{17}	0	k_{57}	5.1271×10^{-11}	k_{97}	1.155×10^{-4}
k_{18}	4×10^{7}	k_{58}	2.8041×10^{-10}	k_{98}	5.0228×10^{-4}
k_{19}	1×10^{1}	k_{59}	3.9549×10^{-11}	k_{99}	5.0228×10^{-4}
k_{20}	2×10^{1}	k_{60}	6.8863×10^{-11}	k_{100}	5.0228×10^{-4}
k_{21}	1×10^{2}	k_{61}	1.9252×10^{-11}	k_{101}	2.6866×10^{-4}
k_{22}	2.9619×10^{-8}	k_{62}	2.3695×10^{-11}	k_{102}	2.6866×10^{-4}
k_{23}	2.3258×10^{6}	k_{63}	4.2398×10^{-11}	k_{103}	3.2090×10^{-4}
k_{24}	1.1629×10^{9}	k_{64}	6.4929	k_{104}	3.2090×10^{-4}
k_{25}	5×10^{-4}	k_{65}	7.0867×10^{-1}	k_{105}	4.2317×10^{-5}
k_{26}	5	k_{66}	9.5085×10^{-2}	k_{106}	4.2317×10^{-5}
k_{27}	1×10^{3}	k_{67}	3.1983×10^{-1}	k_{107}	4.2317×10^{-5}
k_{28}	1.1629×10^{4}	k_{68}	3.2786×10^{-1}	k_{108}	4.2317×10^{-5}
k_{29}	1.359×10^{-2}	k_{69}	4.2782×10^{-1}	k_{109}	8.2518×10^{-4}
k_{30}	2.19×10^{6}	k_{70}	1.4314×10^{-1}	k_{110}	1.0858×10^{-3}
k_{31}	2.007×10^{-2}	k_{71}	1.0644×10^{1}	k_{111}	8.7358×10^{-4}
k_{32}	6.493×10^{6}	k_{72}	1.9626×10^{3}	k_{112}	3.2001×10^{-4}
k_{33}	5.906×10^{-2}	k_{73}	6.5801×10^{2}	k_{113}	5.4743×10^{-4}
k_{34}	5×10^{-5}	k_{74}	3.7541×10^{-2}	k_{114}	4.6448×10^{-4}
k_{35}	5×10^{2}	k_{75}	6.1608×10^{-1}	k_{115}	3.3942×10^{-4}
k_{36}	6.493×10^{6}	k_{76}	3.9053×10^{-2}	k_{116}	8.0689×10^{-4}
k_{37}	5.906×10^{-2}	k_{77}	3.8508×10^{-5}	k_{117}	1.2097×10^{-3}
k_{38}	6.493×10^{6}	k_{78}	3.8508×10^{-5}	k_{118}	1.7115×10^{-3}
k_{39}	5.906×10^{-2}	k_{79}	3.6137×10^{-4}	k_{119}	5.3858×10^{-4}
k_{40}	2.9619×10^{-13}	k_{80}	1.2836×10^{-4}	k_{120}	8.2518×10^{-4}
				k_{121}	8.2518×10^{-4}

The initial condition values in Table 7.2 and Table 7.3 were available in the form of concentrations; therefore, we will calculate the number of molecules for all species according to the following formula:

$$Molecules = concentration \times A_V \times V_{os}, \quad (102)$$

where A_V is the Avogadro constant (6.022140857×10^{23} mol^{-1}) and V_{os} is the active volume (osmotically) obtained from the solid base volume V_b (41%) and total volume V_t of the *C. albicans* cell (Schaber et al., 2012). All the constants related to the *C. albicans* cells used in the model integration are given in Table 7.6.

Table 7.6. Constants involved in the oxidative stress response in the *C. albicans* pathway.

Constant	Value	Details
A_V	6.023×10^{23}	Avogadro number
a	1×10^{-3}	Factor of the volume conversion
V_t	6.545×10^{-14} L	Total volume of *C. albicans* cells having a radius of $5\mu m$
V_b	2.683×10^{-14} L	Solid base volume of *C. albicans* cells (Schaber et al., 2012)
V_{os}	3.862×10^{-14} L	Active volume (osmotically)
V_m	2.805×10^{-11} L	Media volume $726.74 \times V_{os}$
A	$7.854 \times 10^{-7}\ cm^2$	Cells surface area of *C. albicans*

These protein species counts and reaction propensities will be used to define the state-space of the model for $ISP\ LAS$ and will be responsible for changes in the number of states. Therefore, we track the copy counts of these species as:

$$([Chemical\ species]) \in \tilde{\mathrm{N}} \equiv (Model\ variables), \quad (103)$$

for all the modules in Table 7.1. From section 7.2.1 to section 7.2.4, we will discuss about the nature of the reactions and transitions in different modules and their integration in $ISP\ LAS$.

7.2.1 Transporter Module

In reactions R_1 and R_3, the copy counts of x_0 (extra-cellular H_2O_2) increased in the presence of stress signal (S) and intra-cellular oxidative stress respectively. R_3 also respresents the diffusion of x_1 (intra-cellular hydrogen peroxide) into the medium. The intra-cellular oxidative stress x_{46} is given by $x_{46} = x_1 - 10^{-9}$ (see Table 7.2), where, under normal conditions, the steady state value is 10^{-9}. Reaction R_3 is modelled as the diffusion of x_1 (intra-cellular H_2O_2) based on the surface area of the cell. In reaction R_2, x_0 is diffused into the cytosol and, similarly, the reaction is modelled as the diffusion of x_0 (extra-cellular H_2O_2) based on the surface area of the cell. Reaction R_4 is a zero-order mass action reaction that represents x_1 production that maintains the steady state value of x_1. In reaction R_5, the excessive x_1 (intra-cellular hydrogen peroxide) is detoxified by x_2 (catalase) enzymatically and the reaction is based on mass action kinetics. Reactions R_{77} and R_{78}, represent the decay rates of x_0 (extra-cellular) and x_1 (intracellular) hydrogen peroxide biochemical species.

It is important to track the number of copies of the species through these reactions. Therefore, we note the changes in the counts of all the biochemical species (see transporter module in Table 7.1) and define the transitions associated with R_M (Table 7.4) in the stoichiometric vector V_M, matrix for all the Ñ species involved in the transporter module reactions. Based on interactions of the species in Figure 7.1, we have R_M channels {1, 2, 3, 4, 5, 77, 78} and Ñ species - x_0, x_1, x_{46}, x_{48} and x_{49} through the given number of reaction channels. The transitions associated with $R_{M=1,2,3,4,5,77,78}$ and Ñ in stoichiometric vector matrix is given as:

$$V_M = \begin{pmatrix} (1, 0, 0, 0, 0) \\ (-1, 1, 0, 0, 0) \\ (1, 0, -1, 0, 0) \\ (0, 1, 0, 0, 0) \\ (0, -1, 0, 1, 1) \\ (-1, 0, 0, 0, 0) \\ (0, -1, 0, 0, 0) \end{pmatrix}$$

The associated Markov chain of the transporter module network is treated as a Markov chain graph as discussed in section 3.3, and the states in terms of the nodes with additional information such as the number of R_M reactions required to reach the state. In the equivalent growing graph tree, the transition between the nodes is defined by the typical form of the growing dictionary of *transporter module*. Further, we express the propensity functions of all the $R_{M=1,2,3,4,5,77,78}$ reactions (see Table 7.4) involved in the model for selective M as:

$$R_1: \quad a_1([x_0],[x_1],[x_{46}],[x_{48}],[x_{49}]) = S(t) \tag{104}$$

$$R_2: \quad a_2([x_0],[x_1],[x_{46}],[x_{48}],[x_{49}]) = k_2 \times x_0(t) \tag{105}$$

$$R_3: \quad a_3([x_0],[x_1],[x_{46}],[x_{48}],[x_{49}]) = k_3 \times x_{46}(t) \tag{106}$$

$$R_4: \quad a_4([x_0],[x_1],[x_{46}],[x_{48}],[x_{49}]) = k_4 \tag{107}$$

$$R_5: \quad a_5([x_0],[x_1],[x_{46}],[x_{48}],[x_{49}]) = k_5 \times x_1(t) \times x_1(t) \tag{108}$$

$$R_{77}: \quad a_{77}([x_0],[x_1],[x_{46}],[x_{48}],[x_{49}]) = k_{77} \times x_0(t) \tag{109}$$

$$R_{78}: \quad a_{78}([x_0],[x_1],[x_{46}],[x_{48}],[x_{49}]) = k_{78} \times x_1(t) \tag{110}$$

The associated Markov chain graph can now be used to visualise the Markov process of the transporter module with $\mathbf{n}_J = (\mathbf{X}_K, \mathrm{d}_l)$ number of states stored in all the nodes and directly reachable from the initial node $N_1 = (X_0, \mathrm{d}_{l=1})$ via R_M reactions. The transition weightage between the different states in transporter module is defined by their propensity values at a particular time in the process. In next section 7.2.2, we will discuss about the nature of the reactions and transitions in *Antioxidant module* and its integration in $ISP\ LAS$.

7.2.2 Antioxidant Module

Catalase specifies the H_2O_2 detoxification pathway through the enzyme catalase. Here, reaction R_{40} is a zero-order reaction that produces x_{33} ($CAT1\ mRNA$) at basal rates, whereas, the mass action law based reaction R_{64}, translate this x_{33} ($CAT1\ mRNA$) into x_2 ($Cat1$). Lastly, reactions R_{79} and R_{109} represents the first order decay of species x_2 and x_{33}, respectively, in catalase.

During oxidative stress, *Thioredoxin* is responsible for repairing protein di-thiols and this demonstrates their role in the elimination of $\mathrm{H_2O_2}$. Here, reaction R_{16} is based on the law of mass action, where x_{18} (local thioredoxin) is reduced by x_{13} (protein di-sulphides) and this increased the counts of x_{12} and x_{19}. Reaction R_{17} is a cellular redox reaction based on the law of mass action, responsible for the oxidation of x_{14} (peroxiredoxin), and increase the copy counts of oxidised x_{15}. Reaction R_{18} is based on the law of mass action where excessive x_1 (intra-cellular hydrogen peroxide) is detoxified by x_{14} to produce the oxidised x_{15}. In reaction R_{19}, x_{15} (oxidised peroxiredoxin) is reduced by the action of x_{18} to produce x_{19} (oxidised thioredoxin) and x_{14} (reduced peroxiredoxin). In reaction R_{20}, x_{19} (oxidised thioredoxin) is reduced by x_{16} (thioredoxin reductase) to produce x_{17} (oxidised thioredoxin reductase) and

x_{18}. In reaction R_{21}, x_{17} (oxidised thioredoxin reductase) is reduced by x_{20} ($NADPH$) to produce x_{16} (thioredoxin reductase) and release the $NADP^{+}$. In reactions R_{69}, R_{70} and R_{71}, species x_{37}, x_{38} and x_{39} are translated and it is assumed (Komalapriya et al., 2015) that this produces reduced forms of x_{14}, x_{18} and x_{16} proteins. Reactions R_{89}, R_{90}, R_{91}, R_{92}, R_{93}, R_{94}, R_{95}, R_{96}, R_{97}, R_{114}, R_{115} and R_{116} represent the first order decay of x_{12}, x_{13} (protein di-sulphides), x_{14} (reduced peroxiredoxin), x_{15}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}, x_{37}, x_{38}, x_{39}.

The *pentose phosphate pathway* (PPP) keeps the track of the maintenance of $NADPH$ under different conditions (stress and non-stress). Reaction R_7 is modelled based on the Michaelis-Menten reaction (see C.2 of Appendix C) where x_4 is reduced by enzyme x_{20}. Reaction R_{21} is again considered in this pathway as x_{20} ($NADPH$) and plays a role in reducing x_{17} (oxidised thioredoxin reductase). In reaction R_{22}, the metabolic flux is redirected under oxidative stress to produce x_{20}. This induction of x_{20} is done terms of Hill function (Santillán et al., 2008). Reaction R_{73} is based on the law of mass action that translate the x_{41} (hypothetical) into x_{20}. The last reaction R_{97} involved in PPP represents the first order decay of x_{20} ($NADPH$).

It is important to track the number of copies of the biochemical species through the reactions in the *catalase*, *thioredoxin* and *pentose phosphate pathway*. Therefore, we note the changes in the counts of critical biochemical species (see the antioxidant module in Table 7.1) and define the transitions associated with R_M (associated with *antioxidant module*) in the stoichiometric vector V_M matrix for all Ñ species involved in the *catalase*, *thioredoxin* and *pentose phosphate pathway* that defines the antioxidant module. Based on interactions of the species in Figure 7.1, we have:

1) R_M channels for *catalase* as {40, 64, 79, 109} and Ñ species - x_2, x_{33} through the given number of reaction channels
2) R_M channels for *thioredoxin* as {16, 17, 18, 19, 20, 21, 69, 70, 71, 89, 90, 91, 92, 93, 94, 95, 96, 97, 114, 115, 116} and Ñ species - x_{12}, x_{13}, x_{14}, x_{15}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}, x_{37}, x_{38}, x_{39}, x_{46} through the given number of reaction channels
3) R_M channels for *pentose phosphate pathway* as {7, 21, 22, 73, 97} and Ñ species - x_4, x_{20}, x_{50}, x_{51}, x_{17}, x_{41} through given number of reaction channels.

To examine the stiffness of the modules it is important to consider the stoichiometric vectors individually for the *catalase*, *thioredoxin* and *pentose phosphate pathway* within the

antioxidant module as they define the sub-modules of the Integrated model and contribute to the different number of states in the domain.

The transitions associated with R_M channels and Ñ in the stoichiometric vector matrix for *catalase* are given as:

$$\text{for } catalase,\ V_M = \begin{bmatrix} (0, 1) \\ (1, 1) \\ (-1, 0) \\ (0, 1) \end{bmatrix}$$

The transitions associated with R_M channels and Ñ in the stoichiometric vector matrix for *thioredoxin* is given as:

$$\text{for } thioredoxin,\ V_M = \begin{pmatrix} (1, -1, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0) \\ (0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, -1) \\ (0, 0, 1, -1, 0, 0, -1, 1, 0, 0, 0, 0, 0) \\ (0, 0, 0, 0, -1, 1, 1, -1, 0, 0, 0, 0, 0) \\ (0, 0, 0, 0, 1, -1, 0, 0, -1, 0, 0, 0, 0) \\ (0, 0, 1, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0) \\ (0, 0, 0, 0, 0, 0, 1, 0, 0, 0, -1, 0, 0) \\ (0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, -1, 0) \\ (-1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0) \\ (0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0) \\ (0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0) \\ (0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0) \\ (0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0) \\ (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0) \\ (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0) \end{pmatrix}$$

The transitions associated with R_M channels and Ñ in the stoichiometric vector matrix for *pentose phosphate pathway* are given as:

$$\text{for } PPP,\ V_M = \begin{pmatrix} (-1, -1, -1, 1, 0, 0) \\ (0, -1, -1, 1, -1, 1) \\ (0,\ 1,\ 0, -1, 0, 0) \\ (0,\ 1,\ 0, 0, 0, -1) \\ (0, -1,\ 0, 0, 0,\ 0) \end{pmatrix}$$

The associated assembly of Markov chains of the *catalase*, *thioredoxin* and *Pentose phosphate pathway* network is treated as a Markov chain graph for antioxidant module, as shown in Figure 7.1, and states in terms of the nodes with additional information such as the number of R_M reactions required to reach the state. In the equivalent growing graph tree, the transition between the nodes is defined by the typical form of the growing dictionary.

Further, we express the propensity functions of all the $R_{M=40,64,79,109}$, $R_{M=7,21,22,73,97}$ and $R_{M=16,17,18,19,20,21,69,70,71,89,90,91,92,93,94,95,96,97,114,115,116}$ reactions (see Table 7.4) involved in the *catalase*, *thioredoxin* and *pentose phosphate pathway*, respectively for selective M as:

$$R_M: \quad a_M([\tilde{\text{N}}\ species\ vector\ in\ Catalase]) = k_M([Product\ of\ reactant\ species]) \tag{111}$$

$$R_M: \quad a_M([\tilde{\text{N}}\ species\ vector\ in\ Thioredoxin]) = k_M([Product\ of\ reactant\ species]) \tag{112}$$

$$R_M: \quad a_M([\tilde{\text{N}}\ species\ vector\ in\ PPP]) = k_M([Product\ of\ reactant\ species]) \tag{113}$$

The associated Markov chain graph can now be used to represent the Markov process of the antioxidant module with $\mathbf{n}_J = (\mathbf{X}_K, \mathrm{d}_l)$ number of states stored in all the nodes directly reachable from the initial node $N_1 = (X_0, \mathrm{d}_{l=1})$ via R_M reactions. In section 7.3, we will discuss the results of the computational experiment and visualise the state-space and conditional probabilities of the species. The transition weightage between different states in the antioxidant module is defined by the propensity values at a particular time in the three processes (the *catalase*, *thioredoxin* and *pentose phosphate pathways*). In the next section 7.2.3, we will discuss the nature of reactions and transitions in the *protein-thiol module* and its integration in $ISP\ LAS$.

7.2.3 Protein-thiol Module

During oxidative stress the, *protein-thiol module* tracks the damage and repairs the protein mono-thiols. Here, reaction R_9 is a zero-order mass action reaction responsible for production of x_9 (protein mono-thiols). In reaction R_{10}, through x_{46} (excessive intra-cellular hydrogen peroxide) x_9 (protein mono-thiols) is oxidised to form x_{10} (protein sulfenic acid). Further in reaction R_{11}, this x_{10} (protein sulfenic acid) is translated to x_{11} (disulphide) through S-glutathionylation. This reaction is also based on the law of mass action. In reaction R_{12}, x_{11} (disulphide) de-glutathionylation takes place through the action of x_7 (glutaredoxin) to produce x_9 (protein mono-thiols). Reaction R_{14} is a zero-order mass action reaction that produces x_{12} (protein di-thiols). Further, in reaction R_{15}, this x_{12} (protein di-thiols) is oxidised by x_{46} (excessive intra-cellular hydrogen peroxide) to produce x_{13} (protein disulphide). In reaction R_{16}, x_{13} (protein disulphide) is reduced by x_{18} (thioredoxin) to produce x_{12} (protein di-thiols) and x_{19} (oxidised thioredoxin). In further, reactions R_{86}, R_{87}, R_{88}, R_{89}, R_{90} represent the first order decay of the biochemical species x_9, x_{10}, x_{11}, x_{12}, x_{13}, respectively.

It is important to track the number of copies of the species through these reactions. Therefore, we note the changes in the counts for of all the biochemical species (see *protein-thiol* in Table 7.1) and define the transitions associated with R_M (associated with *protein-thiol module*) in stoichiometric vector V_M matrix for all Ñ species involved in the *protein-thiol module's* reactions. Based on interactions of the species in Figure 7.1, we have R_M channels {9, 10, 11, 12, 14, 15, 16, 86, 87, 88, 89, 90} and Ñ species - x_9, x_{46}, x_{10}, x_{11}, x_7, x_3, x_{12}, x_{13} and x_{18} through the given number of reaction channels. The transitions associated with $R_{M=9,10,11,12,14,15,16,86,87,88,89,90}$ and Ñ in stoichiometric vector matrix is given as:

$$V_M = \begin{pmatrix} (1, 0, 0, 0, 0, 0, 0, 0, 0) \\ (-1, -1, 1, 0, 0, 0, 0, 0, 0) \\ (0, 0, -1, 1, 0, -1, 0, 0, 0) \\ (1, 0, 0, -1, -1, 0, 0, 0, 0) \\ (0, 0, 0, 0, 0, 0, 1, 0, 0) \\ (0, -1, 0, 0, 0, 0, -1, 1, 0) \\ (0, 0, 0, 0, 0, 0, 1, -1, -1) \\ (-1, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, 0, -1, 0, 0, 0, 0, 0, 0) \\ (0, 0, 0, -1, 0, 0, 0, 0, 0) \\ (0, 0, 0, 0, 0, 0, -1, 0, 0) \\ (0, 0, 0, 0, 0, 0, 0, -1, 0) \end{pmatrix}$$

The associated Markov chain of the *protein-thiol* module network is treated as a Markov chain graph and states in terms of the nodes with additional information, such as the number of R_M reactions required to reach the state. In the equivalent growing graph tree, the transition between the nodes is defined by the typical form of the growing dictionary.

Further, we express the propensity functions of all the $R_{M=9,10,11,12,14,15,16,86,87,88,89,90}$ reactions (see Table 7.4) involved in *protein-thiols* for the selected M as:

$$R_M\colon\ a_M([\tilde{\mathrm{N}}\ species\ vector\ in\ Protein-thiol]) = k_M([Product\ of\ reactant\ species]) \quad (114)$$

An associated Markov chain graph can now be used to represent the Markovian process of the *protein-thiol module* with $\mathbf{n}_J = (\mathbf{X}_K, \mathrm{d}_l)$ number of states stored in all the nodes directly reachable from the initial node, $N_1 = (X_0, \mathrm{d}_{l=1})$, via R_M reactions. In section 7.3, we will discuss the results of the computational experiments and visualise the state-space and conditional probabilities of the species. The transition weightage between the different states in the *protein-thiol module* is defined by the propensity value at a particular time in the process. In next section 7.2.4, we discuss the nature of the reactions and transitions in the *signalling and gene expression module* and its integration in $ISP\ LAS$.

7.2.4 Signalling and Gene Expression Module

This module is a combination of two major pathways - *cap1* and *hog1 pathways*, which are responsible for the major dynamics of the system. During oxidative stress, they describe the dynamics of the protein kinase pathway as well as activation of factors like AP-1 transcription and antioxidant gene regulation under different conditions (stress and non-stress).

In the *cap1 pathway*, reaction R_{23} is based on the law of mass action and is responsible for the oxidation and activation of x_{21} (native transcription) into x_{22}. Under normal conditions it is assumed that no x_{22} exists in the oxidised state (active) and, hence, it is considered as 0 initially. Reaction R_{24} is based on Hill type kinetics, where during a high value of x_{47}, the activated x_{22} is converted into its inactive form, x_{23}. Under normal conditions it is assumed that no x_{22} exists in oxidised state (in-active) and hence considered as 0 initially. Reaction R_{25} is based on the law of mass action, where x_{23} is converted to x_{22}. Further, reaction R_{26} organises the reduction of x_{22} into x_{21} (native transcription). This reduction is also moderated by x_{18} (native thioredoxin). However, it is not consumed during the reaction but used as a conditioner for the reduction. In reactions R_{53}, R_{54}, R_{55}, R_{56}, R_{57}, R_{58}, R_{59}, R_{60}, R_{61}, R_{62}, the active x_{22} induces x_{33}, x_{32}, x_{34}, x_{35}, x_{36}, x_{37}, x_{38}, x_{39}, x_{40}, x_{41} respectively, and the

reactions are based on the Hill function (Santillán et al., 2008). In reaction R_{65}, x_{32} is translated to produce the x_{21} protein (native form). This reaction is also based on the law of mass action. Reactions R_{98}, R_{99}, R_{100} represent the first order decay of x_{21} (native), x_{22} (active) and x_{23} (in-active) proteins.

In the *hog1 pathway*, reaction R_{28} is based on the law of mass action and is responsible for phosphorylation, as well as activation, of x_{25}. Under normal conditions it is assumed that no of the x_{25} exists in the phosphorylated form (active) and, hence, it is considered as 0 initially. Further, reaction R_{29}, carries out de-phosphorylation as well as de-activation of x_{24} via phosphatases. Reaction R_{30} organises the phosphorylation, as well as activation of x_{27}. This activation is also moderated by x_{25}; however, this is not consumed during the reaction but used as a conditioner for activation. Further, reaction R_{31} carries out de-phosphorylation, as well as de-activation of x_{26} via phosphatases. Reaction R_{32} organises phosphorylation as well as the activation of native x_{28}. This activation is also moderated by x_{27}; however, it did not consumed during the reaction but used as a conditioner for activation. Further, reaction R_{33} carries out de-phosphorylation, as well as de-activation of x_{29} via phosphatases. In reaction R_{34}, the x_{30} oxidised, de-phosphorylated form (in-active) is converted to x_{28}. This conversion is supported by x_{18}, which does not change itself into the x_{19} (oxidised thioredoxin form) during this procedure. Reaction R_{35} is the backward reaction of R_{34} where, under oxidative stress, x_{28} (native) inactivation takes place and, in this process, x_{46} is used as a conditioner. In reaction R_{36}, the phosphorylation of x_{30} oxidised de-phosphorylated form (in-active) takes place. Here, x_{27} is used as a conditioner and is not consumed during the procedure. Reaction R_{37} is the backward reaction of R_{36}, where de-phosphorylation of x_{31} takes place via phosphatases. In reaction R_{38}, x_{31} is converted to x_{29} when moderated by x_{18} (native thioredoxin) and is not consumed in the procedure. Reaction R_{39} is the backward reaction of R_{38} where x_{29} is, again, converted to x_{31} during oxidative stress. Lastly, reactions R_{74}, R_{75}, R_{76} translate the x_{42}, x_{43}, x_{44} into x_{24}, x_{26}, x_{28}, respectively. It is supposed that these translations only result in the un-phosphorylated forms of x_{24}, x_{26}, x_{28}. All the reactions in the *hog1 pathway* are based on the law of mass action.

It is important to track the number of copies of the biochemical species through reactions in the *cap1* and *hog1 pathways*. Therefore, we note the changes in the counts of critical biochemical species (see signalling and gene expression in Table 7.1) and define the transitions associated with R_M (associated with *signalling module*) in the stoichiometric

vector V_M matrix for all the Ñ species involved in the *cap1* and *hog1 pathways* that will define the signalling part of the module. Based on interactions of the species in Figure 7.1, we have:

1) R_M channels for the *cap1 pathway* as {23, 24, 25, 26, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 65, 98, 99, 100} and Ñ species - x_{21}, x_{22}, x_{23}, x_{32}, x_{33}, x_{34}, x_{35}, x_{36}, x_{37}, x_{38}, x_{39}, x_{40}, x_{41}, x_{46}, x_{47} through the given number of reaction channels,
2) R_M channels for the *hog1 pathway* as {28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 74, 75, 76} and Ñ species - x_{24}, x_{25}, x_{26}, x_{27}, x_{28}, x_{29}, x_{30}, x_{31}, x_{42}, x_{43}, x_{44}, x_{46} through the given number of reaction channels.

To examine the stiffness of the module it is important to consider the stoichiometric vectors individually for the *cap1* and *hog1 pathways* within the signalling and gene expression module as they define the sub-modules of the integrated model and contribute different numbers of states in the domain.

The transitions associated with R_M channels and Ñ in stoichiometric vector matrix for the *cap1 pathway* are given as:

$$\text{for } cap1,\ V_M = \begin{pmatrix} (-1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, -1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, -1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, -1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, -1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, -1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0) \\ (0, -1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0) \\ (0, -1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0) \\ (0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0) \\ (0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0) \\ (1, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (-1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \end{pmatrix}$$

The transitions associated with R_M channels and Ñ in the stoichiometric vector matrix for the *hog1 pathway* are given as:

$$\text{for } hog1,\ V_M = \begin{pmatrix} (-1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0) \\ (0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0) \\ (0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0) \\ (0, 0, 0, 0, 1, 0, -1, 0, 0, 0, 0, 0) \\ (0, 0, 0, 0, -1, 0, 1, 0, 0, 0, 0, 0) \\ (0, 0, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0) \\ (0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0) \\ (0, 0, 0, 0, 0, 1, 0, -1, 0, 0, 0, 0) \\ (0, 0, 0, 0, 0, -1, 0, 1, 0, 0, 0, 0) \\ (1, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0) \\ (0, 0, 1, 0, 0, 0, 0, 0, 0, -1, 0, 0) \\ (0, 0, 0, 0, 1, 0, 0, 0, 0, 0, -1, 0) \end{pmatrix}$$

The associated assembly of Markov chains of the *cap1* and *hog1 pathway* networks is treated as a Markov chain graph for the signalling and gene expression module and the states in terms of the nodes, with additional information such as the number of R_M reactions required to reach the state. In the equivalent growing graph tree, the transition between the nodes is defined by the typical form of the growing dictionary.

Further, we express the propensity functions of all $R_{M=23,24,25,26,53,54,55,56,57,58,59,60,61,62,65,98,99,100}$ and $R_{M=28,29,30,31,32,33,34,35,36,37,38,39,74,75,76}$ reactions (see Table 7.4) involved in the *cap1* and *hog1 pathways*, respectively, for a selected M as:

$$R_M:\ a_M([\tilde{\text{N}}\ species\ vector\ in\ Cap1\ pathway]) = k_M([Product\ of\ reactant\ species]) \quad (115)$$

$$R_M:\ a_M([\tilde{\text{N}}\ species\ vector\ in\ Hog1\ pathway]) = k_M([Product\ of\ reactant\ species]) \quad (116)$$

The associated Markov chain graph can now be used to represent the Markov process of the signalling and gene expression module with $\mathbf{n}_J = (\mathbf{X}_K, \eth_l)$ number of states stored in all the nodes directly reachable from the initial node $N_1 = (X_0, \eth_{l=1})$ via R_M reactions. In section 7.3, we will discuss the results of the computational experiment and visualise the state-space and conditional probabilities of the species. The transition weightage between different states in the signalling and gene expression module is defined by the propensity values at particular time in the two processes (*cap1* and *hog1 pathways*).

Combining all the R_M channels, as discussed in sections 7.2.1, 7.2.2, 7.2.3, 7.2.4 of the modules (see Table 7.1), respectively, as {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121} and Ñ species – x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}, x_{13}, x_{14}, x_{15}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}, x_{21}, x_{22}, x_{23}, x_{24}, x_{25}, x_{26}, x_{27}, x_{28}, x_{29}, x_{30}, x_{31}, x_{32}, x_{33}, x_{34}, x_{35}, x_{36}, x_{37}, x_{38}, x_{39}, x_{40}, x_{41}, x_{42}, x_{43}, x_{44}, x_{45}, x_{46}, x_{47}, x_{48}, x_{49}, x_{50}, x_{51} through the given number of reaction channels. The transitions associated with R_M channels and Ñ in stoichiometric vector matrix for *integrated model* will be,

$V_M =$

(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(-1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0)
(0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0)
(0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1)
(0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 1, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 1, 0, 0, 0)
(0, 0, 0, -1, 0, 0, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, -1, 1, 1, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, -1, 1, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 1, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 1, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)

….continue next page

Continued..

(0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0)
(-1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)

In section 7.3, we will discuss the results of the computational experiments and visualise the state-space of the different modules, the time taken to solve the CME and the conditional probabilities of the biochemical species.

7.3 Computational Experiments

In this section, we tested the ability of the *ISP LAS* to reproduce the integrated model of the fungal pathogen, *C. albicans*, through its modules to measure the dynamics of the key biochemical species, as discussed in sections 7.1 and 7.2. We performed the experiment on the carbon-neutral platform of Amazon® Web Service Elastic Computing (EC2) instance type large (m5a) running on HVM (hardware virtual environment) virtualisation with variable ECUs, a multicore environment of 16vCPU @ 2.2GHz, AMD EPYC 7571 running Ubuntu 16.04.1 with the relevant dependencies, a 64GB memory with 8GB Elastic Block Storage (EBS) type General Purpose SSD (GP2) formatted with Elastic File System (EFS). The performance mode was set to general purpose with inputs-outputs per second (IOPS = 100/3000) and a throughput mode of type bursting is set (see Appendix F).

After *ISP LAS* simulation, we collected the outputs from the events as follows:

1) Dimensions of the integrated model of the fungal pathogen, *C. albicans*, involving its systems and pathways as modules. This is the indicating factor for knowing the difficultly level for solving the CME.

2) Time taken by *ISP LAS* to run the modules for CME approximations up to t_f. This is a determining factor for the run-time performance and, subsequently, the total time required to solve the CME.

3) Number of states at the time steps. Because once states are expanded, the domain size shows persistent increase in spite of bunking some states. This is critical for algorithm performance and the maintenance of accuracy in the solution, which requires a much longer time course in comparison with the expansion.

4) Adaptive approximation errors when the states were simultaneously expanded and bunked in different modules of the integrated model.

5) Conditional probabilities of the biochemical species at t_f and the approximate solution of the CME for the modules in terms of probability.

The integrative model (dimension = 45) of the fungal pathogen, *C. albicans*, in the presence of oxidative stress, is considered to be really stiff in nature, so that some of the molecular counts of certain biochemical species increase very quickly and some slowly (as discussed in section 7.1) in different modules, which makes it difficult to choose t_{step} or even, solve for

small durations of time. To expand the state-space and measure the stiffness of modules in the system and time to solve the CME, we will consider the modules (see Table 7.1) of the model individually and solve them for different t_f and t_{step} with $\tau_m = 1e-6$.

To capture the dynamic nature of the integrative model of the fungal pathogen *C. albicans* in the presence of oxidative stress, the *transporter module* and *catalase* modules are considered up to $t_f = 3.01$ *sec* with $t_{step} = 0.01$. Due to the number of reactions and the wide spectrum of variation in the propensities involved in the *thioredoxin*, it is considered to be very stiff in nature; therefore, it is considered with $t_{step} = 0.0001$ and solved up to $t_f = 0.00062$ *sec*. Lastly, the *pentose phosphate pathway* (PPP) is considered up to $t_f = 5.2$ *sec* with $t_{step} = 0.01$. Further, the *protein-thiol module* is also considered stiff in nature due to the nature of its reactions; therefore, it is solved up to $t_f = 0.04$ *sec* with same $t_{step} = 0.0001$ as that of *thioredoxin*. In signalling and gene expression module, the *cap1* and *hog1 pathways* are considered and solved for $t_f = 8.46$ *sec* with $t_{step} = 0.01$ and $t_f = 70.0$ *sec* with $t_{step} = 1.0$ respectively. A single t_f cannot be assigned to all the modules or for the whole integrated system because its modules have different stiffnesses defined by a variety of reactions so, to capture the dynamic nature, it is important to consider for different t_f.

The systematic exploration of nodes carrying probable states for this model are undertaken in similar ways, as discussed previously in Table 4.3, and depicted in Figure 4.5 in several stages (denoted as $\hat{S}$), representing R_M reactions with propensities a_μ with arcs as transitions. The nodes $\mathbf{n}_J = \left(\mathbf{X}_K, \mathrm{d}_{l=1,2,\ldots\ldots}\right)$ carrying states are expanded and updated as per *ISP LAS* trend (see section 4.3) up to t_f for R_M channels {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121} that are responsible for variation in the molecular counts of the species.

The *ISP LAS* responses between the number of states contributed to by the individual modules in the domain and time t is shown in Figure 7.2 to Figure 7.8. The change in size and colour of the *2D pyramid* in Figure 7.2 to Figure 7.8 shows the increase in size of the domain with state explorations.

In Figure 7.2 and Figure 7.3, for $t_{step} = 0.01$, the initial response at $t = 0.2$ *sec* (1726 states) and $t = 1.2$ *sec* (18169 states) in the *transporter module*, and at $t = 0.2$ *sec* (1011 states) and $t = 1.2$ *sec* (9169 states) in *catalase*, suggests that only a few reactions were active and, gradually, after $t = 1.2$ *sec* more reactions trigger that take the exploration up to 78054 and 38235 states at t_f, respectively. This response also shows that both the *transporter module* and *catalase* make steady contributions of the states in the domain without going through a number of excursions in a fraction of time, t. In both modules some species (x_0, x_1, x_{33}) undergo birth and death at very similar rates, which prevent any sudden explosion of states throughout t_f even when all the reactions $R_{M=1,2,3,4,5,77,78}$ (involving species of *transporter module*), $R_{M=40,64,79,109}$ (involving species of *catalase*) (see section 7.2.1) become active in the network.

Similarly in Figure 7.4, for $t_{step} = 0.01$, the initial response at $t = 1.5$ *sec* (6426 states) and $t = 3.2$ *sec* (12960 states) in the *pentose phosphate pathway* suggests that there were no sudden explosions of states and the domain size shows a steady increase when all reactions $R_{M=7,21,22,73,97}$ (involving species of *pentose phosphate pathway*) become active in the network and even when only one species (x_{20}) undergoes birth and death.

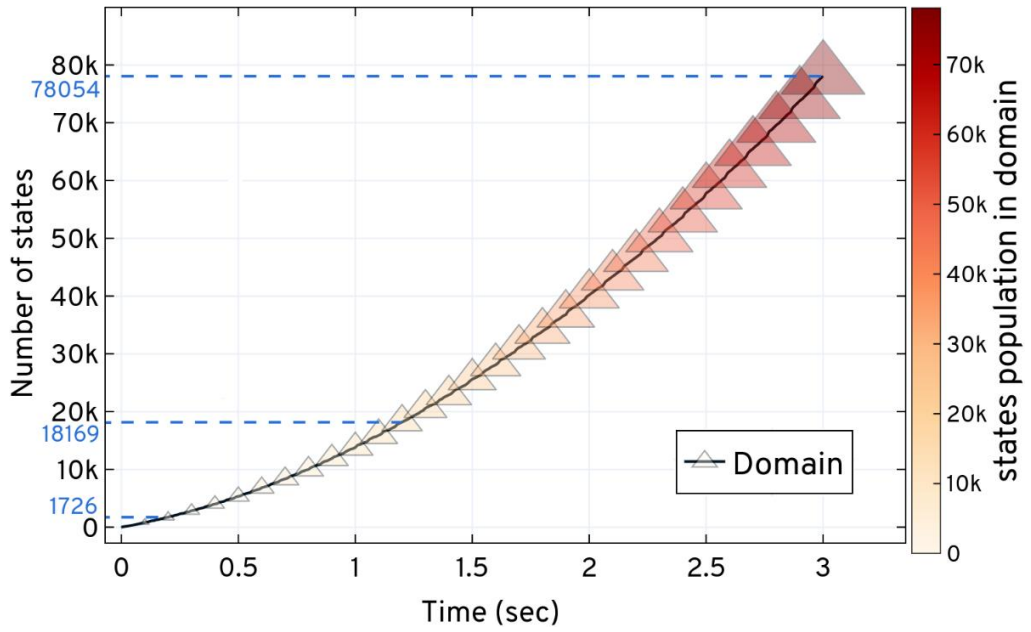


Figure 7.2. Expansion and updating trend of the states for *transporter module* in the integrative model of the fungal pathogen, *C. albicans*, based on the *ISP LAS* method. *ISP LAS* quickly expands the state-space up to 78054 states for 3.01 *sec* for *transporter module*. The state-space expansion of *transporter module* contributes in increasing the number of additions of new states to the domain.

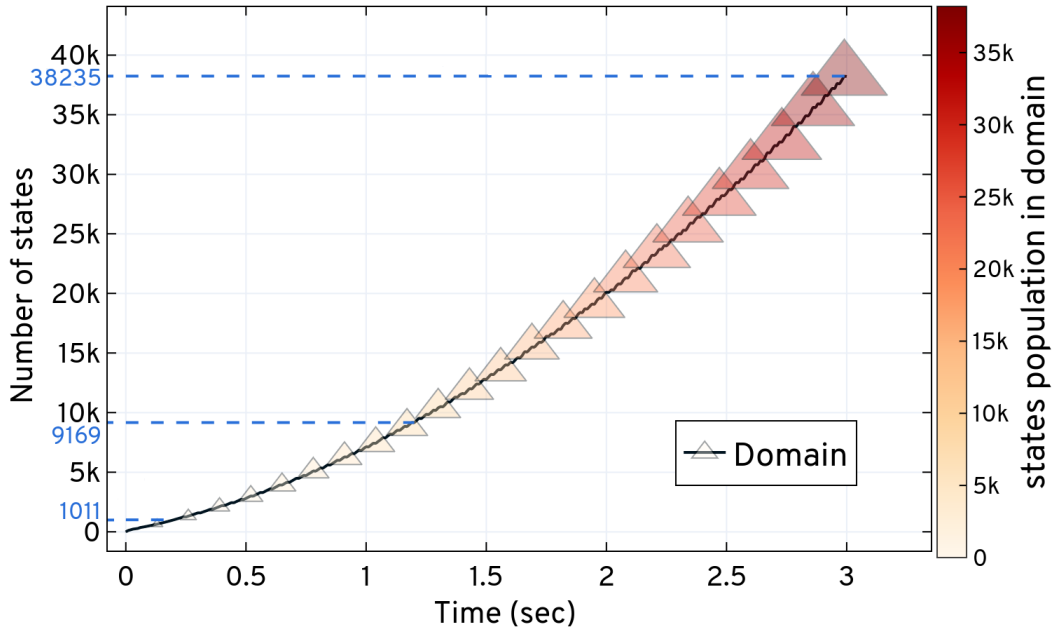


Figure 7.3. Expansion and updating trend of the states for *catalase* (antioxidant) in the integrative model of the fungal pathogen, *C. albicans*, based on the *ISP LAS* method. *ISP LAS* quickly expands the state-space up to 38235 states for 3.01 *sec* for *catalase*. The state-space expansion of *catalase* contributes in increasing the number of additions of new states to the domain.

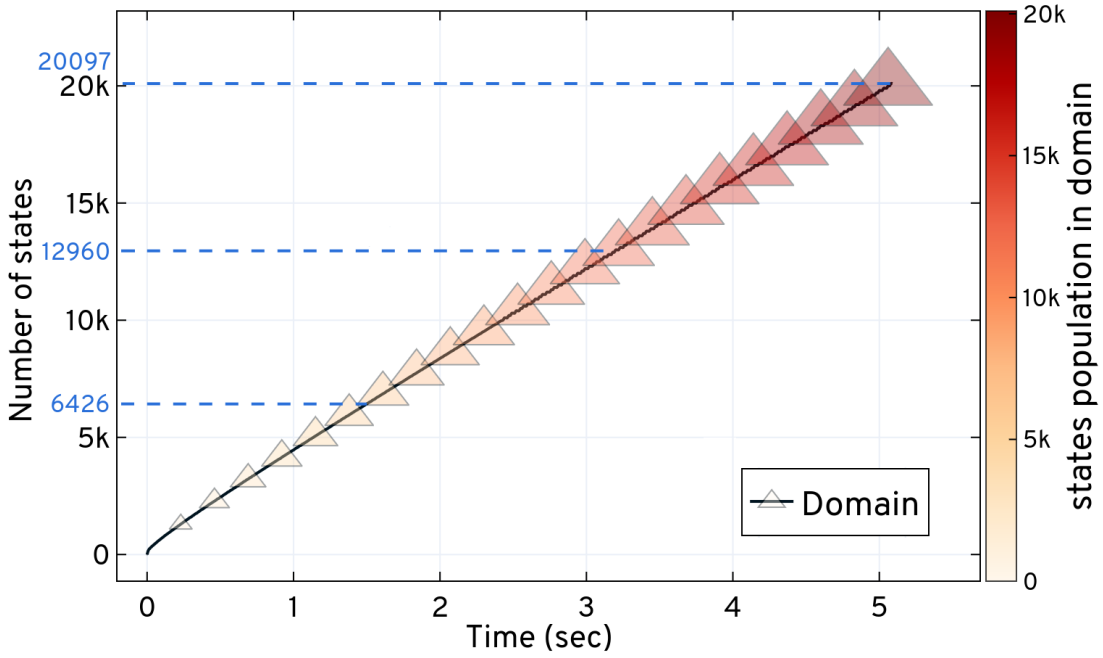


Figure 7.4. Expansion and updating trend of the states for *pentose phosphate pathway* in the integrative model of the fungal pathogen, *C. albicans*, based on the *ISP LAS* method. *ISP LAS* quickly expands the state-space up to 20097 states for 5.1 *sec* for *pentose phosphate pathway*. The state-space expansion of *pentose phosphate pathway* contributes in increasing the number of additions of new states to the domain.

However, in the case of *thioredoxin* in Figure 7.5, the initial response at $t = 0.0002$ *sec* (136916 states) is itself, considered as the first explosion of states followed by a second at $t = 0.0006$ *sec* (6641483 states). The *thioredoxin* part of the network is very stiff in nature due to the typical behaviour of its reactions that undergo birth and death of species (x_{12}, x_{13}, x_{14}, x_{15}, x_{18}, x_{19}, x_{16}, x_{17}, x_{20},) having a high number of molecule counts of certain species (x_{12}, x_{14}, x_{16}, x_{18} and x_{20}) going through a number of excursions in fractions of time, t, that make it difficult to capture the dynamic nature with high t_{step}.

The response of the *protein-thiol module* in Figure 7.6 is quite similar to that of *catalase*; but due to its stiff nature (like *thioredoxin*) some of its species (x_9, x_{10}, x_{11}, x_{12}, x_{13}) that have a high number of molecule counts go through a number of excursions in a fraction of time, t, that leads to more new states per t_{step} = 0.0001. The domain size also shows a steady increase when all the reactions $R_{M=9,10,11,12,14,15,16,86,87,88,89,90}$ (involving species of *protein-thiol*) become active in the network leading to 41185 states at $t = 0.014$ *sec*, 132613 states at $t = 0.028$ *sec* and 245701 states at $t = 0.040$ *sec*.

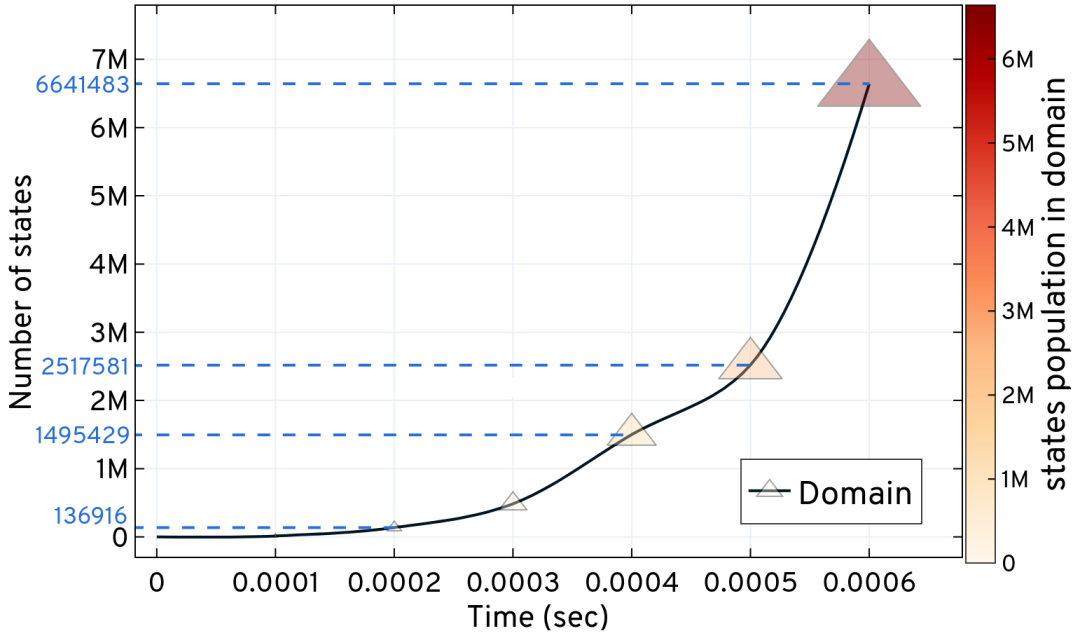


Figure 7.5. Expansion and updating trend of the states for *thioredoxin* (antioxidant) in the integrative model of the fungal pathogen, *C. albicans*, based on the *ISP LAS* method. *ISP LAS* quickly expands the state-space up to 6641483 states for 0.00062 *sec* for *thioredoxin* (antioxidant). The state-space expansion of *thioredoxin* (antioxidant) contributes in increasing the number of additions of new states to the domain.

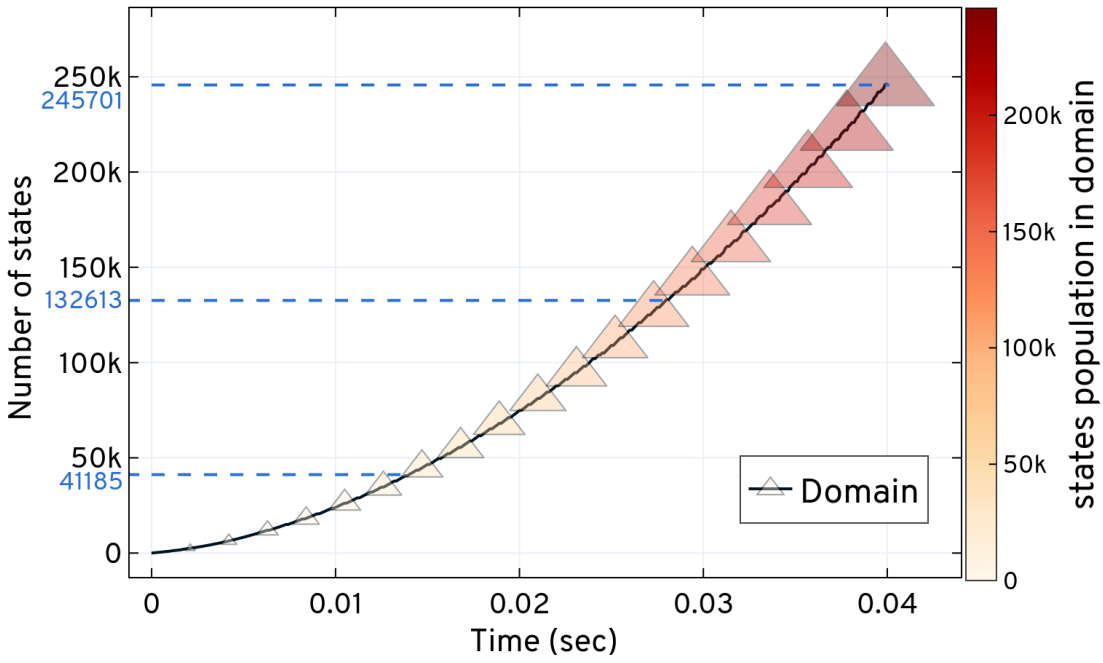


Figure 7.6. Expansion and updating trend of the states for *protein mono and di-thiols* in the integrative model of the fungal pathogen, *C. albicans*, based on the *ISP LAS* method. *ISP LAS* quickly expands the state-space up to 245701 states for 0.040 *sec* for *protein mono and di-thiols*. The state-space expansion of *protein mono and di-thiols* contributes in increasing the number of additions of new states to the domain.

In Figure 7.7, for t_{step} = 0.01, the initial response at $t = 3.01$ *sec* (73220 states) in the *cap1 pathway* suggests that only one reaction was most active ($R_{M=23}$) in the network and, as the number of active reactions increases, the number of steps formed in Fig (6) decrease with time as a single new step adds sufficient new states in the domain. The size of the steps tends to increase from $t = 3.01$ *sec* and continues to explore new states at $t = 7.01$ *sec* (1536188 states) until $t = 8.46$ *sec* (2367590 states). Species x_{21}, x_{22}, x_{23} were most active in the network that leads to antioxidant gene regulation under the condition of oxidative stress. The number of excursions these species go through decides the size of the step formed in Figure 7.7.

Lastly, in Figure 7.8, for t_{step} = 1.0, the initial and intermediate responses at $t = 25.0$ *sec* (3102 states) and $t = 50.0$ *sec* (9716 states) in the *hog1 pathway* suggest that it has the lowest stiffness among all the modules in the integrated model. The active nature of the reactions $R_{M=28,29,30,31,32,33,34,35,36,37,38,39,74,75,76}$ at the initial level leads to steady responses until 9716 states while the domain size remains constant until 6.0 *sec* (9716 states). No species (x_{24}, x_{25}, x_{26}, x_{27}, x_{28}, x_{29}, x_{30}, x_{31}, x_{42}, x_{43}, x_{44}, x_{46}) in the *hog1 pathway* go through the death process, instead they are phosphorylated, de- phosphorylated or activated, de-activated throughout the reactions and, hence, they do not have number of excursions, leading to a constant number of states from $t = 7.0$ *sec* (15980 states) to $t = 200.0$ *sec* (15980 states).

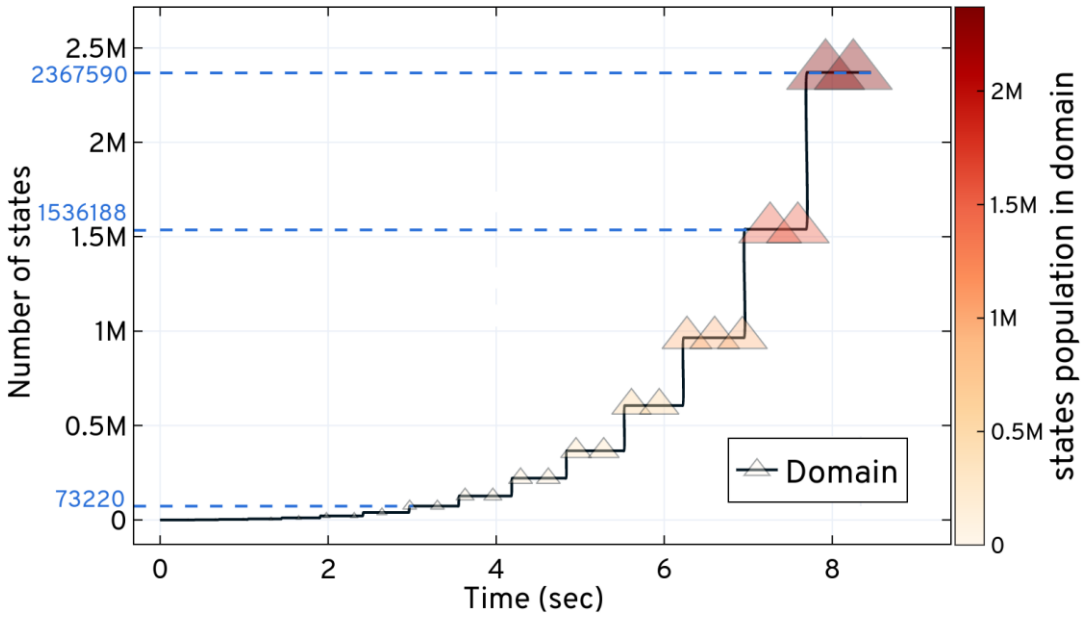


Figure 7.7. Expansion and updating trend of the states for *cap1 pathway* in the integrative model of the fungal pathogen, *C. albicans*, based on the *ISP LAS* method. *ISP LAS* quickly expands the state-space up to 2367590 states for 8.46 *sec* for *cap1 pathway*. The state-space expansion of *cap1 pathway* contributes in increasing the number of additions of new states to the domain.

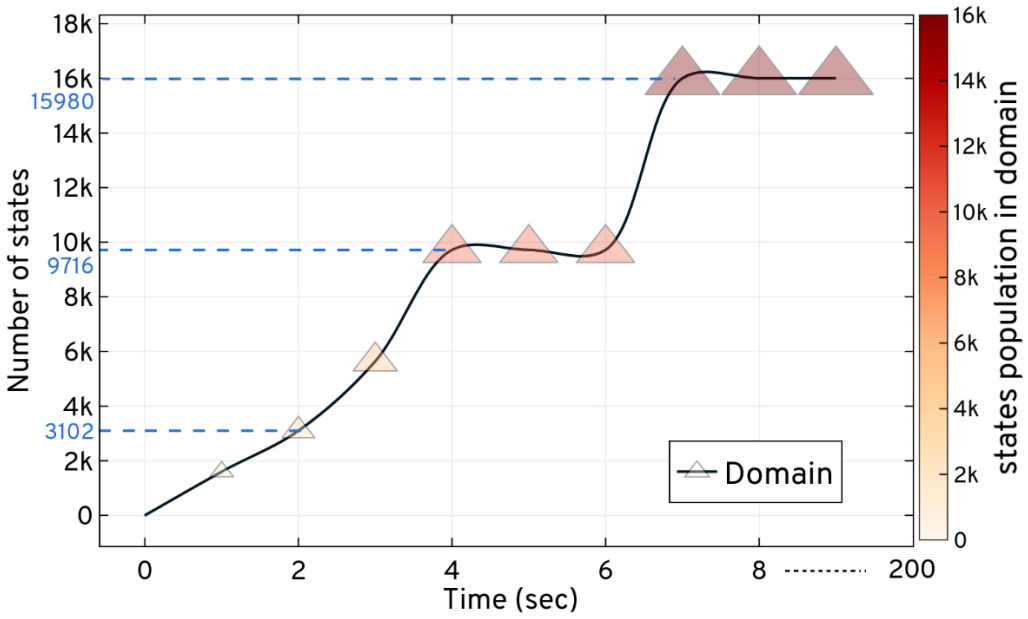


Figure 7.8. Expansion and updating trend of the states for *hog1 pathway* in the integrative model of the fungal pathogen, *C. albicans*, based on the *ISP LAS* method. *ISP LAS* quickly expands the state-space up to 15980 states for 200.0 *sec* for *hog1 pathway*. The state-space expansion of *hog1 pathway* contributes in increasing the number of additions of new states to the domain.

The *transporter module* started with the initial state of the involved species x_0, x_1, x_{46}, x_{48} and x_{49}. In the *antioxidant module*, the *catalase*, *thioredoxin* and *pentose phosphate pathway* started with the initial state of the species involved, x_2, x_{33} and x_{12}, x_{13}, x_{14}, x_{15}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}, x_{37}, x_{38}, x_{39}, x_{46} and x_4, x_{20}, x_{50}, x_{51}, x_{17}, x_{41}, respectively. The *protein-thiol module* started with the initial state of the involved species x_9, x_{46}, x_{10}, x_{11}, x_7, x_3, x_{12}, x_{13} and x_{18}, whereas, the *cap1 and hog1 pathways* started with the initial state of the species involved, x_{21}, x_{22}, x_{23}, x_{32}, x_{33}, x_{34}, x_{35}, x_{36}, x_{37}, x_{38}, x_{39}, x_{40}, x_{41}, x_{46}, x_{47} and x_{24}, x_{25}, x_{26}, x_{27}, x_{28}, x_{29}, x_{30}, x_{31}, x_{42}, x_{43}, x_{44}, x_{46} respectively (see Table 7.2). Then, gradually, when the protein level changes in the system, it exploits the copy counts of other species leading to a new state. A checkpoint has been set in $ISP\ LAS$ for all the modules to examine the initial state of probability over time t_f. The response in Figs (1) to (6) of Figure 7.9 indicate that the probability of the system to remain in the initial (normal) state decreases with time when more reactions become active in the network and this affects the counts of the species. It also shows that for some other states (Figs (1), (2) and (3) of Figure 7.9) (marked with green) the probability increases simulatenously when initial state probability decreases and at a certain time point the system shifts to another state that has a high probability.

This change in probability in the modules enforces the system to shift to new states, which manifest the Markov process of the integrated model. The $ISP\ LAS$ captures this process and defines several bounds of the domain at different time intervals, as defined by Figure 4.2 pyramid. To investigate the expansion of states closely, the order of bounds at different time intervals, and the number of states present in the bounds, are given in Table 7.7 to Table 7.13 for all the modules. The size of the bound created at each duration reveals that for every step, the growth of the domain in the *transporter module* is ≈1/4th times the previous size of the domain. Whereas, in the *cat1 pathway*, the growth of the domain is ≈8% of the previous size of the domain. This rate is also incremental (by 0.40 to 0.70 units) in every $t_{step} = 0.01$ of *cap1 pathway*. Initially, the growth of the domain is variable in the *pentose pathway*; however, after $t = 200.0$ *sec* the domain is updated with 36 or 45 states alternatively for every $t_{step} = 0.01$. In *thioredoxin*, the growth of the domain is variable and falls between 1.5 to 10 times the previous size of the domain. Due to its stiff nature, it also has the least number of bounds but they are the largest in size, among all the modules' bounds.

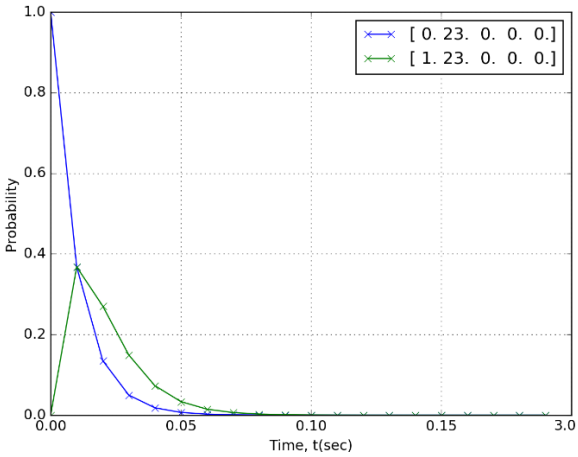


Fig (1). Response of *ISP LAS* checkpoint for examining the *transporter module's* initial state and any random state probability over time $t_f = 3.0\ sec.$

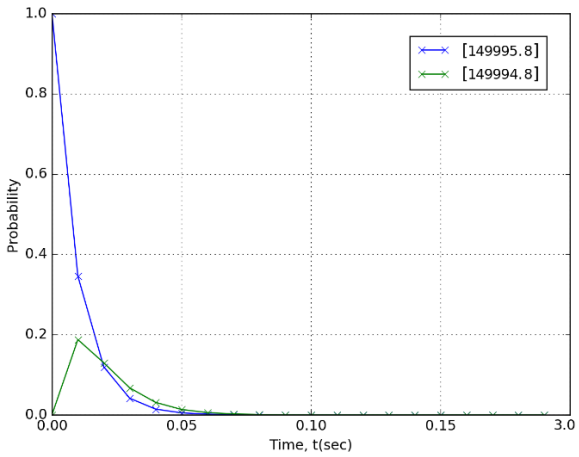


Fig (2). Response of *ISP LAS* checkpoint for examining the *catalase's* (antioxidant) initial state and any random state probability over time $t_f = 3.0\ sec.$

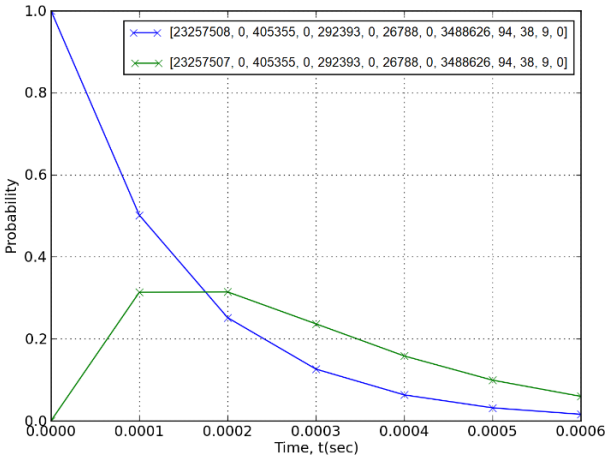


Fig (3). Response of *ISP LAS* checkpoint for examining the *thioredoxin* (antioxidant) initial state and any random state probability over time $t_f = 0.00062\ sec.$

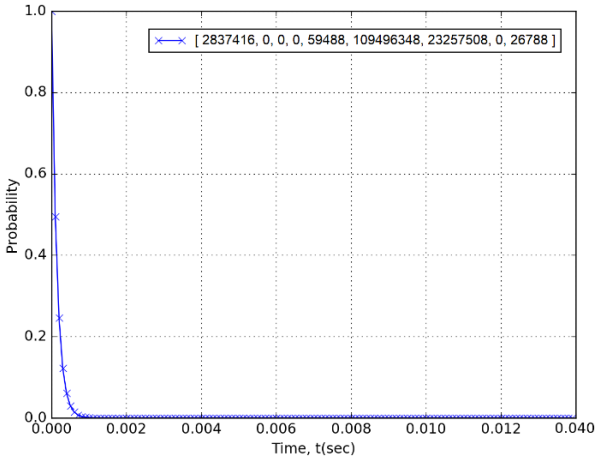


Fig (4). Response of *ISP LAS* checkpoint for examining the *protein mono and di-thiol* initial state over time $t_f = 0.04\ sec.$

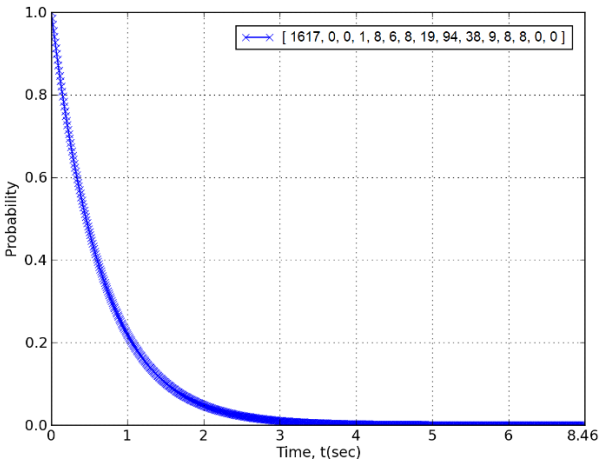


Fig (5). Response of *ISP LAS* checkpoint for examining the *cap1 pathway's* initial state over time $t_f = 8.46\ sec$.

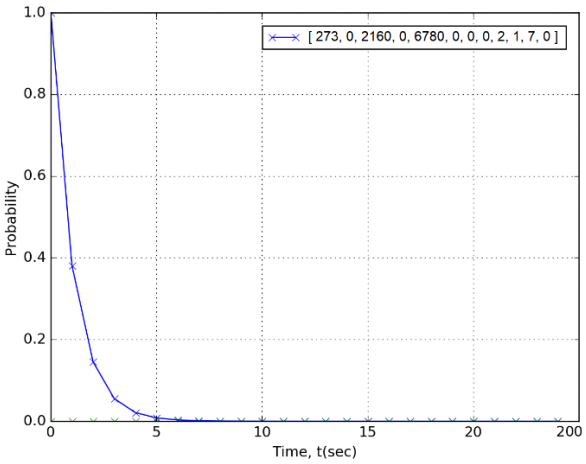


Fig (6). Response of *ISP LAS* checkpoint for examining the *hog1 pathway's* initial state over time $t_f = 200.0\ sec$.

Figure 7.9. Response of the *ISP LAS* checkpoint for examining the integrative model of fungal pathogen *C. albicans* module's (Figs (1) for *transporter module's*, Figs (2) for *catalase's*, Figs (3) for *thioredoxin*, Figs (4) for *protein mono and di-thiol*, Figs (5) for *cap1 pathway's*, Figs (6) for *hog1 pathway's*), initial state and any random state probability over time t_f.

In the *protein-thiol module*, the growth of the domain is variable until $t = 0.0025$ *sec*, and after that it increases at the rate of 3.0% of the previous size of the domain. This rate is also incremental (by 0.50 units) in every $t_{step} = 0.0001$. In the *cap1 pathway*, the growth of the domain is also variable throughout $t_f = 8.46\ sec$; however, after several $t_{step} = 0.01$ the explosion of states updates the domain with a large number of states. The number of unique states added in the domain increases with t, together with the number of explosions in the expansion process. In the *hog1 pathway*, the growth of the domain is also variable until $t = 7.0$ *sec* and, after this, the size of the domain remains constant until $t_f = 200.0\ sec$ as there are no new states added in the domain.

The number of sets of states that created the several bounds at t are shown in Figure 7.10 to Figure 7.16 for all modules of the integrated model. In the *transporter module* in Figure 7.10, with the exploration of the set of 99 states, $Bound(1)_{upper} = \{X_{0,1,2\ldots.99}\}$ is formed at 0.01 *sec* carrying 100 states. Some states were bunked after 0.3 *sec* resulting in approximation error reaching to $2.48\text{e}-05$ at 0.4 *sec*. At t_f, the *LAS* ends with a domain defined by $Bound(301)_{upper} = \{X_{0,1,2\ldots.78053}\}$ carrying 78054 states with a $1.99\text{e}-04$ approximation error. A set of nodes $N_1, N_2, \ldots.. N_{78054}$ carries these unique states representing the set of $state(n_{78054}) = \left(X_{0,1,2,\ldots 78053}\right)$ forming the state-space of the *transporter module* up to t_f.

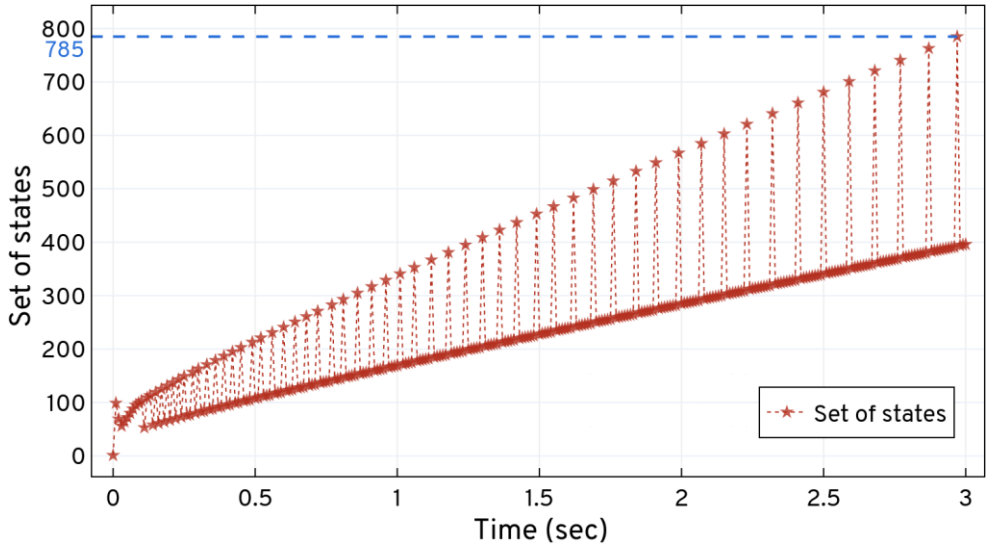


Figure 7.10. The set of states explored for the *transporter module* using the *ISP LAS* algorithm. Based on network of reactions, *ISP LAS* unfolds the state-space pattern to update states in the domain and expands 78054 states up to t_f. ★ shows the time point where the new set of states are explored and updated in the domain.

Table 7.7. Lower and upper Bounds of the *transporter module* given by *ISP LAS* trend.

Z	$Bound(Z)_{lower}$	$Bound(Z)_{upper}$	States	Duration
1	$Bound(1)_{lower} = \{X_0\}$ formed at $t = 0.0$ *sec* $Approximation = 1$	$Bound(1)_{upper} = \{X_{0,1,2\ldots.99}\}$ formed at $t = 0.01$ *sec* $Approximation = 0.9989841635$	100	0.0 – 0.01 *sec*
2	$Bound(2)_{lower} = Bound(1)_{upper}$ formed at $t = 0.01$ *sec* $Approximation = 0.998984163$	$Bound(2)_{upper} = \{X_{0,1,2\ldots.168}\}$ formed at $t = 0.02$ *sec* $Approximation = 0.9971932273$	169	0.01 – 0.02 *sec*
3	$Bound(3)_{lower} = Bound(2)_{upper}$ formed at $t = 0.02$ *sec* $Approximation = 0.997193227$	$Bound(3)_{upper} = \{X_{0,1,2\ldots.224}\}$ formed at $t = 0.03$ *sec* $Approximation = 0.9917109929$	225	0.02 – 0.03 *sec*
4	$Bound(4)_{lower} = Bound(3)_{upper}$ formed at $t = 0.03$ *sec* $Approximation = 0.991710992$	$Bound(4)_{upper} = \{X_{0,1,2\ldots.288}\}$ formed at $t = 0.04$ *sec* $Approximation = 0.9955283311$	289	0.03 – 0.04 *sec*
5	$Bound(5)_{lower} = Bound(4)_{upper}$ formed at $t = 0.04$ *sec* $Approximation = 0.995528331$	$Bound(5)_{upper} = \{X_{0,1,2\ldots.360}\}$ formed at $t = 0.05$ *sec* $Approximation = 0.9931313777$	361	0.04 – 0.05 *sec*
6	$Bound(6)_{lower} = Bound(5)_{upper}$ formed at $t = 0.05$ *sec* $Approximation = 0.993131377$	$Bound(6)_{upper} = \{X_{0,1,2\ldots.440}\}$ formed at $t = 0.06$ *sec* $Approximation = 0.9909475971$	441	0.05 – 0.06 *sec*
7	$Bound(7)_{lower} = Bound(6)_{upper}$ formed at $t = 0.06$ *sec* $Approximation = 0.990947597$	$Bound(7)_{upper} = \{X_{0,1,2\ldots.528}\}$ formed at $t = 0.07$ *sec* $Approximation = 0.9891430655$	529	0.06 – 0.07 *sec*
8	$Bound(8)_{lower} = Bound(7)_{upper}$ formed at $t = 0.07$ *sec* $Approximation = 0.98914306$	$Bound(8)_{upper} = \{X_{0,1,2\ldots.623}\}$ formed at $t = 0.08$ *sec* $Approximation = 0.987747528$	624	0.07 – 0.08 *sec*
9	$Bound(9)_{lower} = Bound(8)_{upper}$ formed at $t = 0.08$ *sec* $Approximation = 0.987747528$	$Bound(9)_{upper} = \{X_{0,1,2\ldots.722}\}$ formed at $t = 0.09$ *sec* $Approximation = 0.9867138883$	723	0.08 – 0.09 *sec*
10	$Bound(10)_{lower} = Bound(9)_{upper}$ formed at $t = 0.09$ *sec* $Approximation = 0.986713888$	$Bound(10)_{upper} = \{X_{0,1,2\ldots.825}\}$ formed at $t = 0.1$ *sec* $Approximation = 0.9859677839$	826	0.09 – 0.1 *sec*
:	:		:	:
301	$Bound(301)_{lower} = Bound(300)_{upper}$ formed at $t = 2.9$ *sec* $Approximation = 0.96371671$	$Bound(301)_{upper} = \{X_{0,1,2\ldots.78053}\}$ formed at $t = 3.0$ *sec* $Approximation = 0.9635885099$	78054	2.9 – 3.0 *sec*

(See section G.4 in Appendix G for details of bounds for *transporter module*).

In the *cat1 pathway*, Figure 7.11, with the exploration of the set of 112 states, the $Bound(1)_{upper} = \{X_{0,1,2.....112}\}$ is formed at 0.01 *sec* carrying 113 states. Some states were bunked after 0.22 *sec*, resulting in the approximation error reaching $1.37e-05$ at 0.24 *sec*. At t_f, the *LAS* ends with the domain defined by $Bound(301)_{upper} = \{X_{0,1,2.....38234}\}$ carrying 38235 states with $1.418e-04$ approximation error. The set of nodes $N_1, N_2, \ldots . N_{38235}$ carries these unique states representing the set of $state(n_{38235}) = (X_{0,1,2,...38234})$ forming the state-space of the *cat1 pathway* up to t_f.

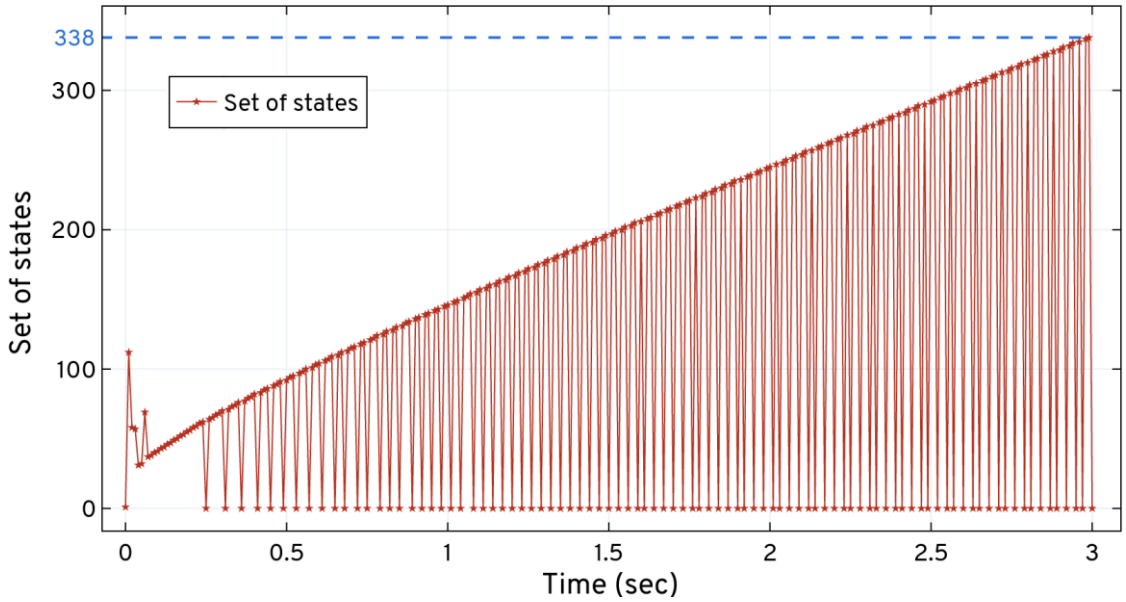


Figure 7.11. The set of states explored for the *cat1 pathway* using *ISP LAS* algorithm. Based on network of reactions, *ISP LAS* unfolds the state-space pattern to update the states in the domain and expands 38235 states up to t_f. ★ shows the time point where the new set of states is explored and updated in the domain.

Table 7.8. Lower and upper Bounds of the *cat1 pathway* given by *ISP LAS* trend.

Z	$Bound(Z)_{lower}$	$Bound(Z)_{upper}$	States	Duration
1	$Bound(1)_{lower} = \{X_0\}$ formed at $t = 0.0$ *sec* $Approximation = 1$	$Bound(1)_{upper} = \{X_{0,1,2.....112}\}$ formed at $t = 0.01$ *sec* $Approximation = 0.9989549833$	113	0.0 – 0.01 *sec*
2	$Bound(2)_{lower} = Bound(1)_{upper}$ formed at $t = 0.01$ *sec* $Approximation = 0.998954983$	$Bound(2)_{upper} = \{X_{0,1,2.....171}\}$ formed at $t = 0.02$ *sec* $Approximation = 0.9969707281$	171	0.01 – 0.02 *sec*
3	$Bound(3)_{lower} = Bound(2)_{upper}$ formed at $t = 0.02$ *sec* $Approximation = 0.997193227$	$Bound(3)_{upper} = \{X_{0,1,2.....227}\}$ formed at $t = 0.03$ *sec* $Approximation = 0.9958099024$	228	0.02 – 0.03 *sec*
4	$Bound(4)_{lower} = Bound(3)_{upper}$ formed at $t = 0.03$ *sec* $Approximation = 0.991710992$	$Bound(4)_{upper} = \{X_{0,1,2.....258}\}$ formed at $t = 0.04$ *sec* $Approximation = 0.9919955001$	259	0.03 – 0.04 *sec*
5	$Bound(5)_{lower} = Bound(4)_{upper}$ formed at $t = 0.04$ *sec* $Approximation = 0.995528331$	$Bound(5)_{upper} = \{X_{0,1,2.....290}\}$ formed at $t = 0.05$ *sec* $Approximation = 0.9958406538$	291	0.04 – 0.05 *sec*
6	$Bound(6)_{lower} = Bound(5)_{upper}$ formed at $t = 0.05$ *sec* $Approximation = 0.993131377$	$Bound(6)_{upper} = \{X_{0,1,2.....359}\}$ formed at $t = 0.06$ *sec* $Approximation = 0.995439096$	360	0.05 – 0.06 *sec*
7	$Bound(7)_{lower} = Bound(6)_{upper}$ formed at $t = 0.06$ *sec* $Approximation = 0.990947597$	$Bound(7)_{upper} = \{X_{0,1,2.....396}\}$ formed at $t = 0.07$ *sec* $Approximation = 0.9946508528$	397	0.06 – 0.07 *sec*
8	$Bound(8)_{lower} = Bound(7)_{upper}$ formed at $t = 0.07$ *sec* $Approximation = 0.98914306$	$Bound(8)_{upper} = \{X_{0,1,2.....434}\}$ formed at $t = 0.08$ *sec* $Approximation = 0.9935874681$	435	0.07 – 0.08 *sec*
9	$Bound(9)_{lower} = Bound(8)_{upper}$ formed at $t = 0.08$ *sec* $Approximation = 0.987747528$	$Bound(9)_{upper} = \{X_{0,1,2.....474}\}$ formed at $t = 0.09$ *sec* $Approximation = 0.992328262$	475	0.08 – 0.09 *sec*
10	$Bound(10)_{lower} = Bound(9)_{upper}$ formed at $t = 0.09$ *sec* $Approximation = 0.986713888$	$Bound(10)_{upper} = \{X_{0,1,2.....515}\}$ formed at $t = 0.1$ *sec* $Approximation = 0.9909426395$	516	0.09 – 0.1 *sec*
:	:		:	:
301	$Bound(301)_{lower} = Bound(300)_{upper}$ formed at $t = 2.9$ *sec* $Approximation = 0.96371671$	$Bound(301)_{upper} = \{X_{0,1,2.....38234}\}$ formed at $t = 3.0$ *sec* $Approximation = 0.9740284241$	38235	2.9 – 3.0 *sec*

(See section G.4 in Appendix G for details of bounds for *cat1 pathway*).

In the *pentose pathway* in Figure 7.12, with the exploration of the set of 197 states, the $Bound(1)_{upper} = \{X_{0,1,2.....197}\}$ is formed at 0.01 *sec* carrying 198 states. Some states were bunked after 0.07 *sec* resulting in the approximation error reaching $5.11e-05$ at 0.729 *sec*. At t_f, the *LAS* ends with a domain defined by $Bound(510)_{upper} = \{X_{0,1,2.....20096}\}$ carrying 20097 states with $3.90e-04$ approximation error. The set of nodes, $N_1, N_2, \ldots . N_{20097}$ carries these unique states representing the set of $state(n_{20097}) = (X_{0,1,2,...20096})$ forming the state-space of the *pentose pathway* up to t_f.

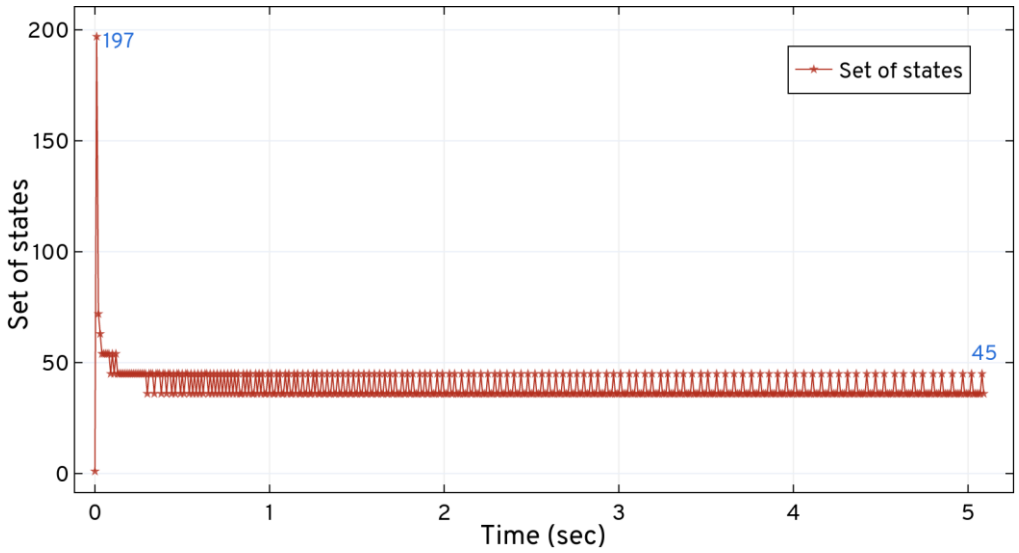


Figure 7.12. The set of states explored for the *pentose pathway* using the *ISP LAS* algorithm. Based on network of reactions, *ISP LAS* unfolds the state-space pattern to update the states in the domain and expands 20097 states up to t_f. ★ shows the time point where the new set of states are explored and updated in the domain.

Table 7.9. Lower and upper Bounds of the *pentose pathway* given by *ISP LAS* trend

Z	$Bound(Z)_{lower}$	$Bound(Z)_{upper}$	States	Duration
1	$Bound(1)_{lower} = \{X_0\}$ formed at $t = 0.0$ *sec* $Approximation = 1$	$Bound(1)_{upper} = \{X_{0,1,2.....197}\}$ formed at $t = 0.01$ *sec* $Approximation = 0.9975105622$	198	0.0 – 0.01 *sec*
2	$Bound(2)_{lower} = Bound(1)_{upper}$ formed at $t = 0.01$ *sec* $Approximation = 0.998954983$	$Bound(2)_{upper} = \{X_{0,1,2.....269}\}$ formed at $t = 0.02$ *sec* $Approximation = 0.9939852266$	270	0.01 – 0.02 *sec*
3	$Bound(3)_{lower} = Bound(2)_{upper}$ formed at $t = 0.02$ *sec* $Approximation = 0.996970728$	$Bound(3)_{upper} = \{X_{0,1,2.....333}\}$ formed at $t = 0.03$ *sec* $Approximation = 0.9972461063$	333	0.02 – 0.03 *sec*
4	$Bound(4)_{lower} = Bound(3)_{upper}$ formed at $t = 0.03$ *sec* $Approximation = 0.995809902$	$Bound(4)_{upper} = \{X_{0,1,2.....386}\}$ formed at $t = 0.04$ *sec* $Approximation = 0.9952259233$	387	0.03 – 0.04 *sec*
5	$Bound(5)_{lower} = Bound(4)_{upper}$ formed at $t = 0.04$ *sec* $Approximation = 0.991995500$	$Bound(5)_{upper} = \{X_{0,1,2.....440}\}$ formed at $t = 0.05$ *sec* $Approximation = 0.9929107688$	441	0.04 – 0.05 *sec*
6	$Bound(6)_{lower} = Bound(5)_{upper}$ formed at $t = 0.05$ *sec* $Approximation = 0.995840653$	$Bound(6)_{upper} = \{X_{0,1,2.....494}\}$ formed at $t = 0.06$ *sec* $Approximation = 0.9907666481$	495	0.05 – 0.06 *sec*
7	$Bound(7)_{lower} = Bound(6)_{upper}$ formed at $t = 0.06$ *sec* $Approximation = 0.995439096$	$Bound(7)_{upper} = \{X_{0,1,2.....548}\}$ formed at $t = 0.07$ *sec* $Approximation = 0.9890067341$	549	0.06 – 0.07 *sec*
8	$Bound(8)_{lower} = Bound(7)_{upper}$ formed at $t = 0.07$ *sec* $Approximation = 0.994650852$	$Bound(8)_{upper} = \{X_{0,1,2.....602}\}$ formed at $t = 0.08$ *sec* $Approximation = 0.9876657292$	603	0.07 – 0.08 *sec*
9	$Bound(9)_{lower} = Bound(8)_{upper}$ formed at $t = 0.08$ *sec* $Approximation = 0.993587468$	$Bound(9)_{upper} = \{X_{0,1,2.....647}\}$ formed at $t = 0.09$ *sec* $Approximation = 0.9853233084$	648	0.08 – 0.09 *sec*
10	$Bound(10)_{lower} = Bound(9)_{upper}$ formed at $t = 0.09$ *sec* $Approximation = 0.992328262$	$Bound(10)_{upper} = \{X_{0,1,2.....701}\}$ formed at $t = 0.1$ *sec* $Approximation = 0.9839872609$	702	0.09 – 0.1 *sec*
:	:		:	:
510	$Bound(510)_{lower} = Bound(509)_{upper}$ formed at $t = 5.08$ *sec* $Approximation = 0.928770480$	$Bound(510)_{upper} = \{X_{0,1,2.....20096}\}$ formed at $t = 5.1$ *sec* $Approximation = 0.9286239553$	20097	5.08 – 5.1 *sec*

(See section G.4 in Appendix G for details of bounds for *pentose pathway*).

In the *thioredoxin* in Figure 7.13, with the exploration of the set of 14452 states, the $Bound(1)_{upper} = \{X_{0,1,2.....14452}\}$ is formed at 0.0001 *sec* carrying 14453 states. Some states were bunked before 0.0001 *sec* resulting in approximation error that quickly reaches $6.92e-07$ at 0.0001 *sec*. At t_f, the *LAS* ends with a domain defined by $Bound(6)_{upper} = \{X_{0,1,2.....6641482}\}$ carrying 6641483 states with $2.14e-06$ approximation error. The set of nodes $N_1, N_2, \ldots . N_{6641483}$ carries these unique states representing the set of $state(n_{6641483}) = \left(X_{0,1,2,...6641482}\right)$ forming the state-space of the *thioredoxin* up to t_f.

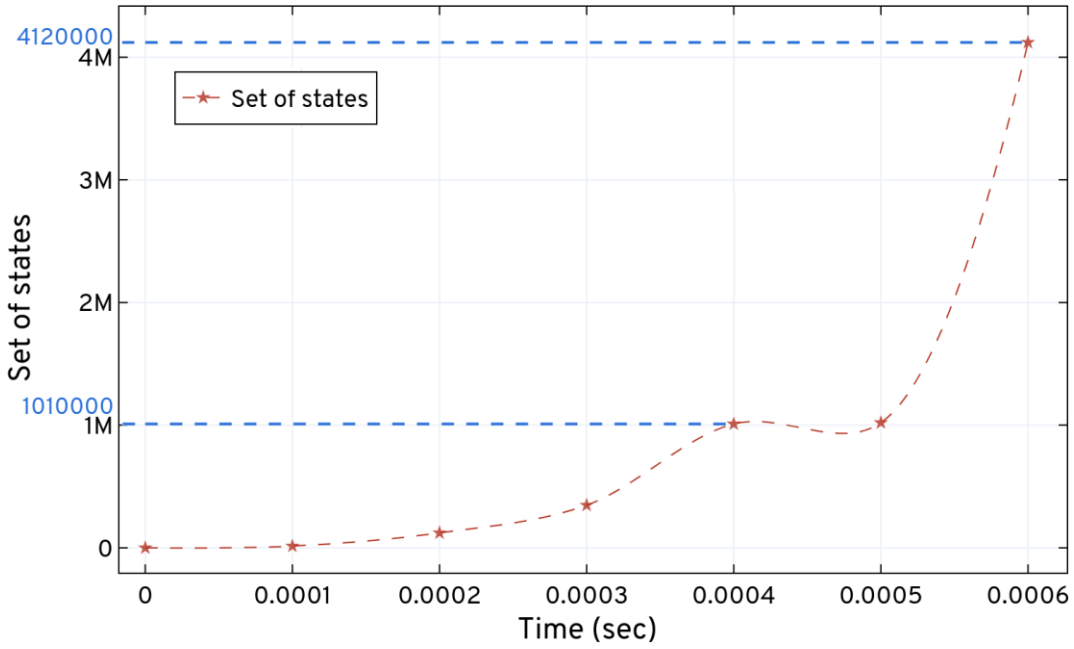


Figure 7.13. The set of states explored for *thioredoxin* using the *ISP LAS* algorithm. Based on network of reactions, *ISP LAS* unfolds the state-space pattern to update states in the domain and expands 6641483 states up to t_f. ★ shows the time point where the new set of states are explored and updated in the domain.

Table 7.10. Lower and upper Bounds of the *thioredoxin* given by *ISP LAS* trend

Z	$Bound(Z)_{lower}$	$Bound(Z)_{upper}$	States	Duration
1	$Bound(1)_{lower} = \{X_0\}$ formed at $t = 0.0$ *sec* $Approximation = 1$	$Bound(1)_{upper} = \{X_{0,1,2.....14452}\}$ formed at $t = 0.0001$ *sec* $Approximation = 0.9936943362$	14453	0.0 – 0.0001 *sec*
2	$Bound(2)_{lower} = Bound(1)_{upper}$ formed at $t = 0.0001$ *sec* $Approximation = 0.993694336$	$Bound(2)_{upper} = \{X_{0,1,2.....136915}\}$ formed at $t = 0.0002$ *sec* $Approximation = 0.9917237593$	136916	0.0001 – 0.0002 *sec*
3	$Bound(3)_{lower} = Bound(2)_{upper}$ formed at $t = 0.0002$ *sec* $Approximation = 0.991723759$	$Bound(3)_{upper} = \{X_{0,1,2.....484636}\}$ formed at $t = 0.0003$ *sec* $Approximation = 0.9971271262$	484637	0.0002 – 0.0003 *sec*
4	$Bound(4)_{lower} = Bound(3)_{upper}$ formed at $t = 0.0003$ *sec* $Approximation = 0.997127126$	$Bound(4)_{upper} = \{X_{0,1,2.....1495428}\}$ formed at $t = 0.0004$ *sec* $Approximation = 0.9967032596$	1495429	0.0003 – 0.0004 *sec*
5	$Bound(5)_{lower} = Bound(4)_{upper}$ formed at $t = 0.0004$ *sec* $Approximation = 0.996703259$	$Bound(5)_{upper} = \{X_{0,1,2.....2517580}\}$ formed at $t = 0.0005$ *sec* $Approximation = 0.9952184871$	2517581	0.0004 – 0.0005 *sec*
6	$Bound(6)_{lower} = Bound(5)_{upper}$ formed at $t = 0.0005$ *sec* $Approximation = 0.995218487$	$Bound(6)_{upper} = \{X_{0,1,2.....6641482}\}$ formed at $t = 0.0006$ *sec* $Approximation = 0.9946855553$	6641483	0.0005 – 0.0006 *sec*

(See section G.4 in Appendix G for details of bounds for *thioredoxin pathway*).

In the *protein thiol* in Figure 7.14, with the exploration of the set of 112 states, the $Bound(1)_{upper} = \{X_{0,1,2\ldots\ldots112}\}$ is formed at 0.0001 *sec*, carrying 113 states. Some states were bunked since the formation of first bound resulting in approximation error that quickly reaches $3.68e-05$ at 0.0067 *sec*. At t_f, the *LAS* ends with a domain defined by $Bound(401)_{upper} = \{X_{0,1,2\ldots\ldots245700}\}$ carrying 245701 states with $1.92e-04$ approximation error. The set of nodes $N_1, N_2, \ldots\ldots N_{245701}$ carries these unique states representing the set of $state(n_{245701}) = \left(X_{0,1,2,\ldots245700}\right)$ forming the state-space of *protein thiol* up to t_f.

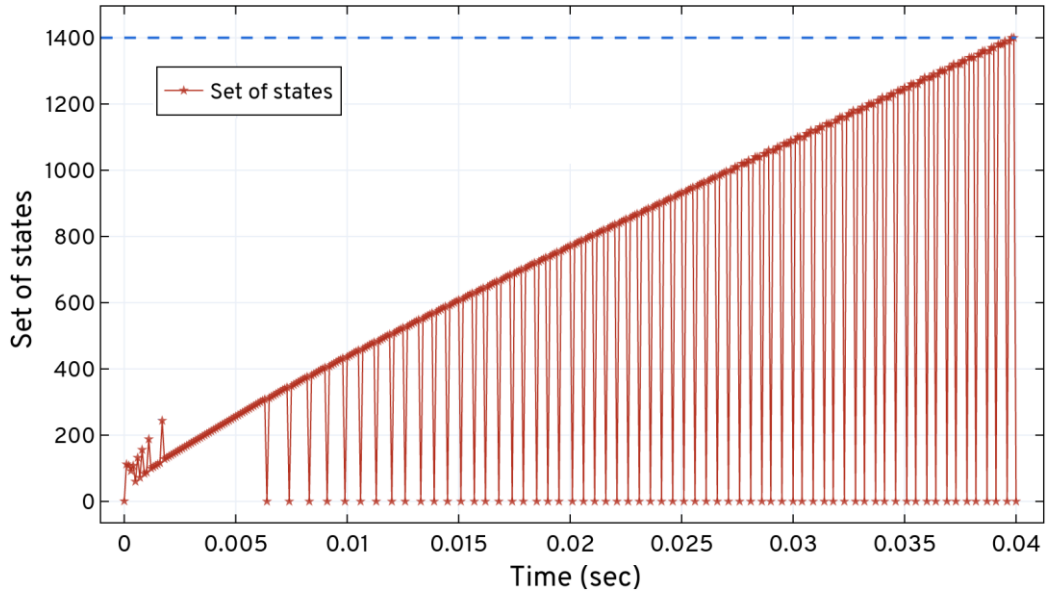


Figure 7.14. The set of states explored for *protein-thiol* using the *ISP LAS* algorithm. Based on network of reactions, *ISP LAS* unfolds the state-space pattern to update states in the domain and expands 245701 states up to t_f. ★ shows the time point where the new set of states are explored and updated in the domain.

Table 7.11. Lower and upper Bounds of the *protein-thiol* given by *ISP LAS* trend

Z	$Bound(Z)_{lower}$	$Bound(Z)_{upper}$	States	Duration
1	$Bound(1)_{lower} = \{X_0\}$ formed at $t = 0.0$ *sec* $Approximation = 1$	$Bound(1)_{upper} = \{X_{0,1,2.....112}\}$ formed at $t = 0.0001$ *sec* $Approximation = 0.992908294$	113	0.0 – 0.0001 *sec*
2	$Bound(2)_{lower} = Bound(1)_{upper}$ formed at $t = 0.0001$ *sec* $Approximation = 0.998954983$	$Bound(2)_{upper} = \{X_{0,1,2.....220}\}$ formed at $t = 0.0002$ *sec* $Approximation = 0.997449364$	221	0.0001 – 0.0002 *sec*
3	$Bound(3)_{lower} = Bound(2)_{upper}$ formed at $t = 0.0002$ *sec* $Approximation = 0.997193227$	$Bound(3)_{upper} = \{X_{0,1,2.....312}\}$ formed at $t = 0.0003$ *sec* $Approximation = 0.9967082171$	313	0.0002 – 0.0003 *sec*
4	$Bound(4)_{lower} = Bound(3)_{upper}$ formed at $t = 0.0003$ *sec* $Approximation = 0.991710992$	$Bound(4)_{upper} = \{X_{0,1,2.....420}\}$ formed at $t = 0.0004$ *sec* $Approximation = 0.9961975942$	421	0.0003 – 0.0004 *sec*
5	$Bound(5)_{lower} = Bound(4)_{upper}$ formed at $t = 0.0004$ *sec* $Approximation = 0.995528331$	$Bound(5)_{upper} = \{X_{0,1,2.....480}\}$ formed at $t = 0.0005$ *sec* $Approximation = 0.9943881051$	481	0.0004 – 0.0005 *sec*
6	$Bound(6)_{lower} = Bound(5)_{upper}$ formed at $t = 0.0005$ *sec* $Approximation = 0.993131377$	$Bound(6)_{upper} = \{X_{0,1,2.....612}\}$ formed at $t = 0.0006$ *sec* $Approximation = 0.9937262782$	613	0.0005 – 0.0006 *sec*
7	$Bound(7)_{lower} = Bound(6)_{upper}$ formed at $t = 0.0006$ *sec* $Approximation = 0.990947597$	$Bound(7)_{upper} = \{X_{0,1,2.....684}\}$ formed at $t = 0.0007$ *sec* $Approximation = 0.992028333$	685	0.0006 – 0.0007 *sec*
8	$Bound(8)_{lower} = Bound(7)_{upper}$ formed at $t = 0.0007$ *sec* $Approximation = 0.98914306$	$Bound(8)_{upper} = \{X_{0,1,2.....840}\}$ formed at $t = 0.0008$ *sec* $Approximation = 0.9915177101$	841	0.0007 – 0.0008 *sec*
9	$Bound(9)_{lower} = Bound(8)_{upper}$ formed at $t = 0.0008$ *sec* $Approximation = 0.987747528$	$Bound(9)_{upper} = \{X_{0,1,2.....924}\}$ formed at $t = 0.0009$ *sec* $Approximation = 0.9903551753$	925	0.0008 – 0.0009 *sec*
10	$Bound(10)_{lower} = Bound(9)_{upper}$ formed at $t = 0.0009$ *sec* $Approximation = 0.986713888$	$Bound(10)_{upper} = \{X_{0,1,2.....1012}\}$ formed at $t = 0.001$ *sec* $Approximation = 0.9885308137$	1013	0.0009 – 0.001 *sec*
:	:		:	:
401	$Bound(401)_{lower} = Bound(400)_{upper}$ formed at $t = 0.039$ *sec* $Approximation = 0.965035445$	$Bound(401)_{upper} = \{X_{0,1,2.....245700}\}$ formed at $t = 0.04$ *sec* $Approximation = 0.9648706046$	245701	0.039 – 0.04 *sec*

(See section G.4 in Appendix G for details of bounds for *protein-thiol*).

In the *cap1 pathway* in Figure 7.15, with the exploration of the set of 19 states, the $Bound(1)_{upper} = \{X_{0,1,2.....19}\}$ is formed at 0.01 *sec* and carryies 20 states. Some states were bunked since the formation of first bound resulting in approximation error reaching to $3.64e-05$ at 1.209 *sec*. At t_f, the *LAS* ends with a domain defined by $Bound(847)_{upper} = \{X_{0,1,2.....2367589}\}$ carrying 2367590 states with $3.348e-04$ approximation error. A set of nodes $N_1, N_2, \ldots . N_{2367589}$ carries these unique states represent the set of $state(n_{2367590}) = (X_{0,1,2,...2367589})$ forming the state-space of the *protein thiol* up to t_f.

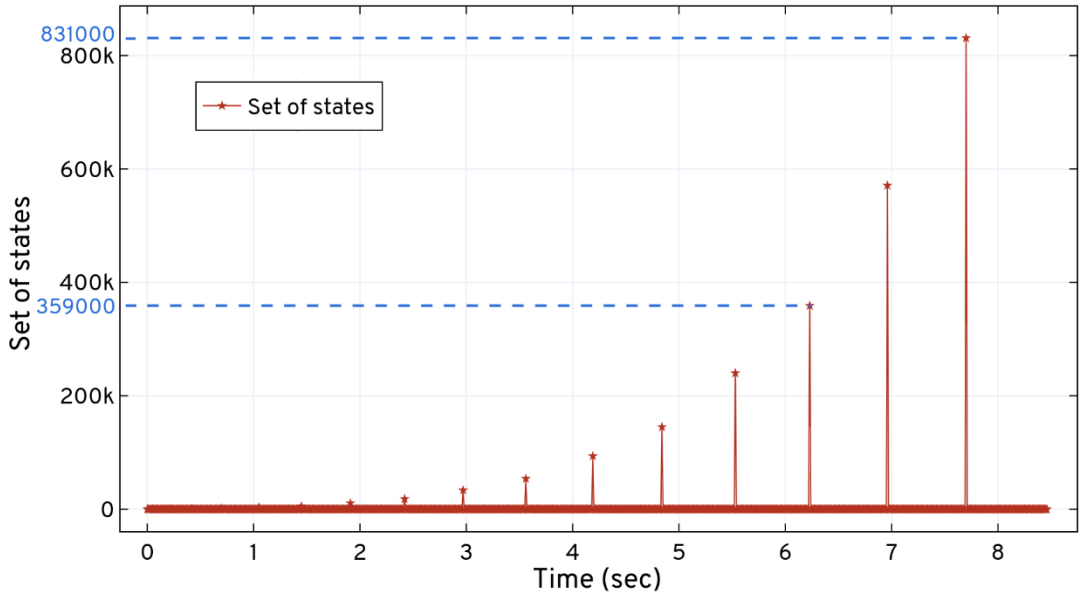


Figure 7.15. The set of states explored for *cap1 pathway* using the *ISP LAS* algorithm. Based on network of reactions, *ISP LAS* unfolds the state-space pattern to update states in the domain and expands 2367590 states up to t_f. ★ shows the time point where new set of states are explored and updated in the domain.

Table 7.12. Lower and upper bounds of the *cap1 pathway* given by *ISP LAS* trend.

Z	$Bound(Z)_{lower}$	$Bound(Z)_{upper}$	States	Duration
1	$Bound(1)_{lower} = \{X_0\}$ formed at $t = 0.0$ *sec* $Approximation = 1$	$Bound(1)_{upper} = \{X_{0,1,2\ldots.19}\}$ formed at $t = 0.01$ *sec* $Approximation = 0.9970856252$	20	0.0 – 0.01 *sec*
2	$Bound(2)_{lower} = Bound(1)_{upper}$ formed at $t = 0.01$ *sec* $Approximation = 0.997085625$	$Bound(2)_{upper} = \{X_{0,1,2\ldots.58}\}$ formed at $t = 0.02$ *sec* $Approximation = 0.9970053796$	59	0.01 – 0.02 *sec*
3	$Bound(3)_{lower} = Bound(2)_{upper}$ formed at $t = 0.02$ *sec* $Approximation = 0.997005379$	$Bound(3)_{upper} = \{X_{0,1,2\ldots.186}\}$ formed at $t = 0.08$ *sec* $Approximation = 0.9947475241$	187	0.02 – 0.08 *sec*
4	$Bound(4)_{lower} = Bound(3)_{upper}$ formed at $t = 0.08$ *sec* $Approximation = 0.994747524$	$Bound(4)_{upper} = \{X_{0,1,2\ldots.458}\}$ formed at $t = 0.21$ *sec* $Approximation = 0.985246467$	459	0.08 – 0.21 *sec*
5	$Bound(5)_{lower} = Bound(4)_{upper}$ formed at $t = 0.21$ *sec* $Approximation = 0.985246467$	$Bound(5)_{upper} = \{X_{0,1,2\ldots.1160}\}$ formed at $t = 0.42$ *sec* $Approximation = 0.991476497$	1161	0.21 – 0.42 *sec*
6	$Bound(6)_{lower} = Bound(5)_{upper}$ formed at $t = 0.42$ *sec* $Approximation = 0.991476497$	$Bound(6)_{upper} = \{X_{0,1,2\ldots.2512}\}$ formed at $t = 0.7$ *sec* $Approximation = 0.985122613$	2513	0.42 – 0.7 *sec*
7	$Bound(7)_{lower} = Bound(6)_{upper}$ formed at $t = 0.7$ *sec* $Approximation = 0.985122613$	$Bound(7)_{upper} = \{X_{0,1,2\ldots.5515}\}$ formed at $t = 1.05$ *sec* $Approximation = 0.9766664895$	5516	0.7 – 1.05 *sec*
8	$Bound(8)_{lower} = Bound(7)_{upper}$ formed at $t = 1.05$ *sec* $Approximation = 0.976666489$	$Bound(8)_{upper} = \{X_{0,1,2\ldots.10884}\}$ formed at $t = 1.45$ *sec* $Approximation = 0.966943632$	10885	1.05 – 1.45 *sec*
9	$Bound(9)_{lower} = Bound(8)_{upper}$ formed at $t = 1.45$ *sec* $Approximation = 0.966943632$	$Bound(9)_{upper} = \{X_{0,1,2\ldots.21622}\}$ formed at $t = 1.91$ *sec* $Approximation = 0.9554756462$	21623	1.45 – 1.91 *sec*
10	$Bound(10)_{lower} = Bound(9)_{upper}$ formed at $t = 1.91$ *sec* $Approximation = 0.955475646$	$Bound(10)_{upper} = \{X_{0,1,2\ldots.39731}\}$ formed at $t = 2.42$ *sec* $Approximation = 0.9423703393$	39732	1.191 – 2.42 *sec*
:	:		:	:
401	$Bound(847)_{lower} = Bound(846)_{upper}$ formed at $t = 7.69$ *sec* $Approximation = 0.944723401$	$Bound(847)_{upper} = \{X_{0,1,2\ldots.2367589}\}$ formed at $t = 8.46$ *sec* $Approximation = 0.938679241$	2367590	7.69 – 8.46 *sec*

(See section G.4 in Appendix G for details of bounds for *cap1 pathway*).

In the *hog1 pathway* in Figure 7.16, with the exploration of the set of 1595 states, the $Bound(1)_{upper} = \{X_{0,1,2.....1595}\}$ is formed at 1.0 *sec* carryies 1596 states. Some states were bunked since the formation of the first bound resulting in approximation error reaching $3.79e-07$ at 2.0 *sec* and, after 8.0 *sec*, no new states are added in the domain until t_f. At t_f, the *LAS* ends with a domain defined by $Bound(200)_{upper} = \{X_{0,1,2.....15979}\}$ carrying 15980 states with $1.162e-06$ approximation error. A set of nodes $N_1, N_2, \ldots . N_{15980}$ carries these unique states representing the set of $state(n_{15980}) = \left(X_{0,1,2,\ldots 15979}\right)$ to form the state-space of the *hog1 pathway* up to t_f.

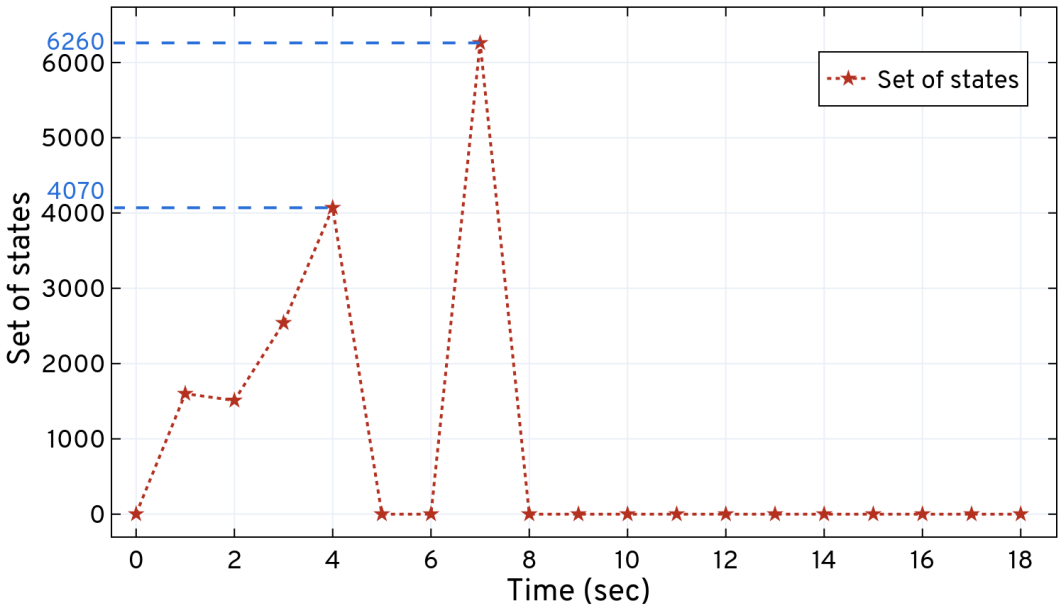


Figure 7.16. The set of states explored for the *hog1 pathway* using the *ISP LAS* method. Based on network of reactions, *ISP LAS* unfolds the state-space pattern to update states in the domain and expands 15980 states up to t_f. ★ shows the time point where the new set of states are explored and updated in the domain.

Table 7.13. Lower and upper bounds of the *hog1 pathway* given by *ISP LAS* trend.

Z	$Bound(Z)_{lower}$	$Bound(Z)_{upper}$	States	Duration
1	$Bound(1)_{lower} = \{X_0\}$ formed at $t = 0.0$ *sec* $Approximation = 1$	$Bound(1)_{upper} = \{X_{0,1,2.....1595}\}$ formed at $t = 1.0$ *sec* $Approximation = 0.9990242863$	1596	0.0 – 1.0 *sec*
2	$Bound(2)_{lower} = Bound(1)_{upper}$ formed at $t = 1.0$ *sec* $Approximation = 0.999024286$	$Bound(2)_{upper} = \{X_{0,1,2.....3101}\}$ formed at $t = 2.0$ *sec* $Approximation = 0.9965421436$	3102	1.0 – 2.0 *sec*
3	$Bound(3)_{lower} = Bound(2)_{upper}$ formed at $t = 2.0$ *sec* $Approximation = 0.996542143$	$Bound(3)_{upper} = \{X_{0,1,2.....5641}\}$ formed at $t = 3.0$ *sec* $Approximation = 0.9954889199$	5642	2.0 – 3.0 *sec*
4	$Bound(4)_{lower} = Bound(3)_{upper}$ formed at $t = 3.0$ *sec* $Approximation = 0.995488919$	$Bound(4)_{upper} = \{X_{0,1,2.....9715}\}$ formed at $t = 4.0$ *sec* $Approximation = 0.9953457544$	9716	3.0 – 4.0 *sec*
5	$Bound(5)_{lower} = Bound(4)_{upper}$ formed at $t = 6.0$ *sec* $Approximation = 0.995345754$	$Bound(5)_{upper} = \{X_{0,1,2.....15979}\}$ formed at $t = 7.0$ *sec* $Approximation = 0.99711969$	15980	6.0 – 7.0 *sec*
:	:	:	:	:
200	$Bound(200)_{lower} = Bound(4)_{upper}$ formed at $t = 6.0$ *sec* $Approximation = 0.985246467$	$Bound(200)_{upper} = \{X_{0,1,2.....15979}\}$ formed at $t = 200.0$ *sec* $Approximation = 0.99711969$	15980	6.0 – 200.0 *sec*

(See section G.4 in Appendix G for details of bounds for *hog1 pathway*).

The solution of the CME up to t_f for the domain created by *ISP LAS* for the modules of the integrated model is shown in Table 7.14. In three test runs, the total run time of *ISP LAS* for solving the integrated model comprising several modules takes an average of ≈10722.708 *sec* (2 *hours* 59 *minutes* 11 *sec*) based on the duration of the expansion (see Table 7.14) chosen for the modules, giving average error of 0.034057 at t_f.

Table 7.14. *ISP LAS* expansion output and solution at t_f for the Integrated model based on its different modules.

Modules of the integrated model	Run-time (sec)	Domain (states)	Duration of expansion (sec)	Error at t_f
Transporter, $t_{step} = 0.01$	189.252	78054	3.01	$1.988e-04$
Cat1 pathway, $t_{step} = 0.01$	26.360	38235	3.01	$1.418e-04$
Pentose pathway, $t_{step} = 0.01$	1065.755	20097	5.09	$3.897e-04$
Thioredoxin, $t_{step} = 0.0001$	5450.078	6641483	0.0006	$2.144e-06$
Protein-thiol, $t_{step} = 0.0001$	1351.363	245701	0.04	$1.918e-04$
Cap1 pathway, $t_{step} = 0.01$	2625.657	2367590	8.46	$3.348e-04$
Hog1 pathway, $t_{step} = 1.0$	14.243	15980	200	$1.162e-06$

The response of $ISP\ LAS$ given in Figure H. 1 to Figure H. 7 of Appendix H for all the modules is the probabilities of the system bunked at t' while expansion (w.r.t approximation) when number of states increases with the expansion and with that, $ISP\ LAS$ produce minimal average error of order 10^{-4}, as given in Table 7.14 and Figure H. 1 to Figure H. 7 of Appendix H.

The conditional probabilities of the system's species at different times are given in Figure 7.17 to Figure 7.23 describes the dynamic nature of the system. At $t = 0.2$ *sec* in Fig (1) of Figure 7.17, the probability of x_0 (extra-cellular H_2O_2) remains high (0.09112), which tends to decrease with time 0.03655 (at $t = 1.2$ *sec*) and 0.023028 (at $t = 3.0$ *sec*) and shifts to the right when catalase degrades this extra-cellular H_2O_2 through diffusion into cytosol. However, the rate of decrease of the x_1 (intra-cellular H_2O_2) probability over time is less than x_0 (extra-cellular H_2O_2) because its rate of diffusion into the medium is based on the surface area of the cell and at the same time, R_4 produces x_1 which partially maintains its steady state value throughout t_f as seen in Fig (2) of Figure 7.17.

At $t = 0.2$ *sec*, the probability of x_2 $(Cat1)$ remains at 0.1156 which decreases to 0.04972 at 1.2 *sec* and tends to shift to right as seen in Fig (1) of Figure 7.18. However, a further decline in the probability shifts the distribution to the left at 3.0 *sec*. The continouous decline in probability shows that the production of x_2 $(Cat1)$ decreases its contribution of states to the domain over time. The high probability in Fig (2) of Figure 7.18 shows that the x_{33} $(CAT1)$ is produced at a high basal rate (see Table 7.5) initially and, eventually, this rate decreases with time, as evident by the decreasing probability distribution in Fig (2) of Figure 7.18. It is important to note that as R_{64} translates this x_{33} into x_2, the rate of production of x_2 also decreases with time, as seen in Fig (1) of Figure 7.18. The distribution in Fig (2) of Figure 7.18 tends to remain at the left, which also suggests that the x_{33} death process quickly degrades its counts but has a high probability of the states.

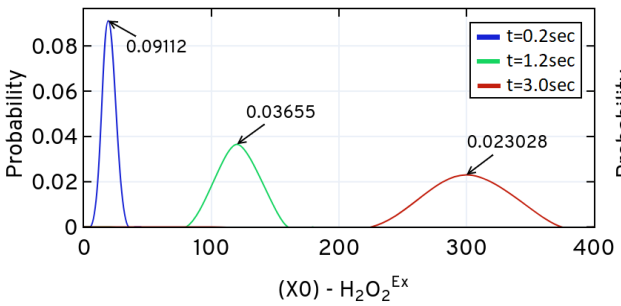


Fig (1). Probability of $H_2O_2^{Ex}$ over t_f

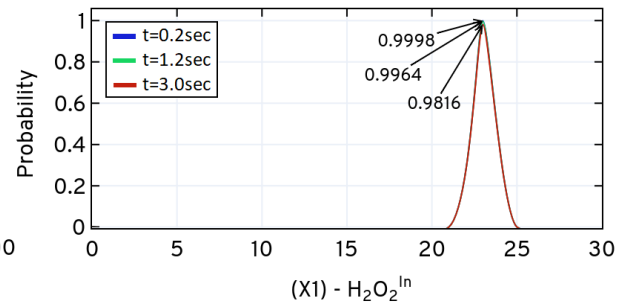


Fig (2). Probability of $H_2O_2^{In}$ over t_f

Figure 7.17. Conditional probability of the species x_0 (extra-cellular H_2O_2), x_1 (intra-cellular H_2O_2) evaluated at $t = 0.2$ *sec*, 1.2 *sec* and 3.0 *sec* with $t_{step} = 0.01$ using *ISP LAS*.

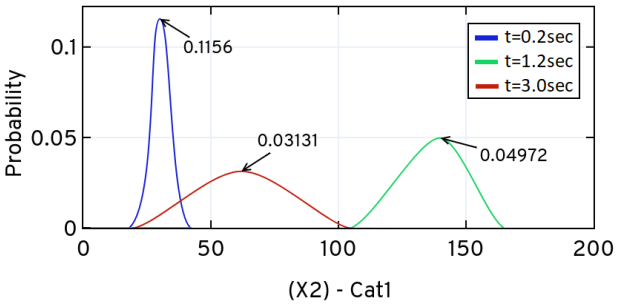


Fig (1). Probability of $Cat1$ over t_f

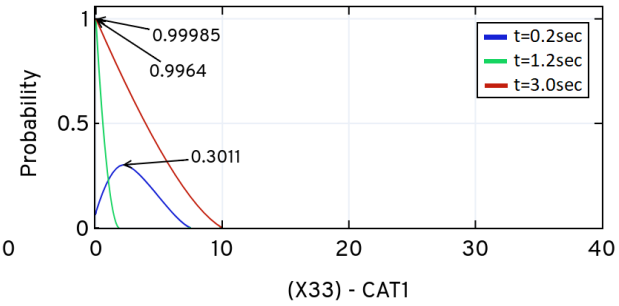


Fig (2). Probability of $CAT1$ over t_f

Figure 7.18. Conditional probability of the species x_2 ($Cat1$), x_{33} ($CAT1$ or $CAT1\ mRNA$) evaluated at $t = 0.2$ *sec*, 1.2 *sec* and 3.0 *sec* with $t_{step} = 0.01$ using *ISP LAS*.

At $t = 1.5$ *sec*, the probability of x_{20} ($NADPH$) remains at 0.1628, which then decreases to 0.0115 at $t = 3.2$ *sec* and the distribution tends to shift to the right as seen in Fig (1) of Figure 7.19, when x_{20} is used to reduce the x_4 ($GSSG$) and x_{17} ($Trr1^{Ox}$). Simultaneously, the probability of x_{20} further declines when it moves through first order decay in R_{97}. The distribution of x_{41} in Fig (2) of Figure 7.19 tends to remain at the left with almost similar probabilities for all time points.

Figs (1) to (11) of Figure 7.20 show the dynamic nature of the species in *thioredoxin*. At $t = 0.00028$ *sec*, the species x_{12}, x_{14}, x_{16}, x_{20}, x_{39} has the highest probability but this tends to decrease over time as it approaches $t = 0.00042$ *sec* and then it further decreases at $t = 0.00062$ *sec*, as seen in Figs (1), (2), (4), (8), (9) of Figure 7.20. When x_{18} is reduced, initially it increases the counts of x_{12} and x_{19}, and keeps the x_{19} probability distribution constant over time; however, the probability of x_{12} declines. Here, R_{17} and R_{18} oxidise x_{14} and detoxifies x_1, respectively, keeping the x_{15} probability high over time. Further, the probability distribution of x_{17} slightly declines through x_{20} via R_{21}, which releases $NADP^+$ from the system. The individual species, x_{15}, x_{17}, x_{18}, x_{19}, x_{20}, x_{38} have quite similar distributions with high probabilities for all time points, $t = 0.00028$ *sec*, 0.00042 *sec*, 0.00062 *sec*. Moreover, the dynamic nature of *thioredoxin* is described by x_{12}, x_{14}, x_{16}, x_{20}, x_{39} due to their variable probabilities.

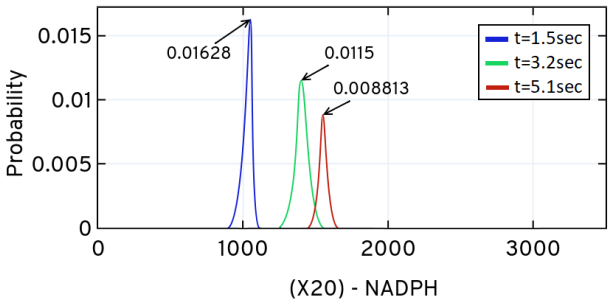


Fig (1). Probability of $NADPH$ over t_f

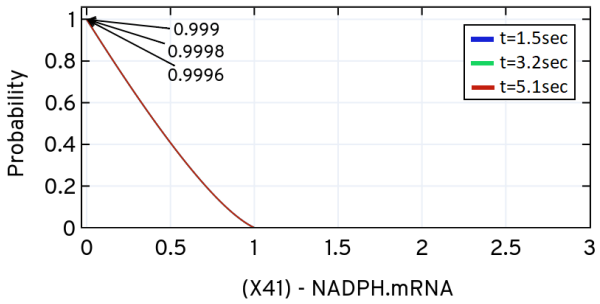


Fig (2). Probability of $NADPH$. mRNA over t_f

Figure 7.19. Conditional probability of the species x_{20}, x_{41} evaluated at $t = 1.5$ *sec*, 3.2 *sec* and 5.1 *sec* with $t_{step} = 0.01$ using *ISP LAS*.

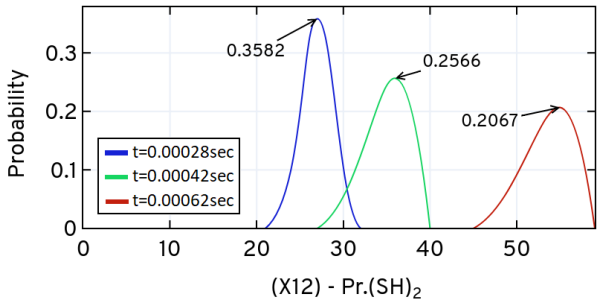


Fig (1). Probability of $Pr.(SH)_2$ over t_f

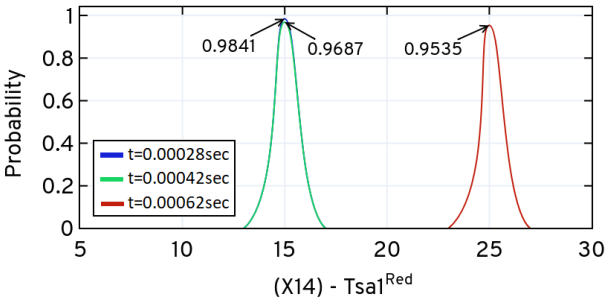


Fig (2). Probability of $Tsa1^{Red}$ over t_f

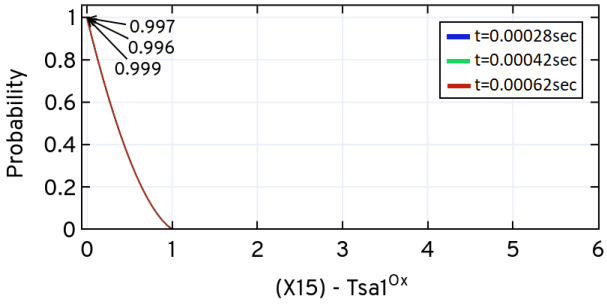


Fig (3). Probability of $Tsa1^{Ox}$ over t_f

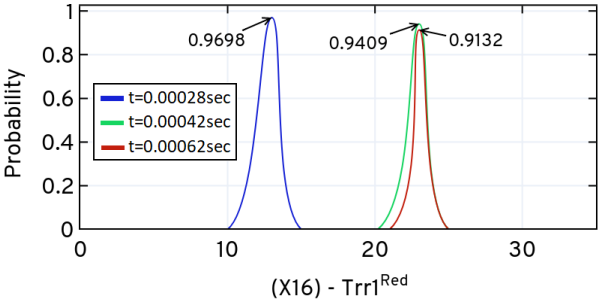


Fig (4). Probability of $Trr1^{Red}$ over t_f

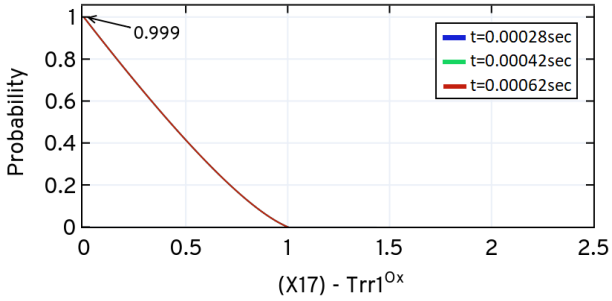


Fig (5). Probability of $Trr1^{Ox}$ over t_f

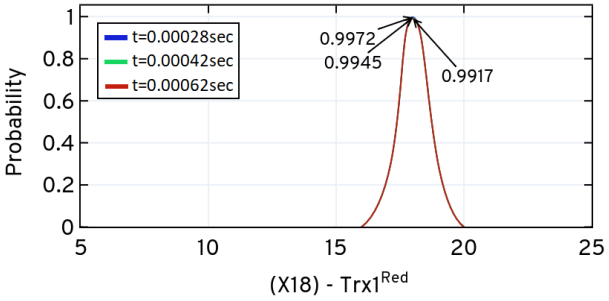


Fig (6). Probability of $Trx1^{Red}$ over t_f

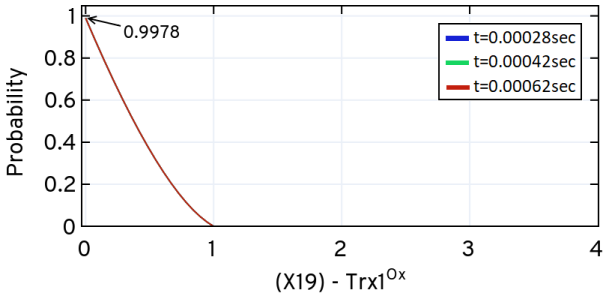


Fig (7). Probability of $Trx1^{Ox}$ over t_f

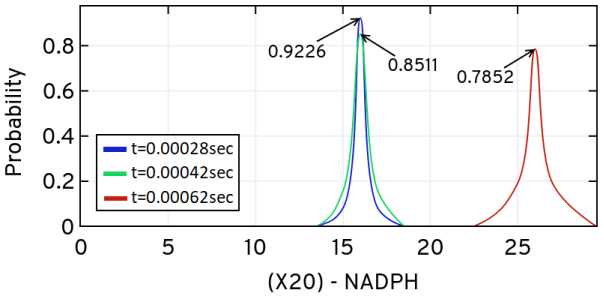


Fig (8). Probability of $NADPH$ over t_f

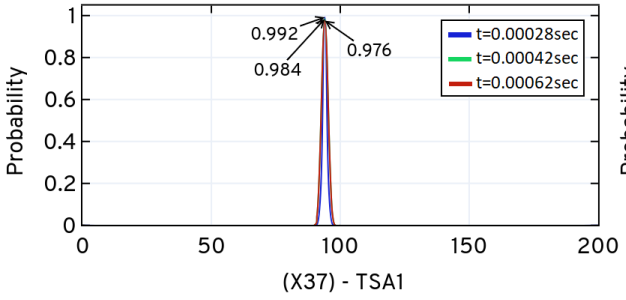


Fig (9). Probability of $TSA1$ over t_f

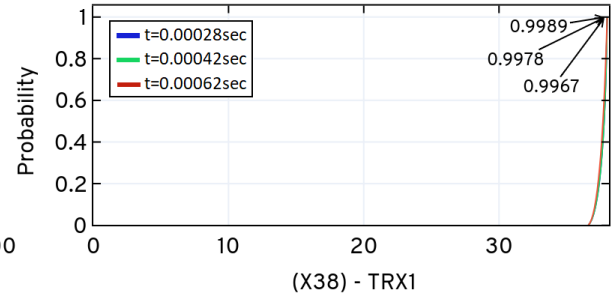


Fig (10). Probability of $TRX1$ over t_f

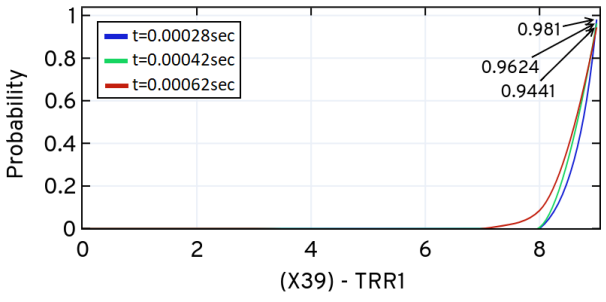


Fig (11). Probability of $TRR1$ over t_f

Figure 7.20. Conditional probabilities of species x_{12}, x_{14}, x_{15}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}, x_{37}, x_{38}, x_{39} evaluated at $t = 0.00028$ *sec*, 0.00042 *sec* and 0.00062 *sec* with $t_{step} = 0.0001$ using *ISP LAS*.

Fig (1) and (2) of Figure 7.21 shows the dynamic nature of the species x_9, x_{12} present in the *protein-thiol module*. At $t = 0.014$ *sec*, the probability distributions of both the species, x_9, x_{12}, were 0.123 and 0.04279, respectively, due to zero order mass action reactions R_9 and R_{14} which increase the counts of the species. Further, at $t = 0.028$ *sec*, the probability decreases in spite of an increase in the copy counts because x_9 (protein mono-thiols) is oxidised to form the protein, sulfenic acid, whereas, x_{12} (protein di-thiols) is oxidised to form protein disulphide. The probability distribution of both the species tends to shift to the right with an increase in copy counts seen at $t = 0.040$ *sec.* This behaviour of the distribution reveals the dynamic nature of the *protein-thiol module* in the integrated model.

The oxidation and activation of x_{21} into x_{22} decreases its counts and tends to shift the probability distribution to the left, as seen in Fig (1) of Figure 7.22. Initially, none of the x_{22} exists in the active oxidised state but once it is activated through x_{21}, it is reduced to x_{21} which, ultimately, decreases its probability over time. x_{23} is converted to x_{22} through the law of mass action reaction; however, the probability still decreases over time. Because of the reversible nature of the reactions (R_{23}, R_{24}, R_{25}, R_{26}, R_{27}), species x_{21}, x_{22}, x_{23} remains most active in the *cap1 pathway*. When x_{22} is activated, it quickly induces the other species (x_{33}, x_{32}, x_{34}, x_{35}, x_{36}, x_{37}, x_{38}, x_{39}, x_{40}, x_{41}) and this results in a similar probability distribution, as seen in Figs (3) and (4) of Figure 7.22 and the probability of x_{22} declines over time, as seen in Fig (2) of Figure 7.22.

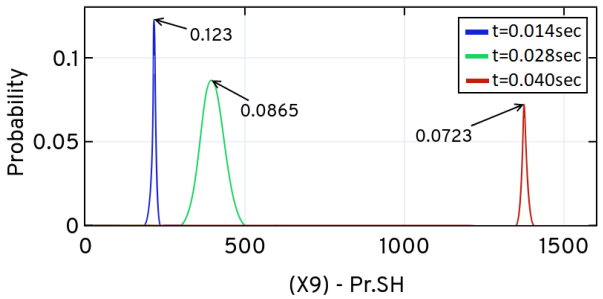


Fig (1). Probability of $Pr.SH$ over t_f

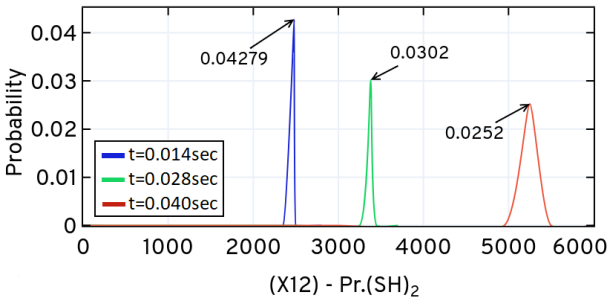


Fig (2). Probability of $Pr.(SH)_2$ over t_f

Figure 7.21. Conditional probability of the species x_9, x_{12} evaluated at $t = 0.014$ *sec*, 0.028 *sec* and 0.040 *sec* with $t_{step} = 0.01$ using *ISP LAS*.

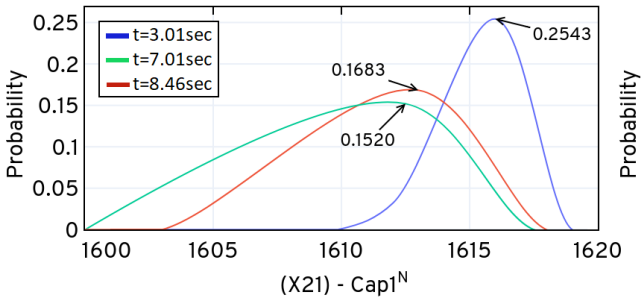


Fig (1). Probability of $Cap1^N$ over t_f

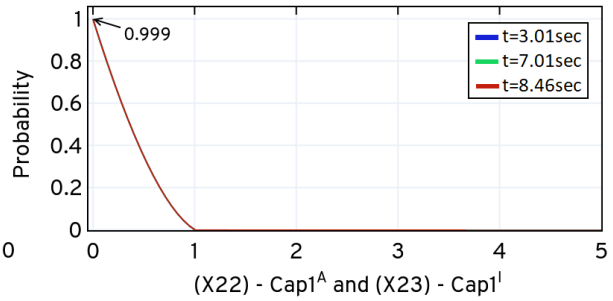


Fig (2). Probability of $Cap1^A$, $Cap1^I$ over t_f

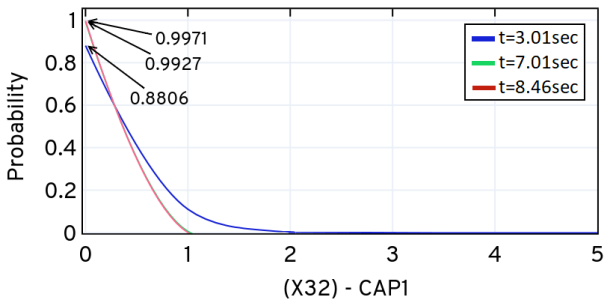


Fig (3). Probability of $CAP1$ over t_f

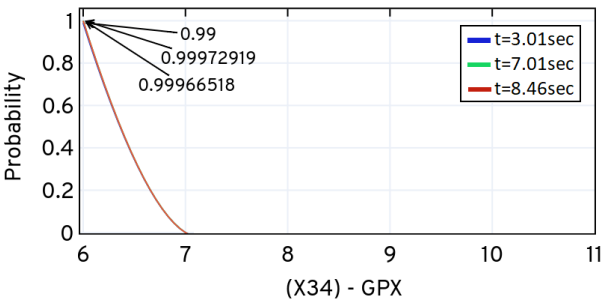


Fig (4). Probability of GPX over t_f

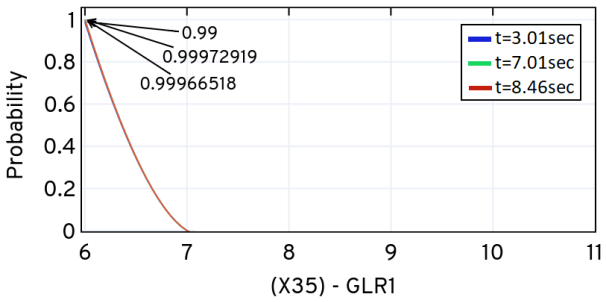


Fig (5). Probability of $GLR1$ over t_f

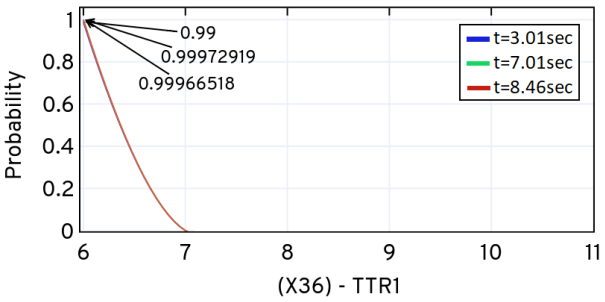


Fig (6). Probability of $TTR1$ over t_f

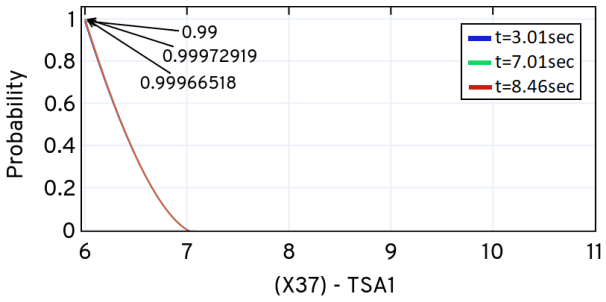


Fig (7). Probability of $TSA1$ over t_f

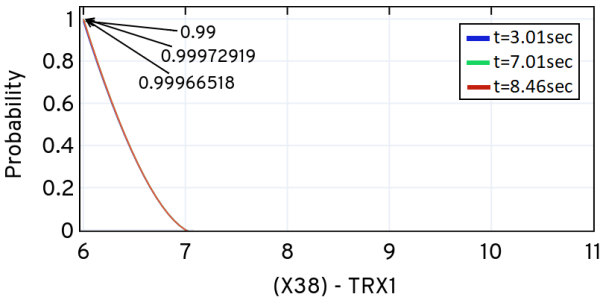


Fig (8). Probability of $TRX1$ over t_f

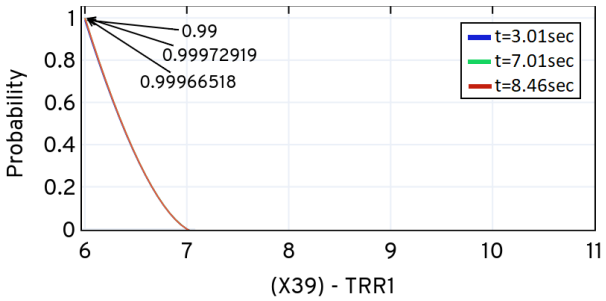


Fig (9). Probability of $TRR1$ over t_f

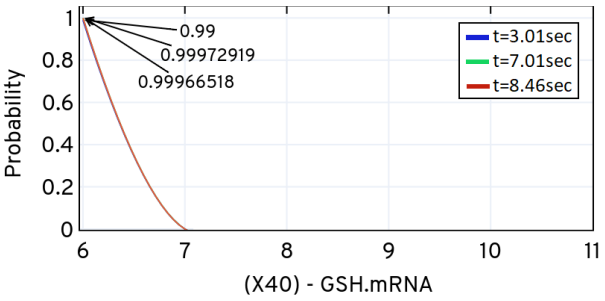


Fig (10). Probability of $GSH.$ mRNA over t_f

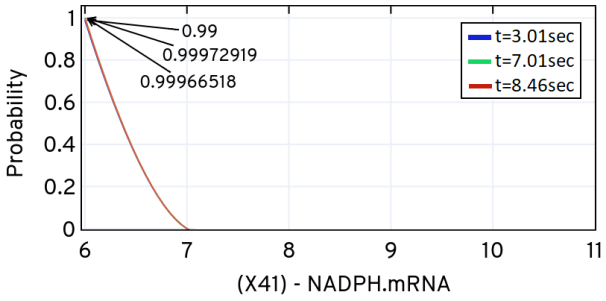


Fig (11). Probability of $NADPH.$ mRNA over t_f

Figure 7.22. Conditional probability of the species x_{21}, x_{22}, x_{32}, x_{34}, x_{35}, x_{36}, x_{37}, x_{38}, x_{39}, x_{40}, x_{41}, evaluated at $t = 3.01$ *sec*, 7.01 *sec* and 8.46 *sec* with $t_{step} = 0.01$ using *ISP LAS*.

Regardless of the de-activation and de-phosphorylation of x_{24} through phosphatases, its probability distribution increases with time (at $t = 25.0$ *sec*, 50.0 *sec* and 200.0 *sec*), as seen in Fig (1) of Figure 7.23. Similarly, x_{26} is de-activated and de-phosphorylated through phosphatases but its probability distribution remains the same and does not shift over time, as seen in Fig (2) of Figure 7.23. Initially, the probability distribution of x_{28} remains at 0.288 at $t = 25.0$ *sec* and, with the increase in the number of copy counts, its probability distribution increases and tends to shift to the right, as seen in Fig (3) of Figure 7.23. The probability distribution of x_{42} ($SSK2$) and x_{44} ($HOG1$) at $t = 25.0$ *sec* remains at 0.4824 and 0.288, respectively, but decreases with the number of copy counts while tending to shift to the left. It is interesting to see that both x_{28} ($Hog1^N$) and x_{44} ($HOG1$) have similar peak probability distributions. However, the shifts in these distributions are different for both species depending upon the changes in copy counts. The species, x_{25}, x_{27}, x_{28}, x_{29}, x_{30}, x_{31} involved in reversible reactions have similar probability distributions at $t = 25.0$ *sec*, 50.0 *sec* and 200.0 *sec*, as seen in Figs (5), (7), (8), (9), (10), (11) of Figure 7.23, respectively.

The parameters of the modules, involving the probabilities of species in Figure 7.17 to Figure 7.23 become important for the case of oxidative stress as well as under normal conditions. Unitedly, the conditional probabilities of these parameters indicate the stochastic behaviour of the network.

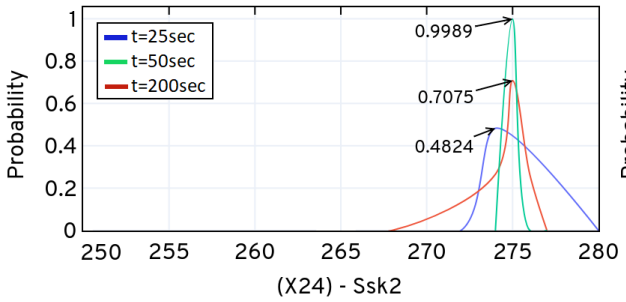


Fig (1). Probability of $Ssk2$ over t_f

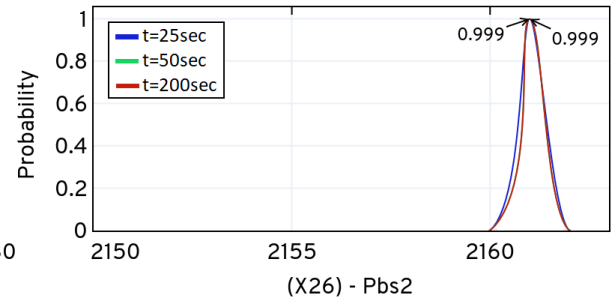


Fig (2). Probability of $Pbs2$ over t_f

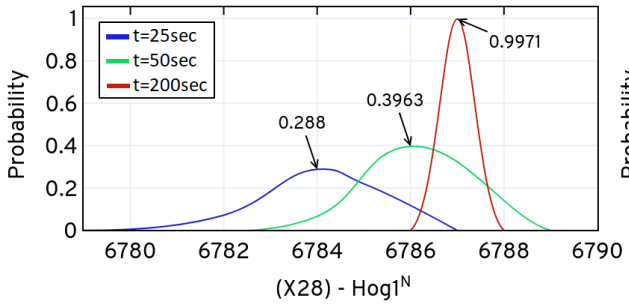


Fig (3). Probability of $Hog1^N$ over t_f

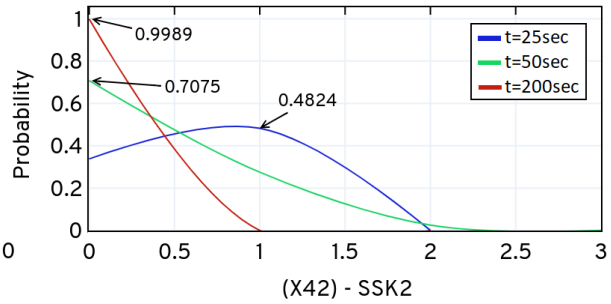


Fig (4). Probability of $SSK2$ over t_f

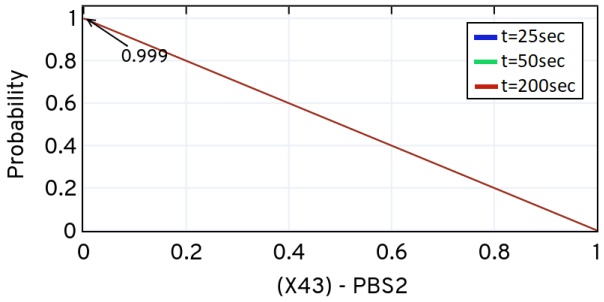


Fig (5). Probability of $PBS2$ over t_f

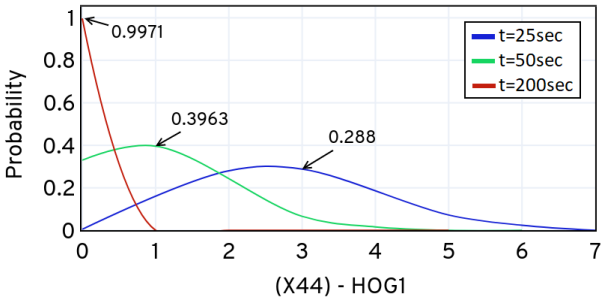


Fig (6). Probability of $HOG1$ over t_f

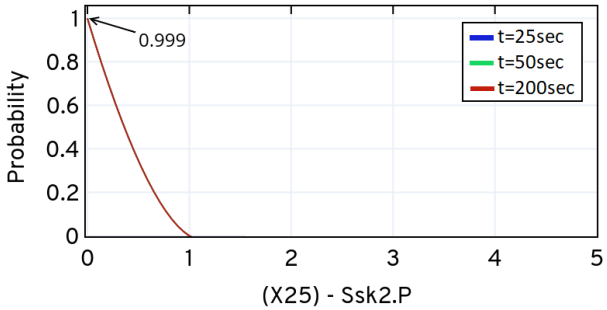


Fig (7). Probability of $Ssk2.P$ over t_f

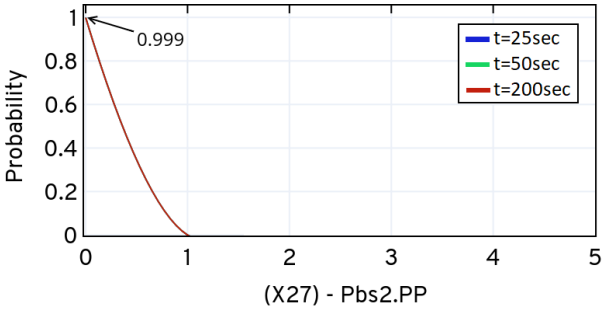


Fig (8). Probability of $Pbs2.PP$ over t_f

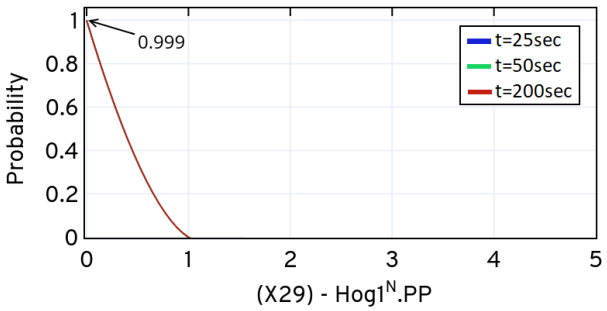


Fig (9). Probability of $Hog1^{N}.PP$ over t_f

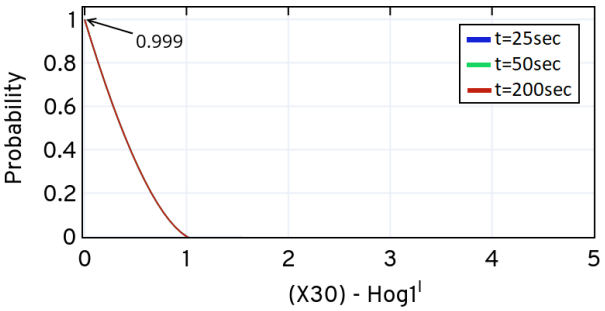


Fig (10). Probability of $Hog1^{I}$ over t_f

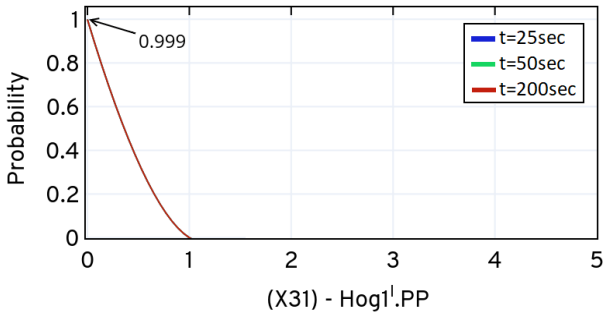


Fig (11). Probability of $Hog1^{I}.PP$ over t_f

Figure 7.23. Conditional probability of the species x_{24}, x_{26}, x_{25}, x_{27}, x_{28}, x_{29}, x_{30}, x_{31}, x_{42}, x_{43}, x_{44}, evaluated at $t = 25.0$ *sec*, 50.0 *sec* and 200.0 *sec* with $t_{step} = 1.0$ using *ISP LAS*.

7.4 Discussion and Summary

In this chapter, we extended the application of our *ISP* method to a very large model with multi-modules compared to applications presented on models in the previous chapter (see Chapter 1) to investigate the performance of *ISP LAS* for the expansion of the state-space and for an approximate solution of the CME based on bounds produced. We simulated the integrated model for the oxidative stress adaptation in the fungal pathogen, *Candida albicans,* modules under the condition of oxidative stress signals to study the number of states contributed by the different modules at different time points. The *ISP* also predicted the conditional probabilities of the protein species (see Figure 7.17 to Figure 7.23) based on the events leading to different complex formation in the modules of the system.

The simulation results from the expansion of states shows that the large explosion of states (Figure 7.5, Figure 7.7, Figure 7.8) does not take place at every time step but in the gaps in several time steps. However, small continuous explosion of states (Figure 7.2, Figure 7.3, Figure 7.4, Figure 7.6) may take place at every time step depending upon the nature of the model. These sudden explosions are due to changes in the level of counts of the species in the presence of an oxidative stress signal, which is useful in gainig insights into the different species' conditional probabilities over time. This response also reveals that the modules *Thioredoxin*, *Cap1 pathway* and *Hog1 pathway* have fewer number of bounds compared to the other modules. The large explosions of states in the modules captured in Figure 7.5, Figure 7.7, Figure 7.8 show the time points where new set of states are explored and updated in the domain. The modules that are more stiff in nature have lower numbers of bounds as well as fewer numbers of explosions (large) in states.

The response of *ISP LAS* also helps us to better understand the potential outcomes of the domains in terms of the bounds defined at different durations of time. Although, the number of states contributed to by the different modules of the integrated model have different domain growth rates at ≈1/4th times of the previous size of the domain in the *transporter module*, ≈8% of the previous size of the domain in the *cat1 pathway*, ≈1.5 to 10 times the previous size of the domain in *thioredoxin*, 3.0% of the previous size of the domain in the *protein-thiol module*; whereas, for other modules the growth is variable.

The solution in Table 7.14 clearly reflects the pros of *ISP LAS* in terms of performance (run-time), number of states (domain size) and the error at t_f for very large integrated model having variable stiffness due to modules. In addition, the checkpoint response of *ISP* in Figs (1) to (6) of Figure 7.9 shows a decline in the initial state probability over time, while the

probabilities of the other states tend to increase over time. In Figure H. 1 to Figure H. 7 of Appendix H, the adaptive approximation error marked by the $ISP\ LAS$ show the major points when the set of states were explored. This also suggests that different modules in the integrated model explosions bring states with lower probabilities that are bunked from the domain for approximation.

Overall, these results provide a clearer picture about the performance of the $ISP\ LAS$ on very large models and the expansion of the state-space of the modules present in the model in presence of oxidative stress signals and how the probabilities of different biochemical species evolve over time. Collectively, the behaviour of the biochemical species present in the different modules defines the dynamic nature of the fungal pathogen *Candida albicans* model. For replication of this experiment, refer to Appendix F and Appendix G.

Chapter 8
Conclusion and Future Directions

In this thesis, we introduced a novel state-space expansion method, called the *intelligent state projection* (*ISP*) method, to enhance expansion and to advance our understanding of the state-space of biochemical reaction networks for the solution of the CME. We also extended the application of *ISP* to large biochemical reaction networks for state-space expansion. In this chapter, we give a general overview of our study in section 8.1, review the major contributions based on the objectives, in section 8.2 and in section 8.3, we provide future work that can follow on from this study.

8.1 Overview of the study

The *first objective* was to propose a data structure to represent and investigate the classes of state-space growth in terms of the number of states. To accomplish this target, we first considered the finite state as a continuous time Markov chain of an homogenous nature to depict the stochastic process of biochemical reaction systems. This finite state Markov chain is then considered as a sample space where we classified the states (*transient* and *communicating*) into two types based on their biochemical reactions (*forward* and *backward* reactions). Further, we visualised this homogenous Markov chain process as equivalent to a Markov chain tree which is a special type of graph (DAG) that represents the *transient* and *communicating* classes of states without cycles but only when the concactenation of cyclic transitions between the set of states is not considered, making them $DAGs$. Most biochemical models of G_{mc} can be visualised as a DAG irrespective of the nature of reactions present in the model. This visualisation of the process creates the *problem state-space* model of biochemical reaction networks where searching is to be done.

To retain the search track in the graph (or tree) for a *problem state-space* expansion, for each node (carrying states), N_i, of the tree, we created a structure consisting of five elements (state, parent state, depth, cost, action) where the state, parent state or node and action are essential elements in the structure for the expansion, while, depth and cost are optional. To deal with the search using such structures, we have adopted the search standards of AI. The expansion of state-space for biochemical systems is then formulated as a problem of an *uninformed search* in AI. For this purpose we employed Markov chains as Markov chain trees or graphs (G_{mc}) as state-space for biochemical systems. We then developed a *Bayesian likelihood node*

projection ($BLNP$) function based on the Bayes' theorem for the projection of nodes carrying probable states to improve the quality and exploit the direction for expansion accordingly. This function enforces the expansion in the direction of reactions that are firing at high rates.

The *second objective* was to develop a new method for the expansion of the state-space suitable for significant types of biochemical reaction networks. To perform a search considering the data structure (1st objective), we have derived the expansion criteria and ceasing criteria based on the vector form of the CME. When the search is performed, the goal state (or end state) is defined by the action (applied via S_{uc} on parent node to reach N_i) implicitly in intervals based on nature (*fast, slow, reversible and irreversible*) for reactions in the system and duration of expansion. Based on the direction of the search and logic, we have categorised our methods of *intelligent State Projection* (ISP) into *latitudinal search* (LAS) and *longitudinal-Latitudinal search* ($LOLAS$) where nodes carrying probable states are managed by a queue and stack, respectively. The steps of both variants were defined separately by embedding the $BLNP$ function to exploit expansion towards a high probability mass. When ISP was tested using toy model, it was observed that both variants have different time complexities and frequency of expansion, as seen in the *states' blueprints* (shows the addition of new sets of states in the domain). We have also applied ISP on different sizes of biological models to show the acceptability of both variants.

The *third objective* was to study the formation of domains at different time points during expansion and note the accuracy of the domain by solution of the CME, while maintaining minimum computational expense. To achieve this objective, we applied the ISP and other existing efficient methods *r-step reachability* ($OFSP$) and SSA (used by SW) on two biological models – catalytic reaction network and dual enzymatic reaction network – to compare the projection size, accuracy and computational expense. In these experiments, the states selected by the ISP to create the optimum domain have high probabilities compared to other methods, domains or number of realisations required in SSA, as suggested by the accuracy of solution for the CME. In addition, the computational expense of both variants of ISP is much less than in other methods. Based on the overall results, both variants of ISP performs better than *r-step reachability* ($OFSP$) and SSA. The response from ISP provided a great deal of information to understand state-space expansion through blueprints, formation of domains at different time steps, as well as the accuracy of the solution.

The *fourth objective* was to study the application of ISP on large biochemical reaction networks. To achieve this objective, we considered: (i) G1/S model involving DNA-damage

signal transduction pathway; (ii) Candida albicans model with oxidative stress feedback. The $ISP\ LOLAS$ was employed for the G1/S model for state-space expansion. The dimension of this model is 28 (the number of biochemical species), which is the factor that defines the difficulty level for solving the CME. $ISP\ LOLAS$ successfully expanded the G1/S model's state-space (with 3409899 states) upto t_f in just 1372 *secs*, while giving the *blueprint* of the states and the formation of bounds at different time intervals. The probabilities of the species plot by $ISP\ LOLAS$ determines the stochastic nature of the G1/S model. The successful application of $ISP\ LOLAS$ on G1/S model shows its acceptability for large, stiff, biochemical reaction networks.

Secondly, the $ISP\ LAS$ was employed for the *Candida albicans* model for state-space expansion. This is an integrated model comprising of several pathways and sub-networks. The dimension of this model is 45 (number of main biochemical species), which is an indicating factor that defines the difficulty level for solving the CME. To understand the complexity of the integrated model, we studied the pathways one-by-one. The $ISP\ LAS$ successfully expanded the *Candida albicans'* model pathway state-space upto t_f giving the *blueprints* of the states and the formation of bounds at different time interval. The probabilities of the species plot by $ISP\ LAS$ determines the stochastic nature of the *Candida albicans'* model. The successful application of $ISP\ LAS$ on *Candida albicans'* model shows its acceptability for stiff, integrated, and large biochemical reaction networks.

8.2 Contributions

The major contributions of this thesis are:

- Visualisation of state-space of a biochemical reaction network as a Markov chain graph or tree with compatible (to elements of graph or tree) infrastructure (data structure) for searching.

- Defining state-space expansion as a problem state-space model of biochemical reaction network to adopt the *artificial intelligence* searching standards to improve the precision of the domain and its solution.

- Development of *Bayesian Likelihood Node Projection* ($BLNP$) function based on Bayes' theorem to enforce the direction of expansion towards states having high probability mass.

- Development of $ISP\ LAS$ and $ISP\ LOLAS$ search strategies based on conditions derived from the vector form of the CME. Both the variants have different time complexities and can be used according to the availability of the experimental platform's random access memory.

- Advance our understanding about the expansion of the state-space process and formation of domain to improve the selection of states at different time points through ISP.

- Application (on high performance computing [HPC] clusters) and acceptability of the ISP for small (for example: catalytic reaction network, dual enzymatic reaction network) and large (for example: G1/S model, *Candida albicans* model) biochemical reaction networks and prediction of the conditional probabilities of the species present in the network.

8.3 Future Directions

We suggest several future directions to follow the current work:

- The proposed ISP method can be extended to the development of the approximation part of the CME. For instance, the Krylov subspace is one of the methods available that can be joined with the ISP to approximate the solution of the large biochemical system for probabilities. Alternatively, new approximation methods can be developed for ISP.

- The *parallel intelligent state projection* ($PISP$) method can be introduced to allocate computation jobs to multiple cores by bringing into service the linearity of the CME problem and employ high performance computing (HPC) clusters. In addition, parallel guiding functions can be developed to further improve the efficiency.

- Based on the *ISP blueprints* of the set of states, *machine learning* (ML) techniques can be used to predict future patterns of the state-space expansion to select probable states to quickly create the projection and approximate the solution directly. The first few sets of states at different time steps can be used as training data for the ML techniques.

- A cache memory system can be implemented for ISP to introduce the feature that will re-initiate the expansion process from where (any intermediate time between $t = 0$ *sec* and t_f) it was interrupted or stopped. This feature will be useful when solving large biochemical models (high dimension) for large t_f and can stop (terminate the program) the process of the expansion and the approximation of the series and resume from the same time point without repeating the whole experiment from $t = 0$ *sec*.

- Advanced guiding function similar to the *Bayesian Likelihood Node Projection* ($BLNP$) can be developed to embed with ISP to further improve the accuracy and computational time.

8.4 Conclusions

This thesis introduced a novel approach ISP to model biochemical systems that address the performance as well as the accuracy problems for the solution of the CME. Variants of ISP (LAS and $LOLAS$) provide systematic ways of expanding the state-space as long as all probable states are not added into the domain, up to the desired t_f. We have effectively demonstrated the effectiveness of our methods with a number of experiments using real biological models: the catalytic reaction system (small model), dual enzymatic reaction system (small model) and G1/S model (large model), and *Candida albicans* (large model with modules). The results and the response of the algorithm shows improvements in the way how different size of biological networks can be modelled (see section 4.3.2, 4.4.2 and Chapter 6, Chapter 7) and even state-space of 3409900 nodes (see Table 6.4) carrying states up to ≈3.5 million can be explored in a reasonable time. The results also show that the domain laid out by ISP had an optimal order and was successful in finding probable states of the system while maintaining good accuracy and computational timing.

We compared the results of the two popular methods: $OFSP$ (*r-step rechability*) and SSA (τ leaps adaptive) based on their accuracy and efficiency. In the comparative study results, we have seen that ISP outperformed the other methods, in case of computational expense, accuracy as well as projection size. The ISP was more effective in terms of predicting the behaviour of the state-space of the system and in performance management, which is a vital step towards modelling large biochemical systems. Unlike other methods, the ISP keeps the lowest states probabilities in the bunker without removing (as removed in $OFSP$) them before its calculation (as removed without calculation in $FSP\ GORDE$) and then computes the probabilities at t without computing large numbers of realizations (as done in SSA).

The diverging nature of the ISP response with respect to $OFSP$ in Figure 5.1 and Figure 5.4 also showed that the solution improved with t and at a higher t_f. For example, in the large model in case study 2, the computation time was $1372\ sec$ and the solution was $3.52\mathrm{e}-06$ at t_f, which was lower compared to the results from the small model (catalytic reaction system). This also shows the compatibility of ISP with the distinct size of the biochemical models.

These examples showed that the ISP is a very promising technique for system's biology. For stiff models, such as the G1/S and *Candida albicans* models, the nature of the ISP yielded

plenty of information and opened opportunity for stochastic analysis of large models. It was often sufficient to employ the ISP to compute the probabilities of the species up to the required time. One could also use ISP to effectively conduct *robustness* and *sensitivity analysis* on the dynamics of biochemical systems and to keep track of what reactions were more active in the system at a particular time. It also determines the complexity of the system by defining the bounds with number states and keep track of the nested state-space patterns (called as ISP model blueprint) that were updated at the end of each step. Outlining the patterns (Figure 4.12, Figure 4.22, Figure 6.5, Figure 7.10(1) to Figure 7.10(7)) of expansion of states to predict the projection folds can be used for updating of the new states.

We anticipate that the current structure of the ISP variants can be efficiently used for different classes and varieties of biological systems, and for computing the configurations with many reactions, as long as the notable part of the state-space density was present between $Bound(Z)_{lower}$ and $Bound(Z)_{upper}$. When there was a high probability of the molecular population of the species undergoing a number of excursions in a fraction of time, then ISP use small t_{step} to capture the moments. Such computations were still challenging in the expansion phase for typical models but can be addressed more closely in combination with the second part of the solution to the CME, i.e. the approximation phase. There are several methods available to address these challenges.

Approximation methods, such as the *Krylov sub-space*, can be used to effectively compute the matrix exponential times of a vector. It was mathematically attractive to aggregate the states or decompose the large sparse matrix into a small dense matrix using the *Krylov sub-space* but they may not be computationally efficient for many in the absence of an efficient domain. On other hand, the performance can also be enhanced by employing the fast math functions compatible with the multicore environment. We have clearly outlined the core ideas behind the ISP variants by highlighting the differences and similarities between them and other methods that cover the computational and theoretical considerations that are essential before any of the approximation methods becomes feasible for an efficient CME solution.

Appendix A. Probability Laws

To deduce the solution of the CME, it is important to have self-evident parameters and then making a series of self-evident deductions that leads to the desired conclusion/result. To address the problem, we shall begin by considering a simple physical experiment that can introduce us the principal rules of the deduction, which we can use it later for the derivation of the Master Equation for chemically reacting system. It will also provide the indication of the environment that we shall use for our derivation.

The three laws of probability will be our self-evident principals of inference (as shown in figure A.1) not only for our present physical experiment but also for the derivation of the CME (see Appendix B). For self-evident premises (Figure A.1. for the dice tossing problem, we shall assume "the fair tossing of the pair of the dice".

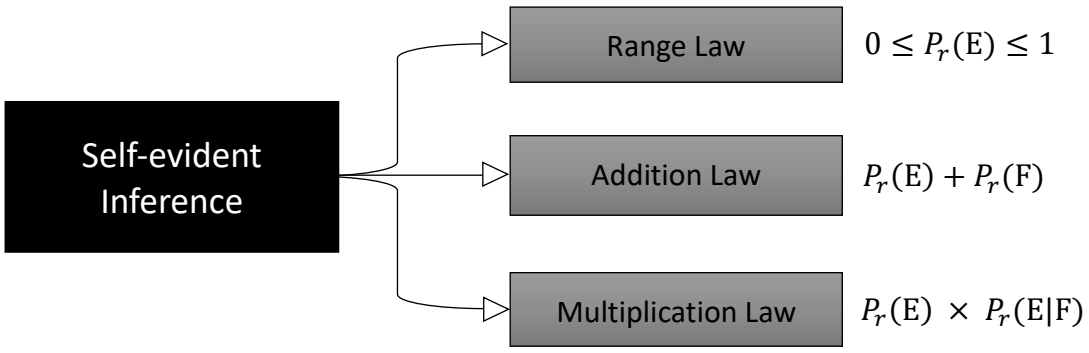


Figure A.1. Self-evident principals of inference showing three laws of probability – range law, addition law, and multiplication law.

Range Law: Probability of occurrence of event E satisfying $0 \leq P_r(\mathrm{E}) \leq 1$.

a) Circumstance $P_r(\mathrm{E}) = 0$, corresponds to E as never occurring.

b) Circumstance $P_r(\mathrm{E}) = 1$, corresponds to E as always occurring.

Addition Law: Probability of occurrence of two events E and F are $P_r(\mathrm{E})$ and $P_r(\mathrm{F})$ respectively, then according to addition law – both the events are mutually exclusive (i.e., they never occur together). The probability of event either E or F will be $P_r(\mathrm{E})$ or $P_r(\mathrm{F}) = P_r(E) + P_r(F)$, where $P_r(\mathrm{F}|\mathrm{E}) = 0$.

Multiplication Law: Probability of occurrence of event E and $P_r(\mathrm{F}|\mathrm{E})$ is the probability of event F given that event E occurs. The probability of event of both "E & F" is $P_r(\mathrm{E}\ \&\ \mathrm{F}) =$

$P_r(\mathrm{E}) \times P_r(\mathrm{F} \mid \mathrm{E})$.

1) Each face of each dice has the same probability of occurrence which is 1/6.

2) Up-face of one dice tells nothing about the up-face of another dice.

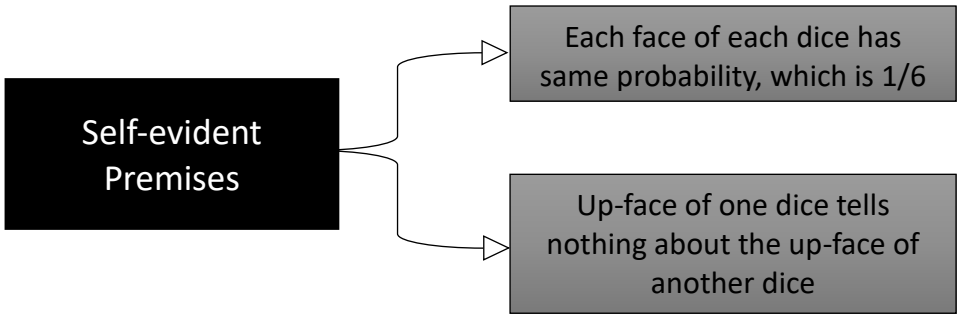


Figure A.2. Self-evident premises for dice experiment showing the probability of event E and F. The occurance of each face of each dice has the same probability, which is 1/6, and the up-face of one dice tells nothing about up-face of another dice.

Some might object that these two premises are not "self-evident" and they need to be proved by solving the appropriate equation for tossing dice experiment. For instance, to deduce the probability of an event in which up-face of dice A plus the up-face of dice B equals to 4. The occurrence of both the events is the occurrence of both the events, either (A = 1 and B = 3) or (A = 2 and B = 2) or (A = 3 and B = 1). By addition law:

$$P_r(\mathrm{A} + \mathrm{B} = 4) = P_r\ (\mathrm{A} = 1\ \&\ \mathrm{B} = 3) + P_r\ (A = 2\ \&\ \mathrm{B} = 2) + P_r\ (\mathrm{A} = 3\ \&\ \mathrm{B} = 1)$$

The first term on the right can be written using multiplication law as:

$$P_r(\mathrm{A} = 1\ \&\ B = 3) = P_r\ (\mathrm{A} = 1)\ \mathrm{x}\ P_r\ (B = 3 \mid \mathrm{A} = 1)$$

Since $P_r(\mathrm{A} = 1) = 1/6$ (by premise 1)

And $P_r(\mathrm{B} = 3 \mid \mathrm{A} = 1) = P_r\ (\mathrm{B} = 3) = 1/6$ (by premise 1 & 2)

Putting the above values in

$$P_r\ (\mathrm{A} = 1\ \&\ \mathrm{B} = 3) = \frac{1}{6} \times \frac{1}{6} = \frac{1}{3}$$

For $P_r(A = 2 \text{ and } B = 2)$ and $P_r(A = 3\ \&\ B = 1)$, we will get the same values, so we conclude that, $P_r(A + B = 4) = \frac{1}{36} + \frac{1}{36} + \frac{1}{36} = \frac{1}{12}$.

Appendix B. Derivation of the CME

To describe the dynamics of stochastic systems, the chemical master equation is used to describe the time evolution of the probability distribution of the population of stochastic systems. For species molecules' geometry level discussion refer to (Gillespie, 1992a). The biochemical system is assumed to be at constant volume and at thermal equilibrium, then

1) The probability of occurrence of only one R_μ reaction in the system in time interval (t, $t + \partial t$) equals to $C_\mu\, h_\mu(n)\partial t \; + \; o(\partial t)$, where $o(\partial t)$ denote the terms that go to zero with ∂t faster than ∂t. C_μ denote the probability, when randomly chosen combination of the reactant species molecules at t will react consistently in next time interval and, h_μ is the number of different combinations of reactant species molecules when there are n_i (where i = 1,…….*N*) number of molecules of species in the system.

Every molecule of reactant species at t is assigned with a unique tag. Then considering any particular combination of the reactant molecules is analogous to the *random selection*. Based on ∂t, the probability of selecting different combination of reactant species molecules at t, that will react in next $t + \partial t$ will be

$$C_\mu\, \partial t\, (1 - C_\mu\, \partial t)^{h_\mu(n)-1} = C_\mu\, \partial t \; + \; o(\partial t), \qquad \text{(B. 1)}$$

By range and addition law (see Appendix A), each has probability $1 - C_\mu\, \partial t$ of not reacting at time $t + \partial t$. While as per multiplication law (see Appendix A), probability that particular of the $h_\mu(n)R_\mu$ reactant combinations react at interval $t + \partial t$

$$h_\mu(n)\, [C_\mu \partial t + o(\partial t)] \; = C_\mu h_\mu(n)\partial t \; + \; o(\partial t), \qquad \text{(B. 2)}$$

Using addition law, we will now calculate the probability such that any of the $h_\mu(n)$ distinct R_μ reactants combinations at time t will react alone at $t + \partial t$. These reactions will be mutually exclusive, so probability of the event will be the sum of the separate probabilities.

2) The probability that no R_j reaction will occur in the system at time interval $(t + \partial t)$ is equals to $1 - \sum_\mu C_\mu h_\mu(n)\partial t \; + \; o(\partial t)$.

In above theorem we noted that each of the distinct $h_\mu(n)$ reactants combination of R_μ reactant molecules of the system at time *t* has probability of $1 - \; C_\mu \partial t$ that it does not react

according to R_μ at time $t + \partial t$. So according to multiplication law, we can write this as

$$(1 - C_\mu\, \partial t)^{h_\mu(n)} = 1 - \, h_\mu(n) C_\mu\, \partial t \, + \, o(\partial t), \qquad \text{(B. 3)}$$

Probability that no R_1 reaction, no R_2 reaction ……. and no R_μ reaction will occur at time interval $t + \partial t$, can be written as

$$\prod_{\mu=1}^{M} \, [1 - h_\mu(n)\, C_\mu\, \partial t \, + \, o(\partial t)] \, = 1 - \sum_{\mu=1}^{M} h_\mu(n)\, C_\mu \partial t \, + \, o(\partial t), \qquad \text{(B. 4)}$$

3) The probability of occurrence of more than one R_μ reaction at a given time interval $t + \partial t$ is $o(\partial t)$.

We are now in position to develop the analytical description of the behaviour of the species population vector. If we fix any initial state at some time to

$$X(t_0) = n_0, \qquad \text{(B. 5)}$$

the deterministic time-evolution equation for $X(t)$ cannot be developed. Instead we shall try to derive the deterministic time-evolution equation for the probability function.

Our strategy to derive the deterministic time-evolution equation for the probability function is to use the three theorems along with addition and multiplication laws of the probability to define.

$P(n, t \mid n_0, t_0)$ = probability that $X(t)= n$, given that $X(t_0) \, = n_0$

In the following paragraph we shall prove by such reasoning that the probability $P(n, t + \partial t \mid n_0, t_0)$ can be written as

$$P(n, t + \partial t \mid n_0, t_0) \, = P(n, t \mid n_0, t_0) \, \times \, (\, 1 - \sum_{\mu=1}^{M} h_\mu(n)\, C_\mu \partial t \, + \, o(\partial t)\,)$$

$$+ \, \sum_{\mu=1}^{M} P\big(n - V_\mu, t \,\big|\, n_0, t_0\big) \, \times \, [\, C_\mu\, h_\mu\big(n - V_\mu\big)\, dt \, + \, o(\partial t)\,] \, + \, o(\partial t) \qquad \text{(B. 6)}$$

We now subtract $P(n, t \mid n_0, t_0)$ from both sides of the above equation and divide through by ∂t, and then take the limit $\partial t \downarrow 0$. Since all $o(\partial t) / \partial t$ terms vanish in this limit, we evidently obtain,

$$\frac{\partial}{\partial t} P(n, t \mid n_0, t_0) = \sum_{\mu=1}^{M} \left[C_\mu \, h_\mu \left(n - V_\mu\right) P\left(n - V_\mu, t \mid n_0, t_0\right) \right.$$

$$\left. - C_\mu \, h_\mu \, (n) \, P(n, t \mid n_0, t_0) \right], \quad \text{(B. 7)}$$

where M = Elementary chemical reaction channels $R_{1,\ldots\ldots\ldots}, R_M$. CME can also be written as

$$\frac{\partial P^{(t)}(X)}{\partial t} = \sum_{\mu=1}^{M} a_\mu\left(X - v_\mu\right) P^{(t)}\left(X - v_\mu\right) - \sum_{\mu=1}^{M} a_\mu(X) \, P^{(t)}(X) \qquad \text{(B. 8)}$$

where $P^{(t)}(X)$ = the probability function representing the time-evolution of the system, given that $t \geq t_0$ and the initial probability is, $P^{(t_0)}(X_0)$, a_μ = chemical reaction propensity of channel $\mu = \{1,2, \ldots . M\}$, and v_μ = stoichiometric vector.

Both the equations (B. 7) and (B. 8) represent the CME. It is a time-evolution equation for the function $P(n, t \mid n_0, t_0)$, for fixed n_0 and t_0, and it is to be solved using initial conditions,

$$P(n, t = t_0 \mid n_0, t_0) = \begin{cases} 1, & if \; n = n_0, \\ 0, & if \; n = n_0, \end{cases}$$

as required by Chemical Master Equation.

Appendix C. Expression for Rate Laws

C.1 Mass Action Based Models

The rate of R_M is proportional to the product of all the reactant concentration (number of molecules). A single step R_M is an elementary reaction if it follows the mass action kinetics. The number of species (only reactants) categorise the type of reaction – monomolecular, bimolecular and trimolecular reactions. For example,

$$A + B \xrightarrow{k} C, \tag{C. 1}$$

is a bimolecular reaction. Where, A and B are reactant species, C is the product and k is the reaction rate parameter. According to mass action law, the rate a_M is proportional to $[A][B]$, and therefore,

$$a_M = k[A][B], \tag{C. 2}$$

where $[A][B]$ represents the product of concentrations of reactant A and B, respectively. (In the following section, we use [species] to denote the concentration of species). The change in concentration of the species A, B, C with time is described by the ordinary differential equations ($ODEs$):

$$\frac{d[A]}{dt} = -k[A][B] = -a_M, \tag{C. 3}$$

$$\frac{d[B]}{dt} = -k[A][B] = -a_M, \tag{C. 4}$$

$$\frac{d[C]}{dt} = k[A][B] = a_M. \tag{C. 5}$$

An elementary reaction can be reversible, if product C transforms into A and B with rate k', such as the following reaction

$$C \xleftrightarrow{k,\ k'} A + B \tag{C. 6}$$

can be treated as two reactions (*forward* and *backward*), where k and k' denote the reaction rate constants, respectively. According to the law of mass action, the forward reaction rate, a_1, is

$$a_1 = k[A][B], \tag{C. 7}$$

and the backward reaction rate, a_2, is

$$a_2 = k^{'}[C]. \tag{C. 8}$$

The change in concentration of these species with time is given by $ODEs$:

$$\frac{d[A]}{dt} = -k[A][B] + k^{'}[C] = -a_1 + a_2 \tag{C. 9}$$

$$\frac{d[B]}{dt} = -k[A][B] + k^{'}[C] = -a_1 + a_2, \tag{C. 10}$$

$$\frac{d[C]}{dt} = k[A][B] - k^{'}[C] = a_1 - a_2. \tag{C. 11}$$

If any reaction has more than one step, then each step will be considered as an elementary reaction. For instance,

$$A \overset{k_1}{\rightarrow} B \overset{k_2}{\rightarrow} C, \tag{C. 12}$$

are two elementary reactions R_1: $A \overset{k_1}{\rightarrow} B$ and R_2: $B \overset{k_2}{\rightarrow} C$, with k_1 and k_2 reaction rate constants respectively. In R_1, the reactant A converts into B, then the a_1, is

$$a_1 = k_1[A]. \tag{C. 13}$$

In R_2, the species B converts into product C, then the a_2, is

$$a_2 = k_2[B]. \tag{C. 14}$$

Then the change in concentration is given by $ODEs$:

$$\frac{d[A]}{dt} = -k_1[A] = -a_1, \tag{C. 15}$$

$$\frac{d[B]}{dt} = k_1[A] - k_2[B] = a_1 - a_2, \tag{C. 16}$$

$$\frac{d[C]}{dt} = k_2[B] = a_2. \tag{C. 17}$$

C.2 Michaelis-Menten Model

C.2.1 Equilibrium approximation

The Michaelis-Menten model is a exemplary example for enzyme kinetic reactions. In these reactions:

$$A + B \overset{k_1;\, k_2}{\longleftrightarrow} AB \overset{k_3}{\longrightarrow} A + P, \tag{C. 18}$$

where, substrate B combines with enzyme A to form the complex AB which represent the forward reaction with k_1 rate constant. The backward reaction will convert this complex AB to substrate B and enzyme A with k_2 rate constant. Further AB forms the product P, which represent the third reaction with k_3 rate constant.

Based on the law of mass action, the change in concentration of these species are given by the $ODEs$:

$$\frac{d[B]}{dt} = -k_1[A][B] + k_2[AB], \tag{C. 19}$$

$$\frac{d[A]}{dt} = -k_1[A][B] + (k_2 + k_3)[AB], \tag{C. 20}$$

$$\frac{d[AB]}{dt} = k_1[A][B] - (k_2 + k_3)[AB], \tag{C. 21}$$

$$\frac{d[P]}{dt} = k_2[AB]. \tag{C. 22}$$

The total concentration of enyzme A_0 is constant and given by

$$[A_0] = [A] + [AB]. \tag{C. 23}$$

AB reaches to equilibrium quickly ($k_1[A][B] = k_2[AB]$) according to Michaelis-Menten (1913) model. Then concentration of the complex is given by

$$[AB] = \frac{k_1}{k_2}[A][B]. \tag{C. 24}$$

Substituting Eq. (C. 23) in Eq. (C. 24) gives,

$$[AB] = \frac{k_1}{k_2}[B]([A_0] - [AB]). \quad \text{(C. 25)}$$

If dissociation constant of the complex AB is represented by $k_{dc} = \frac{k_2}{k_1}$, then Eq. (C. 25) can be rewritten as,

$$[AB] = \frac{[B][A_0]}{k_{dc} + [B]}. \quad \text{(C. 26)}$$

Then reaction rate of the production of P (from Eq. (C. 22)) is given by

$$VL = \frac{d[P]}{dt} = k_3[AB] = k_3 \frac{[B][A_0]}{k_{dc} + [B]}. \quad \text{(C. 27)}$$

If $VL_{max} = k_3[A_0]$ determine the reaction rate (maximum rate) the biochemical system can reach, then, Eq. (C.27) is rewritten as

$$VL = VL_{max} \frac{[B]}{k_{dc} + [B]}. \quad \text{(C. 28)}$$

When $[B] = k_{dc}$, $VL = \frac{1}{2}VL_{max}$. When $k_2 \gg k_3$, then reaction rate of production of P is negatively affected by limits of irreversible reaction.

Appendix D. Dilemma in Selection

Real life example: Suppose a person walking from the car park to the entrance of the fifa world cup stadium and accidently dropped their ticket in a hurry. From behind we can see that the person has long hairs but we are not sure whether they are man or women and what to call out "Excuse me ma'am!" or "Excuse me Sir!". In such situation we have to make a guess.

Now imagine instead this person has reached the entrance and standing in the queue of men's. Knowing this piece of extra information we might make a different guess. $BLNP$ is a way to capture the sense of knowledge about the situation to make better guesses as it treats the probability as beliefs, not frequencies. Now we put numbers to our dilemma at the queue as shown in the Figure D.1. Let us assume that out of 1000 women in the queue 500 have short hair and 50 have long and out of 1000 men in the queue 940 have short hair and 60 have long. In this case we can see that definitely there are more women with long hair than men with long hair, so it is a safe bet to assume it is a women.

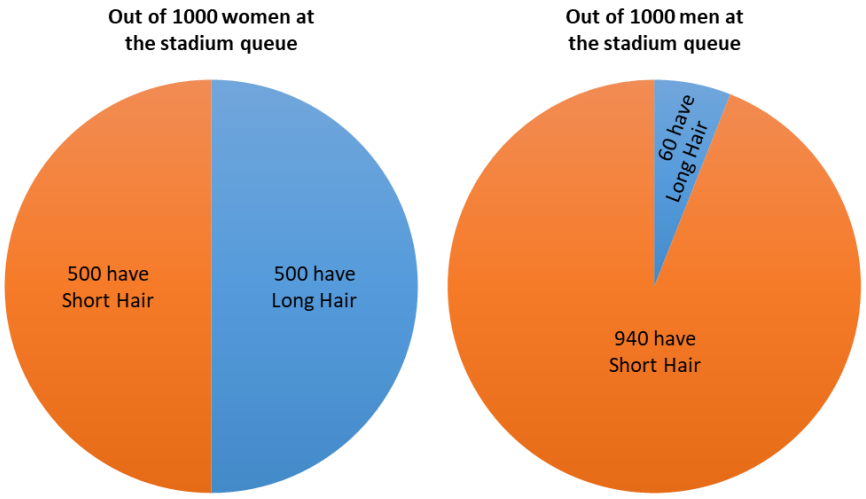


Figure D.1. Showing the ratio of distribution of men and women

Now we just made a subtle assumption that there are about the same number of women and men in the queue of a stadium. This assumption no longer holds when we move to the men's queue. Here let's say there are 20 women out of every 1000 people and 980 are men, as maybe women keeping their partners company. Figure D.2 shows that, there still ten with short hairs and 10 with long hairs, it's still half and half long and short hair but now there are six times as many men with long hair than women.

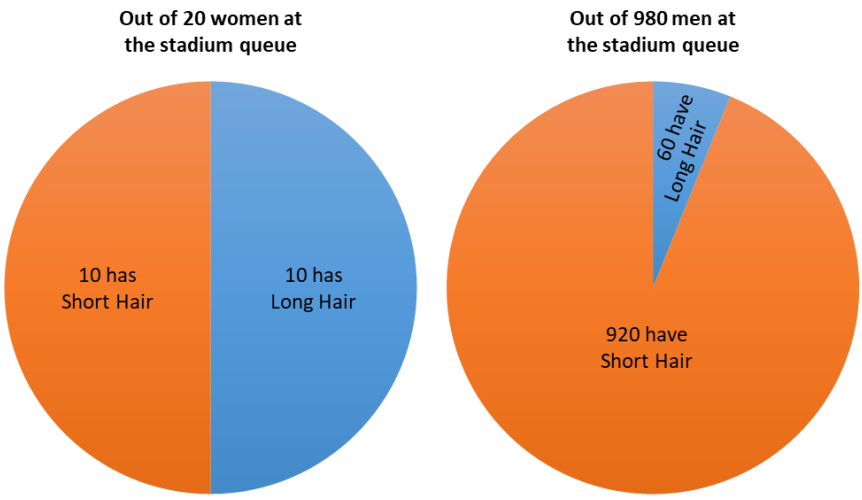


Figure D.2. Showing the distribution of men and women with short and long hairs

Now how we can make safe bet that this person is a man? So to draw this little differently, out of thousand people at the stadium queue overall, we make this assumption explicit that 500 of them are women and 500 of them are men as shown in Figure D.3 (left and right). It shows how different categories break down in the queue for the men's queue then they break down little differently as shown in Figure D.3.

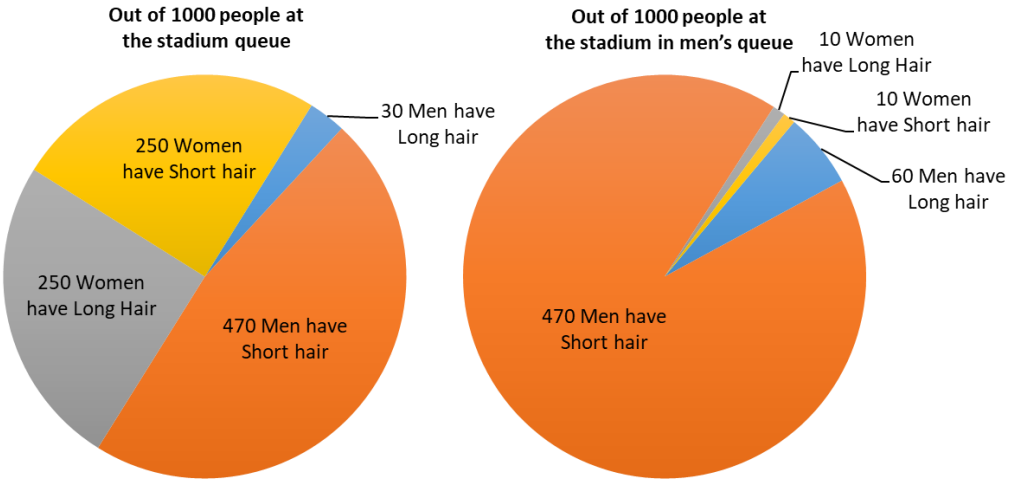


Figure D.3. Showing different distribution fig 42 (left), fig 42 (right) of men and women in a queue of *1000* members

From Figure D.3, the probability that person is a woman and men are given by $P(w) = 0.5$ and $P(men) = 0.5$ respectively, whereas $P(women) = 0.02$ and $P(men) = 0.98$.

Conditional probability: If we know that a person is a woman that is the condition, then the probability that the person has long hair is given by $P(long\ hair|women) = 0.5$. Similarly,

for men the conditional probability that person has long hair is given by $P(long\ hair|men) = 0.06$. For Figure D.2, the conditional probability $P(long\ hair|women) = 0.5$ and $P(long\ hair|men) = 0.12$.

Joint probability: From Figure D.2, the probability that a person is both a woman and has short hair is given by

$$P(women\ with\ short\ hair) = P(women) * P(short\ hair|women)$$

$$= 0.5 * 0.5 = 0.25.$$

Similarly, the probability that a person is both woman and has long hair is given by

$$P(women\ with\ long\ hair) = P(w) * P(long\ hair|women)$$

$$= 0.5 * 0.5 = 0.25.$$

The probability that a person is both man and has short hair is given by

$$P(men\ with\ short\ hair) = P(men) * P(short\ hair|men)$$

$$= 0.5 * 0.94 = 0.47.$$

Similarly, the probability that a person is both man and has long hair is given by

$$P(men\ with\ long\ hair) = P(men) * P(long\ hair|men)$$

$$= 0.5 * 0.06 = 0.03.$$

From Figure D.3, the probability that a person is both a woman and has short hair is given by

$$P(women\ with\ short\ hair) = P(women) * P(short\ hair|women)$$

$$= 0.02 * 0.5 = 0.01.$$

Similarly, the probability that a person is both woman and has long hair is given by

$$P(women\ with\ long\ hair) = P(women) * P(long\ hair|women)$$

$$= 0.02 * 0.5 = 0.01.$$

The probability that a person is both man and has short hair is given by

$$P(men\ with\ short\ hair) = P(men) * P(short\ hair|men)$$

$$= 0.98 * 0.94 = 0.9212.$$

Similarly, the probability that a person is both man and has long hair is given by

$$P(men\ with\ long\ hair) = P(men) *\ P(long\ hair|men)$$

$$= 0.98 * 0.12 = 0.1176.$$

Marginal probability: From Figure D.3, the probability that someone has long hair is given by

$$P(long\ hair) = P(women\ with\ long\ hair) + P(men\ with\ long\ hair)$$

$$=\ 0.01 + 0.1176 = 0.1276$$

and for short hair

$$P(short\ hair) = P(women\ with\ short\ hair) + P(men\ with\ short\ hair)$$

$$=\ 0.01 + 0.9212 = 0.9312$$

If someone has long hairs the probability that they are a man can be calculated by the function

$$P(men \mid long\ hair) = \frac{P(men) * P(long\ hair \mid men)}{P(long\ hair)}$$

$$=\ 0.9216 \text{ or } 92.16\%.$$

Similarly if someone has long hairs the probability that they are a women can be calculated as

$$P(women \mid long\ hair) = \frac{P(women) * P(long\ hair \mid women)}{P(long\ hair)}$$

$$=\ 0.0783 \text{ or } 07.83\%.$$

It is clear from the probability that the ticket dropped by the person is a man. Similarly, such function can be applied to the state-space to choose the probable states carrying most of the probability density and track the active reactions in the biochemical systems.

Appendix E. Complexity Based on Operations

The time complexity of ISP can be evaluated for the worst-case scenario by considering the steps as given in Table 4.2, Table 4.7 and operations as shown in Figure 4.25 by making some simplified assumptions based on size of state-space and operations. To execute each of these steps instructions machine takes discrete amount of time as,

1) In *step 1*, as the initial values are given for the first iteration, it will access the array inputs as:

for 1st iteration it runs for 1 time,

for 2nd iteration it will again run for 1 time,

for I_{tr}th iteration it will run for 1 time.

Total number of times this step will run is the sum of all the run from 1st to I_{tr}th iteration as $1 + 1 + 1 \dots\dots\dots + (I_{tr} - 1) = I_{tr}$, so the time complexity will simply be $O(I_{tr})$. If I_{tr} is equal to the set of nodes in the graph then complexity will become $O(\mathbf{n}_K)$ with the condition that all the nodes are considered as initial node at some point of any iteration, which is true in most of the cases.

2) In *step 2*, for initial node one-time flagging is done for set of states and updates the queue for 1 time as a set. It will continue to flag for other nodes expanded in iterations as:

for 2nd iteration it will again run for 1 time having random number of states,

for I_{tr}th iteration it will run for 1 time having end state of the system within final set of states.

Total number of times this step will run is the sum of all the iterations up to I_{tr}th. On an average if $\mathbf{n}_K$ are carrying $S^{\tilde{\mathrm{N}}}$ states upto d_l level then number of states added to the domain will be $(\mathbf{n}_K{}^{(S^{\tilde{\mathrm{N}}})})$ and complexity will be $O(\mathbf{n}_K{}^{(\mathbf{X}_K)})$. The calculation of Eq. (46) will add constant factor to the complexity, so $\mathbf{n}_K$ carrying set of $\mathbf{X}_K$ will have complexity of $O(\mathbf{n}_K{}^{(\mathbf{X}_K)})$. Most of the time, the condition of Eq. (46) will be true as long as $\mathbf{X}_K$ is getting new states.

3) In *step 3*, sorting the probabilities in every iteration when number of states is growing with exploration leads to the time complexity which depends only on size of $\mathbf{X}_K$, whereas bunking of states with low probabilities from the $queue$ done conditionally in selective iteration at t', so for

for 1st iteration sorting $\mathbf{X}_K$ and bunking of states is done 0 times, so the complexity will be 0,

for 2nd iteration sorting $\mathbf{X}_K$ is done 1 time but bunking is based on Eq. (71), so the complexity will be $O(1^2) + 0 = O(1)$,

for ${I_{tr}}^{\text{th}}$ iteration sorting is done I_{tr} times; so complexity will be $O({I_{tr}}^2)$ and if the bunking is done it will add $O(I_{tr} log I_{tr})$ of complexity. So the complexity of ${I_{tr}}^{\text{th}}$ iteration will become $O({I_{tr}}^2) + O(I_{tr} log I_{tr})$.

The total complexity of the step is the sum of $0 + O(1^2) + 0 +$,……,$+ O({I_{tr}}^2) + O(I_{tr} log I_{tr})$. If any biochemical system has $\mathbf{X}_K$ set of states then the complexity $O(I_{tr} log I_{tr})$ becomes negligible as the size of set of states $\mathbf{X}'_K$ is smaller than the size of the $\mathbf{X}_K$ when Eq. (71) applied conditionally. Therefore, the complexity of the step reduce to $O({I_{tr}}^2)$ if $O(I_{tr} log I_{tr})$ is ignored for most of the cases.

4) In *step 4*, searching adjacent set of nodes $\mathbf{n}_K$ to current node N_i to extend the dictionary $Dict$ will incur $O(1)$ for single reaction for one step as one new node is added carrying new states every time, so complexity for every node for I_{tr} iterations will be $O(\mathbf{n}_K)$ and further computing the $BLNP$ function for $\mathbf{n}_K$ will add $O(\mathbf{n}_K)$ to the complexity. Overall, it will give the complexity of $\mathbb{H}$ for complete search.

5) In step 5, accessing the array (domain) in I_{tr} iterations will cost $O(I_{tr})$ or $O(\mathbf{n}_K) = O(\mathbf{X}_K)$. To commit the updates of propensities of set of states will cost $O(1)$ per value, i.e. for $O(X_0, X_1, \ldots\ldots) = O(\mathbf{X}_K)$ if $\mathbf{n}_K = (X_{0,1,2\ldots\ldots}, \mathrm{d}_l) \notin domain$ then complexity will be $O(1)$ per value in the queue or stack, so for $\mathbf{X}_K$ it will run Z times, $O(I_{tr})$.

6) In step 6, accessing $\mathbf{n}_K = (X_{0,1,2\ldots\ldots}, \mathrm{d}_l)$ for each value from the queue or stack will have same complexity of $O(\mathbf{n}_K)$ for each N_i for I_{tr} iterations, whereas adding the new set of states in the domain will be $O(|{\mathbf{n}_K}^{(\mathbf{X}_K)}|)$.

Combining all the steps complexity,

$$= \underbrace{O(|\mathbf{n}_K|)}_{\textbf{Step 1}} + \underbrace{O(|\mathbf{n}_K{}^{(\mathbf{X}_K)}|)}_{\textbf{Step 2}} + \underbrace{O(|{I_{tr}}^2|) + O(|I_{tr} log I_{tr}|)}_{\textbf{Step 3}} + \underbrace{O(|\mathbf{n}_K|)}_{\textbf{Step 4}} + \underbrace{O(|\mathbf{X}_K|) + O(|I_{tr}|)}_{\textbf{Step 5}} + \underbrace{O(|\mathbf{n}_K|) + O(|\mathbf{n}_K{}^{(\mathbf{X}_K)}|)}_{\textbf{Step 6}}$$

ignoring $O(I_{tr} log I_{tr})$ as it becomes negligible when no states were bunked from the domain or number of states bunked from domain were much lesser than the number of states present in the domain. If estimated number of states $S^{\tilde{\mathrm{N}}}$ creates the perfect domain with set $\mathbf{X}_K$ then total complexity will reduce to

$$= 3 \times O(\mathbf{n}_K) + 2 \text{ x } O(\mathbf{n}_K{}^{(\mathbf{X}_K)}) + O({I_{tr}}^2) + O(I_{tr}) + O(\mathbf{X}_K),$$

therefore, $O\big(Algo(y)\big) = \{0 \leq f(y) \leq c * Algo(y)\ for\ all\ y \geq\ y_0\}$,

where y, y_0 are positive integers. When working with biochemical networks graphs that are large to store explicitly, it is pragmatic to define the overall complexity of the ISP in terms of states and depth of the explored states. As we are interested in end state, so for any iteration if the end state of the system is at $\mathfrak{d}_l$ from initial node N_0 state X_0 then for I_{tr} iterations $O(\mathbf{n}_K{}^{(\mathbf{X}_K)}) = O(\mathfrak{T}^{\mathfrak{d}_l})$, where $\mathfrak{T}$ is the average transitioning factor that defines the average number of transition from N_1 to $N_{N'}$. For small biochemical models every state in the system is crucial as there will be a smaller number of variables that may greatly affect the state-space of the system. In ISP, we increase the depth level by one after the end node has been found to track any reversible reaction that may take the system to previous state and add the end state present with end node, therefore the time complexity of the $ISP\ LAS$ becomes $O(\mathfrak{T}^{\mathfrak{d}_l+1})$. The complexity of the operations carried out in ISP variants for Markov chain tree of a biochemical network is discussed below:

Table E.1. Comparison of the *ISP LAS* average-case Θ and worst-case O *Time-Space* complexity based on operations (*Access, Search, Update, and Remove*) on elements (*Array, Queue, and Tree*).

LAS	**Array**	**Queue**	**Tree**
Time complexity: Average case			
Access	$\Theta(1)$	$\Theta(\mathrm{T}^{\mathrm{d}_l+1})$	$\Theta(log(\mathrm{T}^{\mathrm{d}_l+1}))$
Search	$\Theta(\mathrm{T}^{\mathrm{d}_l+1})$	$\Theta(\mathrm{T}^{\mathrm{d}_l+1})$	$\Theta(log(\mathrm{T}^{\mathrm{d}_l+1}))$
Update	$\Theta(\mathrm{T}^{\mathrm{d}_l+1})$	$\Theta(1)$	$\Theta(log(\mathrm{T}^{\mathrm{d}_l+1}))$
Remove	$\Theta(\mathrm{T}^{\mathrm{d}_l+1})$	$\Theta(1)$	$\Theta(log(\mathrm{T}^{\mathrm{d}_l+1}))$
Time complexity: Worst case			
Access	$O(1)$	$O(\mathrm{T}^{\mathrm{d}_l+1})$	$O(\mathrm{T}^{\mathrm{d}_l+1})$
Search	$O(\mathrm{T}^{\mathrm{d}_l+1})$	$O(\mathrm{T}^{\mathrm{d}_l+1})$	$O(\mathrm{T}^{\mathrm{d}_l+1})$
Update	$O(\mathrm{T}^{\mathrm{d}_l+1})$	$O(1)$	$O(\mathrm{T}^{\mathrm{d}_l+1})$
Remove	$O(\mathrm{T}^{\mathrm{d}_l+1})$	$O(1)$	$O(\mathrm{T}^{\mathrm{d}_l+1})$
Space complexity: Worst case			
Space	$O(\mathrm{T}^{\mathrm{d}_l+1})$	$O(\mathrm{T}^{\mathrm{d}_l+1})$	$O(\mathrm{T}^{\mathrm{d}_l+1})$

Table E.2. Comparison of the *ISP LOLAS* average-case Θ and worst-case O Time-Space complexity based on operations (*Access, Search, Update, and Remove*) on elements (*Array, Stack, and Tree*).

LOLAS	**Array**	**Stack**	**Tree**
Time complexity: Average case			
Access	$\Theta(1)$	$\Theta(\mathrm{T}^{\mathrm{d}_l})$	$\Theta(log(\mathrm{T}^{\mathrm{d}_l}))$
Search	$\Theta(\mathrm{T}^{\mathrm{d}_l})$	$\Theta(\mathrm{T}^{\mathrm{d}_l})$	$\Theta(log(\mathrm{T}^{\mathrm{d}_l}))$
Update	$\Theta(\mathrm{T}^{\mathrm{d}_l})$	$\Theta(1)$	$\Theta(log(\mathrm{T}^{\mathrm{d}_l}))$
Remove	$\Theta(\mathrm{T}^{d})$	$\Theta(1)$	$\Theta(log(\mathrm{T}^{\mathrm{d}_l}))$
Time complexity: Worst case			
Access	$O(1)$	$O(\mathrm{T}^{\mathrm{d}_l})$	$O(\mathrm{T}^{\mathrm{d}_l})$
Search	$O(\mathrm{T}^{\mathrm{d}_l})$	$O(\mathrm{T}^{\mathrm{d}_l})$	$O(\mathrm{T}^{\mathrm{d}_l})$
Update	$O(\mathrm{T}^{\mathrm{d}_l})$	$O(1)$	$O(\mathrm{T}^{\mathrm{d}_l})$
Remove	$O(\mathrm{T}^{\mathrm{d}_l})$	$O(1)$	$O(\mathrm{T}^{\mathrm{d}_l})$
Space complexity: Worst case			
Space	$O(\mathrm{T}\mathrm{d}_l)$	$O(\mathrm{T}\mathrm{d}_l)$	$O(\mathrm{T}\mathrm{d}_l)$

Appendix F. Implementation Details - Environment, Dependencies

F.1 Environment

In this section we discuss about the perquisites, dependencies, and libraries given in following section F.2, must be installed to create the proper environment that enables easier testing and implementation of the presented algorithm(s). The *ssh cmod* with key pair is used with *shell* to connect with the server using Intel®-i3-4005U microprocessor @ 1.70Ghz (2 cores per socket), 4GB DDR3, Ubuntu 16.04.1 desktop 64-bits platform. Files were transferred to AWS® server using *commands,* as well as PuTTy. Before uploading the files using PuTTy, the *ssh* key pair *isp* was generated using PuTTygen (supports RSA, DSA, ECDSA, and Ed25519 keys) to authenticate user through PuTTy and maintain the integrity of the files while transfer. The computing configuration we have used as follows:

Configuration	**Description**
Amazon® Elastic File System (EFS)	➢ Configure file system access ➢ Create mount targets ➢ Configure optional settings a) Performance mode – General Purpose or (2nd option: Max I/O) b) Throughput mode – Bursting or (2nd option: Provisioned) c) Enable encryption – Optional (No)
Storage	Volume type: EBS (1GB memory with 8GB Elastic Block Storage (EBS) type General Purpose SSD (GP2)).
Instance Type	large-m5a (variable ECUs, 16 vCPUs, 2.GHz, AMD EPYC 7571, 64GB memory, EBS only)
Ubuntu 16.04.1	Desktop version running perquisites, dependencies, and essentials

F.2 Dependencies and Libraries

Name	Version / Description
Python	v2.7, v3.7 [1,2] ppa:fkrull/deadsnakes
Python Dev package	Contains header files for the python C - API
GCC Python	Runs code inside GCC as it compiles, exposing internal data structures as a collection of functions and classes [3]
Dependencies and Perquisites	
Date-util	Powerful extension which provides date, time functions [4]
Pyparsing	Used to extract information from textual data (structured) [5]
Libpng-dev	Library of function to create and manipulate image format files like PNG [6]
Pytz	Library allows cross platform time zone calculations accurately [7]
Freetype	Library used to render fonts [8]
Cycler	Provides function to create *base* cycler objects, and class takes care of the iteration logic and composition [9]
Six	Provides function that smoothens the differences between python versions that is compatible with different versions [10]
back ports functools lru cache	Function handles caching mechanism [11]
subprocess 32	Standard library module for python allows you to release new processes [12]
Zlib1g-dev	Library that implements the deflate compression method (as found in gzip etc formats) [13]
Kiwisolver	An efficient C++ implementation of the Cassowary constraint solving algorithm that ranges 10x to 500x faster than original solver typically gaining 40x improvement. Memory savings are greater than 5x times [14]
Intel distribution for python (optional)	package used to boost the computing and operations on elements (optional) [15]
Essentials	
Scipy	V1.1.0 (Builds on Numpy, and used for scientific and technical computing) [16]
Numpy	v1.16.3 (It is a numeric python library which provides fast and efficient operations) [17]
Setuptools	v1.4.2 (Enhance python standard distribution utilities to support projects packaging) [18]
Pip	v19.1.1 (Management system of packages that helps in installation, and management of packages written in python), [for Py2.4, Pip v1.1 is used] [19]
Matplotlib	v1.5.1 (Is a library used for plotting whose numerical extension is Numpy) [20]
Libraries	
Libamd	v2.2.0 [21]
Libblas3gf	shared library that provides basic linear algebra reference implementations [22]
Libc6	C library that contains standard functions that are required to run on Linux/Ubuntu machines [23]
Libgcc1	a library of internal subroutines [24]

Liblapack3gf	library of linear algebra routines 3 [25]
Libumfpack	v5.4.0 [26]
Libstdc++6	Runtime library built with GNU compiler to support C++ programs [27]
Cmepy	Distribute wheels/eggs that includes py libraries to solve cme [28]
gfortan	Invoke Fortran routines like C-code into Python with support for Numeric arrays [29]
Libatlas-sse2-dev	Needed for processors that use sse2 instructions [30]
Python-all-dev	used as a build dependency for other packages to avoid dependencies which are hardcoded (optional) [31]
Pyme	Optimal finite state projection package [32]
Optional	
-U scikit-learn	Analysis, machine learning library [33]
Pandas	library written for the programming (in Python) for analysis and data manipulation [34]
Numba	v0.39.0 High performance python compiler (dependencies: zlib, sqlite, tcl, tk, openssl, mkl, xz, openmp, icc_rt (vs_2015runtime)) [35]
Numexpr	v2.6.5 (dependencies: zlib, sqlite, tcl, tk, openssl, xz, mkl, openmp, icc_rt, numpy) [36]

Links to packages:

[1]. Python 2.7 https://www.python.org/download/releases/2.7/
[2]. Python 3.7 https://www.python.org/download/releases/3.7.0/
[3]. GCC https://gcc-python-plugin.readthedocs.io/en/latest/basics.html
[4]. Dateutil https://pypi.org/project/python-dateutil/
[5]. Pyparsing https://pypi.org/project/pyparsing/
[6]. Libpng-dev http://www.libpng.org/pub/png/libpng.html
[7]. Pytz https://pypi.org/project/pytz/
[8]. Freetype https://pypi.org/project/freetype-py/
[9]. Cycler https://pypi.org/project/Cycler/
[10]. Six https://pypi.org/project/six/
[11]. Backports.functools_lru_cache
https://pypi.org/project/backports.functools_lru_cache/
[12]. Subprocess32 https://pypi.org/project/subprocess32/
[13]. Zlib1g-dev https://packages.ubuntu.com/trusty/zlib1g-dev
[14]. Kiwisolver https://pypi.org/project/kiwisolver/
[15]. Intel distribution for python https://software.intel.com/
[16]. Scipy https://docs.scipy.org/doc/scipy-1.1.0/reference/release.1.1.0.html
[17]. Numpy https://docs.scipy.org/doc/numpy-1.16.3/user/whatisnumpy.html
[18]. Setuptools https://pypi.org/project/setuptools/
[19]. Pip https://pypi.org/project/pip/
[20]. Matplotlib https://sourceforge.net/projects/matplotlib/files/matplotlib/matplotlib-1.5.1/
[21]. Libmad https://packages.ubuntu.com/source/trusty/libmad
[22]. Libblas3gf https://packages.ubuntu.com/search?keywords=libblas3gf

[23]. Libc6 https://packages.ubuntu.com/xenial/libc6-dev-i386
[24]. Libgcc1 https://packages.ubuntu.com/xenial/libgcc1
[25]. Lapack https://launchpad.net/ubuntu/xenial/+source/lapack
[26]. Libumfpack https://packages.ubuntu.com/search?arch=armhf&keywords=umfpack
[27]. Libstdc++6 https://packages.ubuntu.com/xenial/libstdc++6
[28]. Cmepy https://github.com/fcostin/cmepy
[29]. Gfortran https://packages.ubuntu.com/gfortran
[30]. Libatlas-sse2-devel https://pkgs.org/download/libatlas-sse2-devel
[31]. Python-all-dev https://packages.ubuntu.com/trusty/python-all-dev
[32]. Pyme https://github.com/fcostin/cmepy
[33]. Scikit-learn http://scikit-learn.org/stable/documentation.html
[34]. Pandas https://pandas.pydata.org/
[35]. Numba http://numba.pydata.org/
[36]. Numexpr https://pypi.org/project/numexpr/2.4.6/
[37]. Pyssa https://pyssa.readthedocs.io/en/latest/

Appendix G. Replicate Experiments

G.1 Biological Experiments in Literature

Experiments in section 2.5 were performed on:

For example 1: Intel®-i3-4005U microprocessor @ 1.70Ghz (2 cores per socket), 4GB DDR3, Ubuntu 16.04.1 desktop 64-bits platform with python 2.7 for cmepy (see F.2). The model file (tcellhomo.py) can be downloaded from the directory:

> https://github.com/rkosarwal/ispy/others

For example 2: Intel®-i5-7500 CPU microprocessor @ 3.40Ghz, 8GB RAM, Windows 10 desktop 64-bits platform with MATLAB R2018b using high resolution FSP (Munsky et al., 2006) toolkit:

> http://cbe.colostate.edu/~munsky/

For example 3: Intel®-i3-4005U microprocessor @ 1.70Ghz (2 cores per socket), 4GB DDR3, Ubuntu 16.04.1 desktop 64-bits platform with python 2.7 for pyme (see F.2). The model file (mich.py) can be downloaded from the directory:

> https://github.com/rkosarwal/ispy/others

G.2 Biological Experiments using *ISP*

Once environment is setup according to Appendix F, one can now replicate the experiments of *ISP* for the models discussed in section 4.3, 4.4, 5.2, 5.3, 6.3, 7.3 by following the steps:

Step 1: Getting *ISP*

Elementary way to get the *ISP* is to download the TAR or ZIP achieve from GitHub repository:

```
https://rkosarwal.github.io/ispy
```

The repository can be uploaded to the Amazon Web Server in similar way as discussed in section F.1.

Otherwise, *ISP* can be directly forked into Amazon Web Server if Git version control is installed:

```
git clone git://github.com/rkosarwal/ispy.git
```

Step 2: Running the models

All the biochemical system model files are included in the cloned *ISP* directory. To run the model or module use the following command:

```
import ispy.biomodels.<file_name>
```

where, **<file_name>** should be replaced by the model or module file name without .py extension.

For example, if **toy_model.py** be the model file for toy model (discussed in Chapter 4) then to run this model:

```
import ispy.biomodels.toy_model
```

Optional Step: Changing the ODE solver

ISP integrate two types of ODE solver – VODE [1] and LSODA [2]. To change the configuration of the solver based on stiff and non-stiff problems, refer to line 82 and 83 of:

```
ispy/routines/ode.py
```

[1]. http://www.netlib.org/ode/vode.f
[2]. http://www.netlib.org/odepack

G.3 Biological Experiments in Comparative Study

In Chapter 5, we performed the comparative study of algorithms on the carbon-neutral platform of Amazon® Web Service Elastic Computing (EC2) instance type large (m5a) running on HVM (hardware virtual environment) virtualisation with variable ECUs, a multicore environment of 16vCPU @ 2.2GHz, AMD EPYC 7571 running Ubuntu 16.04.1 with the relevant dependencies, a 64GB memory with 8GB Elastic Block Storage (EBS) type General Purpose SSD (GP2) formatted with Elastic File System (EFS). The performance mode was set to general purpose with inputs-outputs per second (IOPS = 100/3000) and a throughput mode of type bursting is set.

For sections 5.2 and 5.3 study: To run the experiments using ISP, refer to section G.2 of Appendix G. To run the experiment using $OFSP$, install the package Pyme (see [32] in section F.2). Model files (catalytic_sys.py) and (dual_enzy.py), respectively, can be downloaded from the directory:

```
https://github.com/rkosarwal/ispy/others
```

and must be placed inside the directory:

```
pyme/Tutorial/
```

of pyme package. To run the experiment using SSA, install the package Pyssa (see [37] in section F.2 under Links to packages). Model files (catalytic_sys_ssa.py) and (dual_enzy_ssa.py), respectively, can be downloaded from the directory:

```
https://github.com/rkosarwal/ispy/others
```

G.4 Output Data of Algorithms

The output data of algorithms can be downloaded from the directory:

https://github.com/rkosarwal/ispy/data

Data file (.txt format) contains the run-time output of $ISP\ LAS$ and $ISP\ LOLAS$ for models. It particularly show the details of formation of bounds with number of states at different time points. It also show the probability bunked while expansion and approximation of the Eq. (9).

Appendix H. Case Study 2 Supporting Information

The response of *ISP LAS* given in Figure H. 1 to Figure H. 7 for all the modules is the total probabilities of states bunked at t' from the domain produced by different modules of *Candida albicans* model, when number of states increases with the expansion and with that, *ISP LAS* produce minimal average error of order 10^{-4}, as given in Table 7.14.

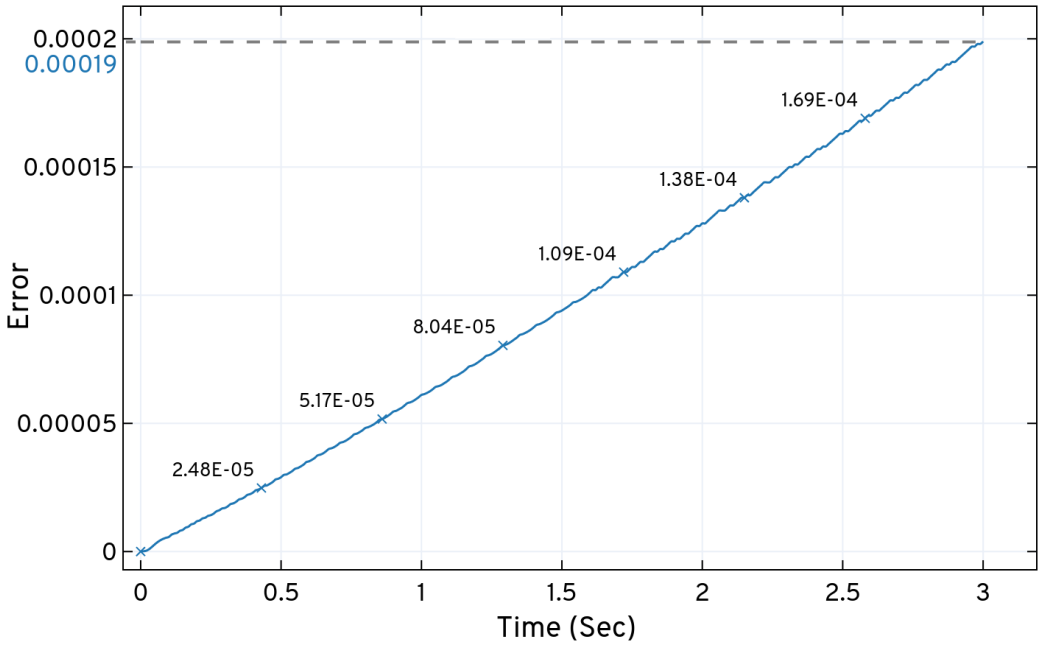


Figure H. 1. Total probability of states bunked at t' from the domain of *transporter module* produced by *ISP LAS* iteration while expansion and solving the CME.

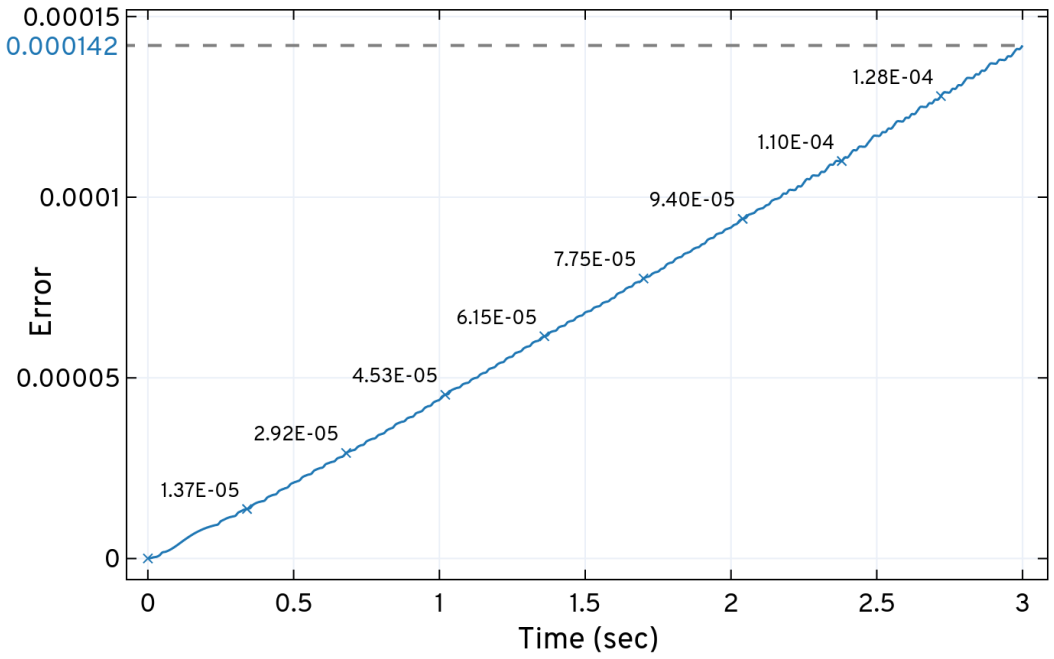


Figure H. 2. Total probability of states bunked at t' from the domain of *catalase (antioxidant)* produced by *ISP LAS* iteration while expansion and solving the CME.

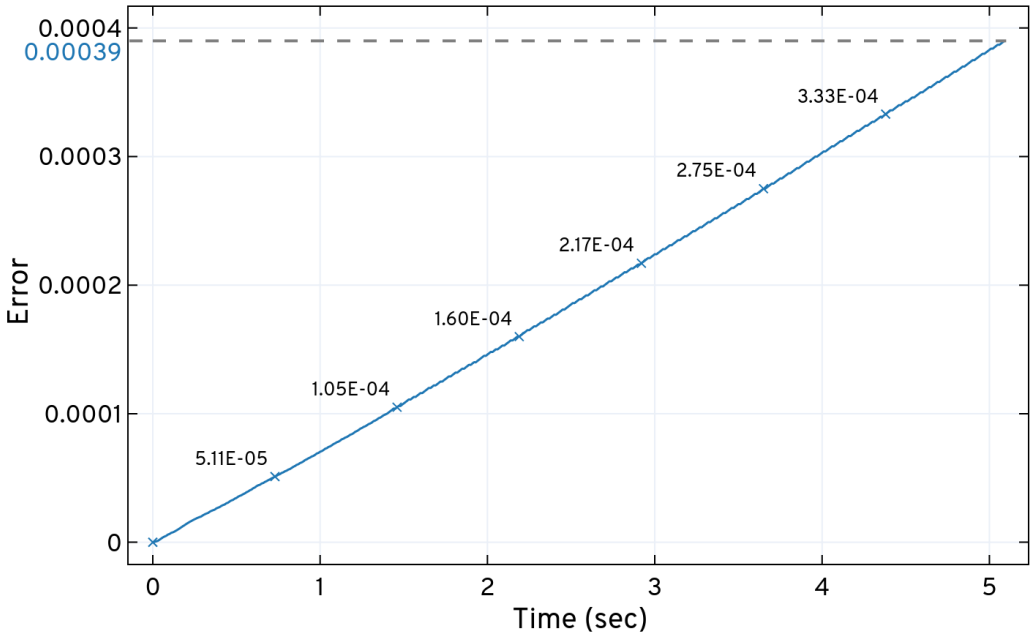


Figure H. 3. Total probability of states bunked at t' from the domain of *pentose phosphate pathway* produced by *ISP LAS* iteration while expansion and solving the CME.

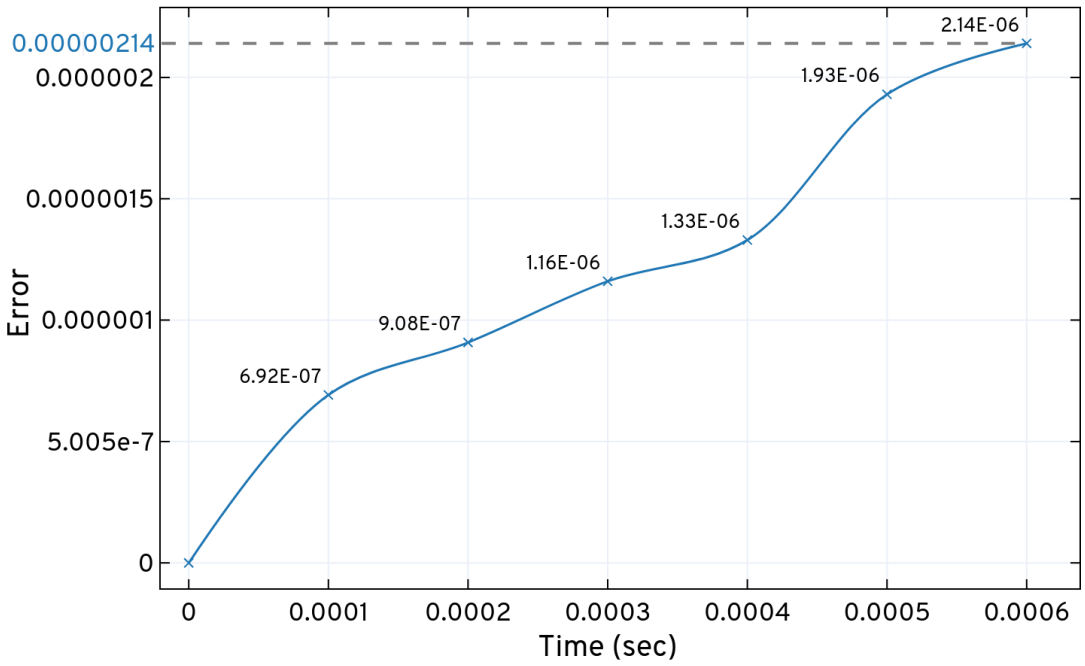


Figure H. 4. Total probability of states bunked at t' from the domain of *thioredoxin (antioxidant)* produced by *ISP LAS* iteration while expansion and solving the CME.

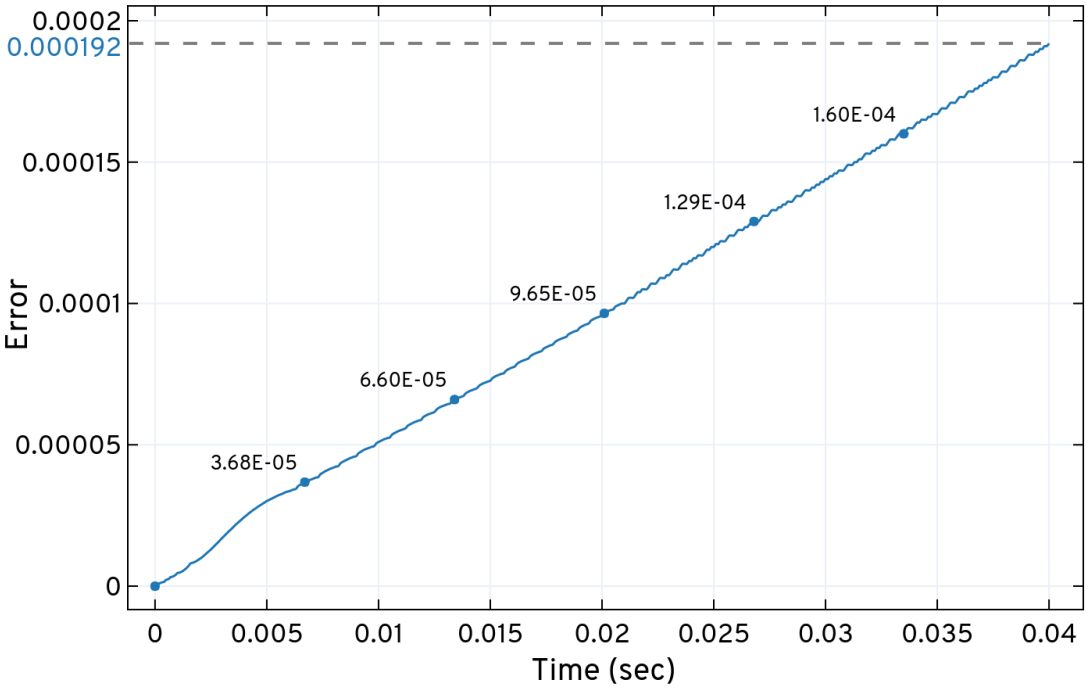


Figure H. 5. Total probability of states bunked at t' from the domain of *protein mono and di-thiols* produced by *ISP LAS* iteration while expansion and solving the CME.

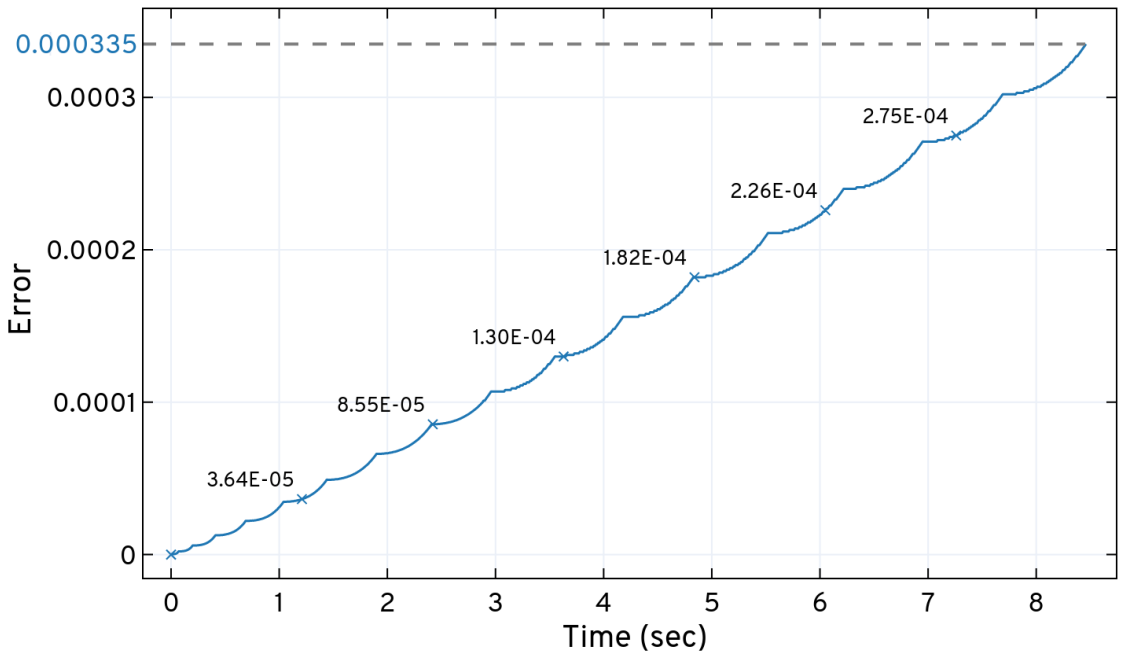


Figure H. 6. Total probability of states bunked at t' from the domain of *cap1 pathway* produced by *ISP LAS* iteration while expansion and solving the CME.

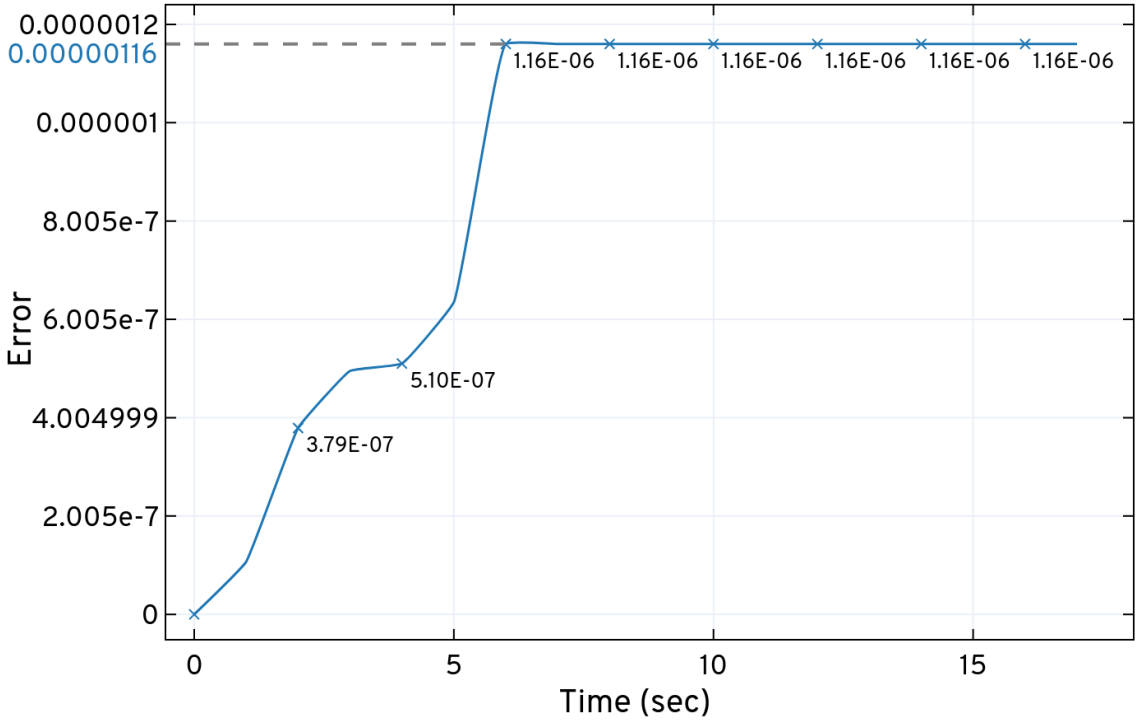


Figure H. 7. Total probability of states bunked at t' from the domain of *hog1 pathway* produced by *ISP LAS* iteration while expansion and solving the CME.

References

Alan Bundy. (1997). *Artificial Intelligence Techniques*. (A. Bundy, Ed.). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-60359-4

Aldous, D. (1992). The Continuum random tree II: an overview. In M. T. Barlow & N. H. Bingham (Eds.), *Stochastic Analysis* (pp. 23–70). Cambridge: Cambridge University Press. https://doi.org/10.1017/CBO9780511662980.003

Alistair J. P. Brown, Ken Haynes, Neil A. R. Gow, J. Q. (2012). *Candida and Candidiasis, Second Edition*. (R. A. Calderone & C. J. Clancy, Eds.). American Society of Microbiology. https://doi.org/10.1128/9781555817176

Anantharam, V., & Tsoucas, P. (1989). A proof of the Markov chain tree theorem. *Statistics and Probability Letters*, *8*(2), 189–192. https://doi.org/10.1016/0167-7152(89)90016-3

Arana, D. M., Alonso-Monge, R., Du, C., Calderone, R., & Pla, J. (2007). Differential susceptibility of mitogen-activated protein kinase pathway mutants to oxidative-mediated killing by phagocytes in the fungal pathogen Candida albicans. *Cellular Microbiology*, *9*(7), 1647–1659. https://doi.org/10.1111/j.1462-5822.2007.00898.x

Barak, Y., Juven, T., Haffner, R., & Oren, M. (1993). Mdm2 Expression Is Induced By Wild Type P53 Activity. *The EMBO Journal*, *12*(2), 461–468. https://doi.org/10.1002/j.1460-2075.1993.tb05678.x

Barr, A., & Feigenbaum, E. (1983). The handbook of artificial intelligence vol. I. *Mathematical Social Sciences*, *4*, 320–324. https://doi.org/10.1016/0165-4896(83)90037-9

Bell, J., & Stevens, B. (2009). A survey of known results and research areas for n-queens. *Discrete Mathematics*, *309*(1), 1–31. https://doi.org/10.1016/j.disc.2007.12.043

Bienert, G. P., Schjoerring, J. K., & Jahn, T. P. (2006). Membrane transport of hydrogen peroxide. *Biochimica et Biophysica Acta (BBA) - Biomembranes*, *1758*(8), 994–1003. https://doi.org/10.1016/j.bbamem.2006.02.015

Branco, M. R., Marinho, H. S., Cyrne, L., & Antunes, F. (2004). Decrease of H 2 O 2 Plasma Membrane Permeability during Adaptation to H 2 O 2 in Saccharomyces cerevisiae. *Journal of Biological Chemistry*, *279*(8), 6501–6506. https://doi.org/10.1074/jbc.M311818200

Brown, G. D. (2011). Innate antifungal immunity: the key role of phagocytes. *Annual Review of Immunology*, *29*, 1–21. https://doi.org/10.1146/annurev-immunol-030409-101229

Brown, G. D., Denning, D. W., Gow, N. A. R., Levitz, S. M., Netea, M. G., & White, T. C. (2012). Hidden killers: human fungal infections. *Science Translational Medicine*, *4*(165), 165rv13. https://doi.org/10.1126/scitranslmed.3004404

Burrage, K., Hegland, M., Macnamara, S., & Sidje, R. (2006). A Krylov-based finite state projection algorithm for solving the chemical master equation arising in the discrete modelling of biological systems. *Proceedings of the Markov Anniversary Meeting*, 1–18.

Cao, Y., Gillespie, D. T., & Petzold, L. R. (2005). The slow-scale stochastic simulation algorithm. *The Journal of Chemical Physics*, *122*(1), 014116. https://doi.org/10.1063/1.1824902

Cao, Y., Gillespie, D. T., & Petzold, L. R. (2006). Efficient step size selection for the tau-leaping simulation method. *Journal of Chemical Physics*, *124*(4), 1–11. https://doi.org/10.1063/1.2159468

Chechik, G., Oh, E., Rando, O., Weissman, J., Regev, A., & Koller, D. (2008). Activity motifs reveal principles of timing in transcriptional control of the yeast metabolic network. *Nature Biotechnology*, *26*(11), 1251–1259. https://doi.org/10.1038/nbt.1499

Chijindu, E. V. C. (2012). Search in Artificial Intelligence Problem Solving, *5*(5), 37–42.

Coqueret, O. (2003). New roles for p21 and p27 cell-cycle inhibitors: a function for each cell

compartment? *Trends in Cell Biology*, *13*(2), 65–70. https://doi.org/10.1016/S0962-8924(02)00043-0

Dasika, G. K., Lin, S. C. J., Zhao, S., Sung, P., Tomkinson, A., & Lee, E. Y. H. P. (1999). DNA damage-induced cell cycle checkpoints and DNA strand break repair in development and tumorigenesis. *Oncogene*, *18*(55), 7883–7899. https://doi.org/10.1038/sj.onc.1203283

Devi, S. G., Selvam, K., & Rajagopalan, S. P. (2011). An Abstract to Calculate Big O Factors of Time and Space Complexity of Machine Code. *International Conference on Sustainable Energy and Intelligent System (SEISCON)*, (Seiscon), 844–847. https://doi.org/10.1049/cp.2011.0483

Diaconis, P., & Efron, B. (1985). Markov Chains Indexed by Trees. *Annals of Statistics*, *13*(3), 845–874.

Dickinson, D. A., & Forman, H. J. (2002). Cellular glutathione and thiols metabolism. *Biochemical Pharmacology*, *64*(5–6), 1019–1026. https://doi.org/10.1016/s0006-2952(02)01172-3

Dinh, K. N., & Sidje, R. B. (2016). Understanding the finite state projection and related methods for solving the chemical master equation. *Physical Biology*, *13*(3). https://doi.org/10.1088/1478-3975/13/3/035003

Dinh, K. N., & Sidje, R. B. (2017). An application of the Krylov-FSP-SSA method to parameter fitting with maximum likelihood. *Physical Biology*, *14*(6), 065001. https://doi.org/10.1088/1478-3975/aa868a

Dyson, N. (1998). The regulation of E2F by pRB-family proteins. - PubMed - NCBI, (617), 2245–2262.

Enjalbert, B., Smith, D. A., Cornell, M. J., Alam, I., Nicholls, S., Brown, A. J. P., & Quinn, J. (2006). Role of the Hog1 Stress-activated Protein Kinase in the Global Transcriptional Response to Stress in the Fungal Pathogen Candida albicans. *Molecular Biology of the Cell*, *17*(2), 1018–1032. https://doi.org/10.1091/mbc.e05-06-0501

Eslahchi, M. R., & Dehghan, M. (2011). Application of Taylor series in obtaining the orthogonal operational matrix. *Computers & Mathematics with Applications*, *61*(9), 2596–2604. https://doi.org/10.1016/j.camwa.2011.03.004

Fahidy, T. Z. (2011). Some Applications of Bayes' Rule in Probability Theory to Electrocatalytic Reaction Engineering. *International Journal of Electrochemistry*, *2011*(1), 1–5. https://doi.org/10.4061/2011/404605

Fallis, A. ., Jorg Liesen, Z. S., Liesen, J., & Strakos, Z. (2012). *Krylov Subspace Methods: Principles and Analysis*. *Journal of Chemical Information and Modeling* (Vol. 53). https://doi.org/10.1017/CBO9781107415324.004

Geva-Zatorsky, N., Rosenfeld, N., Itzkovitz, S., Milo, R., Sigal, A., Dekel, E., … Alon, U. (2006). Oscillations and variability in the p53 system. *Molecular Systems Biology*, *2*, 1–13. https://doi.org/10.1038/msb4100068

Gillespie, D. T. (1977). Exact Stochastic Simulation of Coupled Chemical Reactions. *The Journal of Physical Chemistry*, *81*(1), 2340–2361. Retrieved from https://www.ncbi.nlm.nih.gov/pubmed/17411109

Gillespie, D. T. (1992a). A rigorous derivation of the chemical master equation. *Physica A: Statistical Mechanics and Its Applications*, *188*(1–3), 404–425. https://doi.org/10.1016/0378-4371(92)90283-V

Gillespie, D. T. (1992b). *Markov Processes - An Introduction for Physical Scientists*. Elsevier. https://doi.org/10.1016/C2009-0-22215-X

Gillespie, D. T. (2001). Approximate accelerated stochastic simulation of chemically reacting systems. *The Journal of Chemical Physics*, *115*(4), 1716–1733. https://doi.org/10.1063/1.1378322

Godon, C., Lagniel, G., Lee, J., Buhler, J. M., Kieffer, S., Perrot, M., … Labarre, J. (1998). The H2O2 stimulon in Saccharomyces cerevisiae. *The Journal of Biological Chemistry*,

273(35), 22480–22489. https://doi.org/10.1074/jbc.273.35.22480
Goutsias, J., & Jenkinson, G. (2013). Markovian dynamics on complex reaction networks. *Physics Reports*, *21218*(2), 199–264. https://doi.org/10.1016/j.physrep.2013.03.004
Gursoy, B. B., Kirkland, S., Mason, O., & Sergeev, S. (2012). On the markov chain tree theorem in the max algebra. *Electronic Journal of Linear Algebra*, *26*(12), 15–27. https://doi.org/10.13001/1081-3810.1636
Harrison, R. L., Granja, C., & Leroy, C. (2010). Introduction to Monte Carlo Simulation (pp. 17–21). https://doi.org/10.1063/1.3295638
Haseltine, E. L., & Rawlings, J. B. (2002). Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics. *The Journal of Chemical Physics*, *117*(15), 6959–6969. https://doi.org/10.1063/1.1505860
Helin, K. (1998). Regulation of cell proliferation by the E2F transcription factors. *Current Opinion in Genetics & Development*, *8*(1), 28–35. https://doi.org/10.1016/S0959-437X(98)80058-0
Hiebert, S. W., Chellappan, S. P., Horowitz, J. M., & Nevins, J. R. (1992). The interaction of RB with E2F coincides with an inhibition of the transcriptional activity of E2F. *Genes & Development*, *6*(2), 177–185. https://doi.org/10.1101/gad.6.2.177
Hogervorst, E., Bandelow, S., Combrinck, M., Irani, S. R., & Smith, A. D. (2003). The Validity and Reliability of 6 Sets of Clinical Criteria to Classify Alzheimer's Disease and Vascular Dementia in Cases Confirmed Post-Mortem: Added Value of a Decision Tree Approach. *Dementia and Geriatric Cognitive Disorders*, *16*(3), 170–180. https://doi.org/10.1159/000071006
HUY D. VO. (2017). *Krylov approximation and model reduction methods for solving the Chemical Master Equation*. The University of Alabama.
Ikeda, M. A., Jakoi, L., & Nevins, J. R. (1996). A unique role for the Rb protein in controlling E2F accumulation during cell growth and differentiation. *Proceedings of the National Academy of Sciences*, *93*(8), 3215–3220. https://doi.org/10.1073/pnas.93.8.3215
Iliakis, G., Wang, Y., Guan, J., & Wang, H. (2003). DNA damage checkpoint control in cells exposed to ionizing radiation. *Oncogene*, *22*(37 REV. ISS. 3), 5834–5847. https://doi.org/10.1038/sj.onc.1206682
Iwamoto, K., Tashima, Y., Hamada, H., Eguchi, Y., & Okamoto, M. (2008). Mathematical modeling and sensitivity analysis of G1/S phase in the cell cycle including the DNA-damage signal transduction pathway. *Biosystems*, *94*(1–2), 109–117. https://doi.org/10.1016/j.biosystems.2008.05.016
Jagers, P. (1991). The Growth and Stabilization of Populations. *Statistical Science*, *6*(3), 269–274. https://doi.org/10.1214/ss/1177011694
Jones, M. T. (2005). Estimating Markov Transition Matrices Using Proportions Data: An Application to Credit Risk. *IMF Working Papers*, *05*(219), 1. https://doi.org/10.5089/9781451862386.001
Kabir, M. A., Hussain, M. A., & Ahmad, Z. (2012). Candida albicans : A Model Organism for Studying Fungal Pathogens. *ISRN Microbiology*, *2012*, 1–15. https://doi.org/10.5402/2012/538694
Kaloriti, D., Tillmann, A., Cook, E., Jacobsen, M., You, T., Lenardon, M., … Brown, A. J. P. (2012). Combinatorial stresses kill pathogenic Candida species. *Medical Mycology*, *50*(7), 699–709. https://doi.org/10.3109/13693786.2012.672770
Kholodenko, B. N. (2000). Negative feedback and ultrasensitivity can bring about oscillations in the mitogen-activated protein kinase cascades. *European Journal of Biochemistry*, *267*(6), 1583–1588. https://doi.org/10.1046/j.1432-1327.2000.01197.x
Komalapriya, C., Kaloriti, D., Tillmann, A. T., Yin, Z., Herrero-de-Dios, C., Jacobsen, M. D., … Romano, M. C. (2015). Integrative Model of Oxidative Stress Adaptation in the Fungal Pathogen Candida albicans. *PLOS ONE*, *10*(9), e0137750. https://doi.org/10.1371/journal.pone.0137750

Korf, R. E. (1985). Depth-first iterative-deepening. An optimal admissible tree search. *Artificial Intelligence*, *27*(1), 97–109. https://doi.org/10.1016/0004-3702(85)90084-0
Korf, R. E. (1996). Artificial Intelligence Search Algorithms. *In Algorithms and Theory of Computation Handbook*.
Koza, Z., Matyka, M., Szkoda, S., & Mirosław, Ł. (2012). Compressed Multi-Row Storage Format for Sparse Matrices on Graphics Processing Units, 1–26. https://doi.org/10.1137/120900216
Kubbutat, M. H. G., Jones, S. N., & Vousden, K. H. (1997). Regulation of p53 stability by Mdm2. *Nature*, *387*(6630), 299–303. https://doi.org/10.1038/387299a0
Lahav, G., Rosenfeld, N., Sigal, A., Geva-Zatorsky, N., Levine, A. J., Elowitz, M. B., & Alon, U. (2004). Dynamics of the p53-Mdm2 feedback loop in individual cells. *Nature Genetics*, *36*(2), 147–150. https://doi.org/10.1038/ng1293
Lawlor, O. S. (2013). In-memory Data Compression for Sparse Matrices. *Proceedings of the 3rd Workshop on Irregular Applications: Architectures and Algorithms*, (December), 6:1--6:6. https://doi.org/10.1145/2535753.2535758
Lee, J., Godon, C., Lagniel, G., Spector, D., Garin, J., Labarre, J., & Toledano, M. B. (1999). Yap1 and Skn7 control two specialized oxidative stress response regulons in yeast. *The Journal of Biological Chemistry*, *274*(23), 16040–16046. https://doi.org/10.1074/jbc.274.23.16040
Leone, G., DeGregori, J., Jakoi, L., Cook, J. G., & Nevins, J. R. (1999). Collaborative role of E2F transcriptional activity and G1 cyclindependent kinase activity in the induction of S phase. *Proceedings of the National Academy of Sciences*, *96*(12), 6626–6631. https://doi.org/10.1073/pnas.96.12.6626
Li, G., & Ho, V. C. (1998). p53-dependent DNA repair and apoptosis respond differently to high- and low-dose ultraviolet radiation. *The British Journal of Dermatology*, *139*(1), 3–10. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/9764141
Ling, H., Kulasiri, D., & Samarasinghe, S. (2010). Robustness of G1/S checkpoint pathways in cell cycle regulation based on probability of DNA-damaged cells passing as healthy cells. *BioSystems*, *101*(3), 213–221. https://doi.org/10.1016/j.biosystems.2010.07.005
Ling, Hong. (2011). *Investigation of Robustness and Dynamic Behaviour of G1/S Checkpoint/DNA-damage Signal Transduction Pathway based on Mathematical Modelling and a Novel Neural Network Approach. Thesis. PhD Thesis*. Lincoln University.
MacNamara, S., Bersani, A. M., Burrage, K., & Sidje, R. B. (2008). Stochastic chemical kinetics and the total quasi-steady-state assumption: Application to the stochastic simulation algorithm and chemical master equation. *Journal of Chemical Physics*, *129*(9). https://doi.org/10.1063/1.2971036
Manoukian, E. B. (1986). *Modern Concepts and Theorems of Mathematical Statistics*. New York, NY: Springer New York. https://doi.org/10.1007/978-1-4612-4856-9
Mastny, E. A., Haseltine, E. L., & Rawlings, J. B. (2007). Two classes of quasi-steady-state model reductions for stochastic kinetics. *The Journal of Chemical Physics*, *127*(9), 094106. https://doi.org/10.1063/1.2764480
McQuarrie, D. A. (1967). Stochastic approach to chemical kinetics. *Journal of Applied Probability*, *4*(3), 413–478. https://doi.org/10.2307/3212214
Mikeev, L., Sandmann, W., & Wolf, V. (2013). Numerical approximation of rare event probabilities in biochemically reacting systems. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *8130 LNBI*, 5–18. https://doi.org/10.1007/978-3-642-40708-6_2
Moler, C., & Van Loan, C. (2003). Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, *45*(1), 3–49. https://doi.org/10.1137/S00361445024180
Mouroutsos, S. G., & Sparis, P. D. (1985). Taylor series approach to system identification,

analysis and optimal control. *Journal of the Franklin Institute*, *319*(3), 359–371. https://doi.org/10.1016/0016-0032(85)90056-0
Munsky, B., & Khammash, M. (2006). The finite state projection algorithm for the solution of the chemical master equation. *Journal of Chemical Physics*, *124*(4), 1–13. https://doi.org/10.1063/1.2145882
Munsky, B., & Khammash, M. (2007). A multiple time interval finite state projection algorithm for the solution to the chemical master equation. *Journal of Computational Physics*, *226*(1), 818–835. https://doi.org/10.1016/j.jcp.2007.05.016
Munsky, B., Trinh, B., & Khammash, M. (2009). Listening to the noise: random fluctuations reveal gene network parameters. *Molecular Systems Biology*, *5*(1), 318. https://doi.org/10.1038/msb.2009.75
Murray, J. M., Fanning, G. C., Macpherson, J. L., Evans, L. A., Pond, S. M., & Symonds, G. P. (2009). Mathematical modelling of the impact of haematopoietic stem cell-delivered gene therapy for HIV. *The Journal of Gene Medicine*, *11*(12), 1077–1086. https://doi.org/10.1002/jgm.1401
Nasar, A. A. (2016). The history of Algorithmic complexity. *The Mathematics Enthusiast*, *13*(3), 217–242. Retrieved from http://proxy1.ncu.edu/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=119320485&site=eds-live
Nelson, P. C., & Toptsis, A. A. (1992). Unidirectional and bidirectional search algorithms. *IEEE Software*, *9*(2), 77–83. https://doi.org/10.1109/52.120605
Novozhilov, A. S., Karev, G. P., & Koonin, E. V. (2006). Biological applications of the theory of birth-and-death processes. *Briefings in Bioinformatics*, *7*(1), 70–85. https://doi.org/10.1093/bib/bbk006
Ozer, M., Uzuntarla, M., Perc, M., & Graham, L. J. (2009). Spike latency and jitter of neuronal membrane patches with stochastic Hodgkin–Huxley channels. *Journal of Theoretical Biology*, *261*(1), 83–92. https://doi.org/10.1016/j.jtbi.2009.07.006
Padgett, J. M. A., & Ilie, S. (2016). An adaptive tau-leaping method for stochastic simulations of reaction-diffusion systems. *AIP Advances*, *6*(3). https://doi.org/10.1063/1.4944952
Rinnerthaler, M., Büttner, S., Laun, P., Heeren, G., Felder, T. K., Klinger, H., … Breitenbach, M. (2012). Yno1p/Aim14p, a NADPH-oxidase ortholog, controls extramitochondrial reactive oxygen species generation, apoptosis, and actin cable formation in yeast. *Proceedings of the National Academy of Sciences of the United States of America*, *109*(22), 8658–8663. https://doi.org/10.1073/pnas.1201629109
Roberts, R. M., Cleland, T. J., Gray, P. C., & Ambrosiano, J. J. (2004). Hidden Markov Model for Competitive Binding and Chain Elongation. *The Journal of Physical Chemistry B*, *108*(20), 6228–6232. https://doi.org/10.1021/jp036941q
Rudowsky, I. (2004). Intelligent agents. *Communications of the Association for Information Systems*, *14*(August), 275–290. https://doi.org/10.1016/j.compbiomed.2006.06.016
Santillán, M. (2008). On the Use of the Hill Functions in Mathematical Models of Gene Regulatory Networks. *Mathematical Modelling of Natural Phenomena*, *3*(2), 85–97. https://doi.org/10.1051/mmnp:2008056
Schaber, J., Baltanas, R., Bush, A., Klipp, E., & Colman-Lerner, A. (2012). Modelling reveals novel roles of two parallel signalling pathways and homeostatic feedbacks in yeast. *Molecular Systems Biology*, *8*, 622. https://doi.org/10.1038/msb.2012.53
Schlecht, V. (2014). How to predict preferences for new items. *Investment Management and Financial Innovations*, *5*(4), 7–24. https://doi.org/10.1287
Schulze, J., & Sonnenborn, U. (2009). Yeasts in the Gut. *Deutsches Aerzteblatt Online*. https://doi.org/10.3238/arztebl.2009.0837
Shepherd, D. P., Li, N., Micheva-Viteva, S. N., Munsky, B., Hong-Geller, E., & Werner, J. H. (2013). Counting Small RNA in Pathogenic Bacteria. *Analytical Chemistry*, *85*(10), 4938–4943. https://doi.org/10.1021/ac303792p

Sidje, R. B., & Vo, H. D. (2015). Solving the chemical master equation by a fast adaptive finite state projection based on the stochastic simulation algorithm. *Mathematical Biosciences*, *269*, 10–16. https://doi.org/10.1016/j.mbs.2015.08.010

Sunkara, V. (2013). Analysis and Numerics of the Chemical Master Equation, 1–134. Retrieved from http://www.math.kit.edu/ianm3/~sunkara/media/thesis_sunkara.pdf

Sunkara, V., & Hegland, M. (2010). An optimal finite state projection method. *Procedia Computer Science*, *1*(1), 1579–1586. https://doi.org/10.1016/j.procs.2010.04.177

Tian, T., & Burrage, K. (2004). Binomial leap methods for simulating stochastic chemical kinetics. *The Journal of Chemical Physics*, *121*(21), 10356–10364. https://doi.org/10.1063/1.1810475

Weber, R. (2012). Markov Chains. *Statslab.Cam.Ac.Uk*, 28–49. https://doi.org/10.1017/CCOL0521534283.010

Weinan, E., Liu, D., & Vanden-Eijnden, E. (2007). Nested stochastic simulation algorithms for chemical kinetic systems with multiple time scales. *Journal of Computational Physics*, *221*(1), 158–180. https://doi.org/10.1016/j.jcp.2006.06.019

Woeginger, G. J. (2004). Space and Time Complexity of Exact Algorithms: Some Open Problems (Invited Talk). *Iwpec*, 281–290.

Wolf, V., Goel, R., Mateescu, M., & Henzinger, T. (2010). Solving the chemical master equation using sliding windows. *BMC Systems Biology*, *4*(1), 42. https://doi.org/10.1186/1752-0509-4-42