Simulation of Chemical Reactions Using Stochastic Petri Nets

Mahsa Dadar

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Master of Applied Science (Electrical & Computer Engineering) at

Concordia University

Montréal, Québec, Canada

August 2013

CONCORDIA UNIVERSITY
SCHOOL OF GRADUATE STUDIES

This is to certify that the thesis prepared

By: Mahsa Dadar

Entitled: "Simulation of Chemical Reactions Using Stochastic Petri Nets"

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science

Complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

______________________________ Dr. M. Z. Kabir	Chair
______________________________ Dr. F. Haghighat (BCEE)	Examiner, External To the Program
______________________________ Dr. A. Aghdam	Examiner
______________________________ Dr. S. Hashtrudi Zad	Supervisor

Approved by: ______________________________
Dr. W. E. Lynch, Chair
Department of Electrical and Computer Engineering

____________20_____

____
Dr. C. W. Trueman
Interim Dean, Faculty of Engineering and Computer Science

ABSTRACT

Simulation of Chemical Reactions Using Stochastic Petri Nets

Mahsa Dadar

The recent breakthroughs in biological experiments have enabled the researchers to measure the quantities of different chemicals that build biological units such as cells. This type of information can be used to build models that can explain and predict the behaviour of the system. Such models can later be used to design control mechanisms that can influence the behaviour of the system in a desired way. With the help of medicine and biology researchers, the designed control mechanisms can be translated into drugs that can control or cure major diseases.

Biological systems usually consist of complex networks of biological components that function through various reactions. In order to affect the behaviour of the system efficiently, the chemicals that have the highest influence on the system behaviour have to be found using a sensitivity analysis. Such chemicals, regarded as inputs, will be the targets for drug design (or other control actions).

Various modeling tools have been empolyed to capture the behaviour of biological systems. Perhaps the most widely used models are the ordinary differential equations (ODEs). In this thesis, an alternative model is proppposed for the study of the chemical reactions based on stochastic Petri nets, one type of discrete event systems. It is shown that the proposed method can be used to find the changes of the chemical reactants. The advantage of the proposed method is that it is amenable to implementation on computing systems with parallel processors. This in turn reduces the (time) computational complexity (compared with ODE based simulations). The Petri net based simulations are also used to perform sensitivity analysis. The proposed method is illustrated using the Caspase Apoptosis network.

All models are wrong, but some are useful.

George E. P. Box

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my supervisor, Dr. Shahin Hashtrudi-zad, who has helped and supported me throughout this thesis. This work would not have been possible without his encouragement and guidance. It has been an honour and a privilege to work under his supervision.

I want to thank my committee members Dr. A. Aghdam and Dr. F. Haghighat for their patience, dedication and invaluable advice as well as my chair Dr. Z. Kabir.

I am indebted to my parents for their faith in me and allowing me to be as ambitious as I wanted, and supporting me along the way. To them I dedicate this thesis. I would also like to thank Yashar for his technical and emotional help and support in the past years, I would not be where I am without him.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

BST	**B**iochemical **S**ystems **T**heory
CTMC	**C**ontinuous-**T**ime **M**arkov **C**hains
DES	**D**iscrete **E**vent **S**ystem
KEGG	**K**yoto **E**ncyclopedia of **G**enes and **G**enome
MCA	**M**etabolic **C**ontrol **A**nalysis
MSE	**M**ean **S**quared **E**rror
ODE	**O**rdinary **D**ifferential **E**quation
PN	**P**etri **n**et
RMSE	**R**oot **M**ean **S**quared **E**rror
SPN	**S**tochastic **P**etri **n**et
XIAP	**X**-linked **I**nhibitor of **A**poptosis **P**rotein

Chapter 1

Introduction

Protein reaction networks are one of the building blocks of many biological systems. Various types of interactions are linked and orchestrated to achieve different objectives. From a control engineering point of view, protein networks can be considered as dynamical systems whose behaviour can be studied and analyzed using various analytical methods. Using control systems theory, proper controllers can be designed to influence the behaviour of a system in a specific way. To be able to do so, one first needs to model the system using a modeling formalism suitable for the specific system and the desired application. The next step would be to analyze the behaviour of the system using the tools provided by the selected formalism. Using the information obtained from the analysis methods, one may design control mechanisms to change the behaviour of the system to a desired one. The designed control mechanism can later be constructed and connected to the real system.

A *chemical reaction network* is a network modeling the reactions between different chemicals in a system composed of a set of *reactants*, a set of *products* (that can also be reactants in other reactions), and a set of *reactions*. Chemical reaction networks are usually complicated networks typically including a very large number of interactions which make modeling and analysis tasks more difficult due to their computational complexity. Furthermore, the knowledge of interactions between different entities may not be completely available. Therefore, some modeling methods may fail to model relatively large networks with desirable accuracy and speed. In such cases, choosing the proper modeling formalism and analysis methods is of importance.

Using *Petri nets* for modeling chemical reaction networks provides the analyst with certain advantages. Petri net models consider events in a discrete manner, i.e. each event will modify the state of the system. In addition, the events that may occur simultaneously are modeled randomly. This allows the biologists to infer the workings of the system in a more transparent and biologically meaningful way, rather than to be forced to interpret the actions in terms of *Ordinary Differential Equations (ODEs).* The concentrations of various molecular species can be altered in the model to provide different what-if scenarios. The transitions determine the strength or concentration levels required before activating next state; this allows the internal workings of the model to be observed. Moreover, using discrete event system tools to model a system can improve the computation time immensely. As a result, one might be able to perform simulations that were not possible due to the computational expense of ODE formalism.

1.1 Problem Statement and Motivation

Because of the large number of interactions in chemical reaction networks, a systematic representation of the model is highly desirable. Many of the mathematical models of the intra-cellular biological systems are often based on systems of nonlinear ordinary differential equations [2–7]. While ODE models have proven to be very useful, the computational complexity of the nonlinear ODE model grows rapidly as the number of proteins and interactions increase, which makes it extremely hard, or in some cases even impossible, to analyze an entire chemical reaction network using ODE based analysis methods. The ODEs are usually too complex to have closed solutions and are numerically solved by iterative computations in small steps. Furthermore, the information necessary to build such models are not always available or are expensive in terms of time and cost to obtain.

One of the alternatives that can be used to allow more flexibility in modelling is to use discrete event models or a combination of ODEs and discrete event systems (i.e. hybrid models) to incorporate the advantages of the two formalisms. [8–15].

Numerous diseases are caused by perturbations that drive the biological system out of its usual trajectory or steady state. Sufficiently accurate yet computationally manageable models for biological systems could greatly facilitate the design of drugs and therapy mechanisms for curing and preventing such diseases.

1.2 Research Objectives

In this thesis, a discrete event system modeling (DES) framework is developed to represent and analyze biological systems for which ordinary differential equations are available (e.g. protein interaction networks). This DES model should take into account all the important functions and behaviours of the biological system. This model will then be used to simulate the behavoiur of the system in different situations. Test cases are used to evaluate the speed of the resulting simulation algorithm.

One of the challenges of designing drugs (which can be regarded as one form of control mechanism) is choosing the appropriate target chemical. As was mentioned before, biological networks are usually very large. Therefore, it would be difficult and expensive to design control mechanisms without first knowing which input chemicals have the highest influence on the output chemicals. The simulation algorithms developed in this thesis can be used to carry out sensitivity analysis to determine the most potent input chemicals of interaction networks.

1.3 Literature Review

This section contains a brief overview of the concepts that form the foundation of this thesis. The central goal of this thesis is to propose a discrete event method for modeling and then performing sensitivity analysis on biological systems. First, sytems biology and ordinary differential equation models are reviewed as a commonly used method for modeling biological systems. Then, sensitivity analysis and Petri nets (the selected discrete event model formalism) are reviewed with more emphasis on the specific type of Petri nets that have been used in this thesis.

1.3.1 Systems Biology

Systems biology is a growing field which mainly studies the interactions within biological systems. Systems Biology originated from the molecular biology and genomic biology revolutions. The prospect of being able to analyze and understand how biological systems function by integrated operation of their component parts and considering their interactions (rather than simply focusing on individually isolated components) is of essential importance in medical applications, as well as many other applications such as the environment, materials and manufacturing [16]. Such study of biological systems has an important impact on the better understanding of living systems which in turn, leads to advances in diagnosis, treatment, prevention and relieving of diseases. The ability to collect comprehensive data on system performance and the underlying molecules enables us to obtain a system level understanding of biological phenomena which was not possible before. For a brief review, see [17, 18].

Biochemical systems analysis focuses on the integrated functioning of the intact system rather than the chemical and physical properties of the individually isolated components that build the system. It approaches biochemical systems as a combination of interacting components that can be studied as a whole using different mathematical methods. Everything outside of this whole will be considered as the environment of the system [19].

Cellular processes can be repressented as systems of chemical reactions which can be used to create models of the system. These models were traditionally deterministic ordinary differential equations. But in the case of modelling a cell, usually assumptions such as high levels of key chemicals of the reaction do not hold. As a result, the system behaves more stochastically rather than deterministically [20]. So, using a stochastic method to model the reactions may be more appropriate than a deterministic one. For a review on stochastic methods of modeling biochemical reactions, see [20].

Understanding how the model behaves in response to changes in its inputs is of fundamental importance. Sometimes, the exact values of some variables or parameters are unknown due to measuring difficulties or insufficient precision [21]. In such cases, one needs to know whether the output of the system is sensitive to an imprecise parameter and if variations in a parameter value has negligible effect on the output. On the other hand, one might want to find the parameters that have the highest influence on the output (to be able to manipulate them to influence the output in a desirable manner). One of the important tools used in analysis, design, and optimization of a system is sensitivity analysis. Sensitivity analysis is concerned with how changes in the output of a model (system) can be affected by changes in the inputs of the model.

Biological systems are usually robust against changes in environmental conditions. However, in every system including biological systems, there might be certain parameters that are subject to tight control for which even small perturbations may result in the malfunction or even the death of the system. Examples for such parameters include the operating temperature of the human brain and blood glucose levels [22]. By using a systems perspective, after structural and dynamical analysis of the system, one might be able to make useful predictions of unknown interactions and behaviours and design control mechanisms to foresee and prevent such malfunctions.

1.3.2 Modeling Biological Systems Using Ordinary Differential Equations

ODEs have been commonly used to model and describe the behaviour of biological systems such as regulatory networks. Some of the efforts that have been made in this area are mainly focused on the qualitative aspects of the model (e.g. [2]). Other methods have used ODEs to perform quantitative simulations (e.g. [4]). For a review, see [23].

In some more recent works, researchers have tried to integrate ODEs with other modeling methods and combine the advantages of both models. For example, in [3], Petri nets and ODEs have been combined to create a modeling formalism that can perform quantitative analysis taking into account timing information in order to be able to model real system behaviours. In [6], ODEs have been combined with discrete events that model the switching between the domains of attraction of different steady states in a model to present a methodology for the dynamic analysis and control of mode transitions in biological networks.

1.3.3 Parallel Computaions in Ordinary Differential Equations

In recent years, the availability of parallel processing systems has attracted interested reasearchers to computation approaches that are able to exploit such systems to speed up calculations. In the case of ODEs, researchers have tried to find suitable splittings of the ODE problem whose solution would be equal to the iterative solution of the ODEs [24]. The need for increasing computational speed in ODE based calculations arises in cases where the ODE function is expensive to evaluate, the number of equations is relatively large, the interval of integration is long, or the ODEs need to be solved repetitively [25].

Parallel computing methods that are used in ODE problems can be divided to two classes, parallelism across time or solution space (or a combination of both) [25]. One of the main concerns in such methods is the need for the synchronization in cases where one processor receives information from another [25]. For a review, see [25].

1.3.4 Sensitivity Analysis

Sensitivity analysis is an important tool in studying the dependence of systems on different inputs or parameters. Studying the sensitivity relationships in stoichiometric networks has been addressed by *Biochemical Systems Theory (BST)* [19], and *Metabolic Control Analysis (MCA)* [26–28]. MCA is a theory concerned with the analysis of the distribution of control within a network and dependence of the behaviour of the system on the properties of the components.

The MCA literature usually considers networks in steady state. However, in some applications, it is necessary to investigate sensitivities along non-steady trajectories. In many systems such as signal transduction and cell cycle regulation networks, the transient or oscillatory behaviour of the network is important and cannot be discarded. Some studies have focused on extensions of MCA for non-steady behaviour (mostly special cases of dynamical behaviour such as periodic behaviour or trajectories near a stable steady state.) [29–35]. In [36], these results have been extended by measuring time varying sensitivities along arbitrary trajectories. Using such analysis methods, "one can determine the sensitivity to the perturbation throughout the time evolution of the system, regardless of the nature of the trajectory" [36]. This method is particularly useful when studying systems in which transient behaviour plays a key role in the operation of the system. For a review, see [37, 38]

Researchers have also tried to transform the problem of calculating the sensitivities to equivalent problems that can be solved in parallel to reduce computation speed. In [39], parallel solutions have been proposed for the ODE based sensitivity analysis methods with a moderate number of state variables.

The main concerns in choosing a method for computing sensitivities are its accuracy, computational cost, and in certain cases where the system is very large or complex, ease of implementation [40].

1.3.5 Petri Nets

Petri nets (PNs) historically originate from the dissertation of Carl Adam Petri, a German mathematician and computer scientist, for the purpose of representing the actual physical flow of information in the solution of a recursive problem [41]. Petri nets are an excellent modeling formalism for describing and studying systems that are characterized as being concurrent, asynchronous, distributed, parallel or non-deterministic. Due to their firm mathematical foundation, Petri nets are applied in practice in many fields, such as asynchronous circuit design, communication protocols, distributed computing, production systems, flexible manufacturing, transportation, and systems biology. Particularly in systems biology, different classes of Petri nets have been used to model a large variety of biological systems. In [42], Petri nets have been compared with other similar methods to evaluate their suitability for representing and simulating biological processes.

The Petri net formalism combines an intuitive graphical notation with advanced analysis techniques. Classical Petri net analysis techniques are structural analysis techniques that study the properties of the system by investigating the structure of the corresponding Petri net. Such studies do not investigate the changes in the state of system (*marking*) during the evolution of the Petri net. In some cases, the initial marking (state) is also considered. Several studies have used Petri nets for qualitative analysis of biochemical networks to avoid the computational expense and lack of data [43–45]. Petri net models have been widely used for qualitative analysis of logical properties of biological systems such as the presence of system deadlocks or live-locks, invariants, relationships between events (causality, mutual exclusion, concurrency), and boundedness of the system state space [46–49]. Such qualitative methods are independent of the parametric information of the system (such as rate constants and cooperativity indices in biological systems).

On the other hand, in other analysis techniques, the reachability graph of the Petri net or its simulation is used for the verification of the qualities of the system properties. This type of analysis of Petri nets is based on investigating the possible markings (states). Some studies have tried to reduce the size of the model to be able to perform simulations [50, 51].

The PN semantic does not state which of multiple simultaneously enabled transitions fires first, so a PN analysis must either examine every possible firing order or find a way to find the trajectories that are specific to its application. The firing of transitions in each step of the Petri nets is random. So, one cannot determine what trajectory the system will follow starting from a specific initial point. The Petri net models that have been used for analyzing biological systems so far addressed this problem by calculating all the possible trajectories that may occur from the initial point (the reachability tree of the Petri net) and studying these trajectories as a group [35]. Other studies using Petri nets have only focused on the structural properties of the Petri nets that can be analyzed without computing the trajectories (such as studying invariants, deadlocks, livelocks, reachability, controllability, etc.) to avoid the computation expense caused by calculating all the trajectories [46–49]. Furthermore, most of the trajectories that are calculated in the reachability tree never occur in the real system and its ODE model equivalent. So, it is of great value to be able to find a way to find the trajectories that really occur using a Petri net model. In [52], a comprehensive review on the recent research on Petri nets is presented.

1.3.6 Stochastic Petri nets

Many extensions of PNs have been proposed to increase either the class of problems that can be represented or their capability to deal with the common behaviours of real systems. *Stochastic Petri nets* are a form of Petri nets that have been extensively used due to their ability to capture timing and uncertainty in the models [53–55]. In [56], "Modeling Power" is defined as the ability of the PN formalism to capture the details of a system by allowing different types of firing distributions. "Logical Constructs" are defined as Petri nets that change the firing rule of the transitions to achieve a goal, and "Stochastic Constructs" as Petri nets that associate a stochastic interpretation to the evolution of the Petri net [56]. The second category can be divided into two groups:

☐ Probability Distribution for the Firing Times of the Transitions: There are many different choices for the distribution types and other features.

☐ Relative Probability of Firing for Conflicting Transitions: When several transitions are enabled simultaneously, the transition to be fired can be selected according to some firing rule. The Petri nets that have been used in this thesis are of this type.

Petri nets do not include time concepts in their original definitions. By using stochastic Petri nets (SPNs), the modeling power can be increased by associating random firing times with the transitions. A transition's firing time represents the amount of time required by the activity associated with the transition [57]. Temporal specification in PN models were introduced later with different approaches, first based on the use of deterministic timing and later based on stochastic timing. Particularly in SPNs, transition firing delays are exponentially distributed random variables [58], i.e., a random firing delay is associated to each transition T_i whose probability density function is a negative exponential with a rate λ_i.

The semantics of SPNs is described by a race model: when several transitions are enabled in a given state, all activities associated with these transitions are assumed to be executed in parallel and the change of state is caused by the firing of the transition with the minimal firing delay, i.e., the transition with minimal delay wins the race. The firing of the winning transition models the completion of the represented activity and the behaviour of the losing transitions can be specified by different policies [59].

When the set of enabled transitions $T_{enabled}(M)$ in marking (state) M contains more than one element, the probability that transition T_i actually fires can be obtained as:

$$P(T_i|M) = \frac{\lambda_i}{\sum\limits_{T_j \in T_{enabled}(M)} \lambda_j} \tag{1.1}$$

The PNs that have been used in this thesis are also stochastic PNs, but the definition of the probability function is different from the one mentioned in (1.1). The subject will be discussed thoroughly in Chapter 3.

1.4 Contributions of the Thesis

In this work, discrete event models (specifically, Petri nets) have been developed to represent biological systems (more specifically, protein networks) using the available biological data. The goal is to define a discrete model of the biological system using the currently available ODE model or its parameters. This model will later be used to calculate a measure of sensitivity. Within the proposed framework the following key contributions have been made:

- Defining of a new type of stochastic Petri net that can produce the same results as the ODE model: We have proposed a method in which single trajectories of the ODE model can be calculated from the expected values of the trajectory of a suitably defined stochastic Petri net.

- Evaluating of the proposed method by calculating the mean and variance of the values and showing that the expected value of the results of the Petri net model is equal to the results of the ODEs.

- Studying the effect of quantization (of converting a continuous ODE to a discrete Petri net) on the proposed model.

- Applying parallel computation in order to achive a higher computational speed (compared with ODE simulations).

- Defining an efficient sensitivity measure suitable for the proposed discrete model and developing an algorithm to calculate it.

- Developing MATLAB code for performing the simulations and visualizing the desired outputs.

1.5 Organization of the Thesis

The rest of this thesis is organized as follows. Chapter 2 covers the necessary background on Petri nets, sensitivity analysis, chemical reaction networks, and the basic biological knowledge needed to follow the concepts in the thesis. In Chapter 3, the main ideas of the thesis are developed and the results are derived. In chapter 4, the apoptotis protein network which has been used in the thesis as case study is examined and the simulation results verifying the results obtained in Chapter 3 are presented. The thesis concludes in Chapter 5 by summarizing our work and discussing some directions that future researches may take.

Chapter 2

Background

This chapter covers the background material necessary for the development of the thesis. First, a brief overview of protein reaction networks and the pathway that has been studied throughout the thesis is provided. Then, Petri nets are introduced with more emphasis on the specific concepts that are used in this thesis. Chemical reactions and their ordinary differential equation models are reviewed. Finally, sensitivity analysis is breifly reviewed and the sensitivity analysis method that has been used in this study is explained.

2.1 Protein Reaction Networks

One of the popular areas of research in systems biology is the study of protein reaction networks and their properties. Protein reaction networks usually involve metabolic networks or cell signaling networks. A *protein pathway* (also called metabolic pathway) is a pathway consisted of a set of proteins or chemicals and a series of chemical reactions that occur within a cell and modify the chemicals in the pathway. Various chemicals can be involved in a pathway. As a result, protein pathways are usually very complex. A large number of pathways can coexist within a cell. These sets of pathways are called *protein networks*.

In a chemical reaction, the chemicals that begin the reaction and are converted to a new set of chemicals (called *products*) through the reaction are called *reactants*. The chemical reactions in a pathway can be (and usually are) reversible, i.e. in the course of the reaction, the products are also converted to their producing reactants. Chemical reactions are usually heavily dependent on the availability of different groups of reactant proteins.

Proteins are large biological molecules that form most of the mass of an organism's cell and dictate its entire chemistry, as well as its form and behaviour. Proteins carry out many vital functions in the cell, such as maintaining cell structure, catalyzing reactions inside the cell, replicating DNA, responding to stimuli, and selectively transporting molecules from one location to another. Various groups of larger proteins are assembled in the cell to carry out discrete sets of functions [60, 61].

Studying protein pathways and their functions and properties can help us gain invaluable information about the cells of living organisms. Being able to model and predict the behaviour of a specific pathway can enable us to understand the nature of the cell better and design control mechanisms that can change the behaviour of the pathway.

The main objective of this thesis is to develop an analysis methodology for the behaviour of biological systems based on discrete-event models, more specifically, Petri nets. While the discussion of biological details have been avoided as much as possible, a basic understanding of some aspects is necessary.

The pathway that is studied in this thesis is called the *Cell Apoptosis Pathway* [62]. Apoptosis is the internally executed process of cell death that occurs in multi-cellular organisms. Various proteins interact together in this pathway to create a signal that brings about cell suicide. The Apoptosis pathway will be reviewed in more details in Chapter 4.

2.2 Petri Nets

Petri nets are an excellent modelling formalism for describing and studying the characteristics of biologicals systems due to their intuitive graphical notation and firm mathematical foundations. Petri nets are applied in many different fields such as software design, workflow management, process modeling, discrete process control, transportation, and systems biology. Particularly in systems biology, several different classes of Petri nets have been used to model a large variety of biological systems. Petri nets provide rich mathematically founded analysis techniques that can be used to study the structural and behavioural properties of the system.

A Petri net is a bipartite directed graph that contains two types of nodes: *places* and *transitions*. Places are signified by circles and model passive system elements such as resources, conditions, and signals. Transitions are signified by bars and usually represent active system elements such as events, actions, or chemical reactions [63]. In the context of biological systems, places typically represent chemicals such as proteins and transitions represent chemical reactions. More details will be provided in Chapter 4.

No two places or transitions are connected directly. Places can be connected to transitions via directed (weighted) *arcs* that describe which places are pre/post conditions for which transitions. Such connections model the causal relationship between the active and passive elements of the system. A place is connected to a transition (pre place) if the firing of that transition is conditional on the state of that place. Similarly, a transition is connected to a place if its firing changes the state of that place (post place).

During the execution of a Petri net, places hold a zero or positive discrete number of *tokens*. Tokens are dynamic system elements that determine the state (marking) of the Petri net by their distribution over different places throughout the Petri net. A transition may fire (an action which leads to a change in the marking of the net) whenever there are sufficient (equal to or more than the weight of the connecting arcs) tokens in all of its preplaces. When a transition fires, it consumes all these tokens and places them in the postplaces at the end of the arcs. The number of tokens removed from (or added to) each place is equal to the weight of the arc connecting that place to the fired transition. When such firings occur, the state (marking) of the system changes.

The execution of Petri nets is nondeterministic, in the sense that when multiple transitions are enabled at the same time, any one of them may fire, hence, different states may be reached from a single state. If a transition is enabled, it may fire, but it does not have to. Since firing is nondeterministic and multiple tokens may be present anywhere in the net, Petri nets are well suited for modeling the concurrent behavior of distributed systems.

Consider five chemicals $x_1, ..., x_5$ engaged in the following three reactions.

$$\begin{aligned} &T_1 : x_1 + x_2 \longrightarrow x_3 + x_4 \\ &T_2 : x_2 \longrightarrow x_4 \\ &T_3 : x_3 + x_4 \longrightarrow x_5 \end{aligned} \tag{2.1}$$

Figure 2.1 shows a simple Petri net with five places and three transitions that can be used to represent the reactions. The initial number of tokens are chosen in proportion to the initial concentration of the chemicals. Here it is assumed $x_0 = [3, 5, 2, 1, 0]^T$. The weights of the arcs are equal to 1.

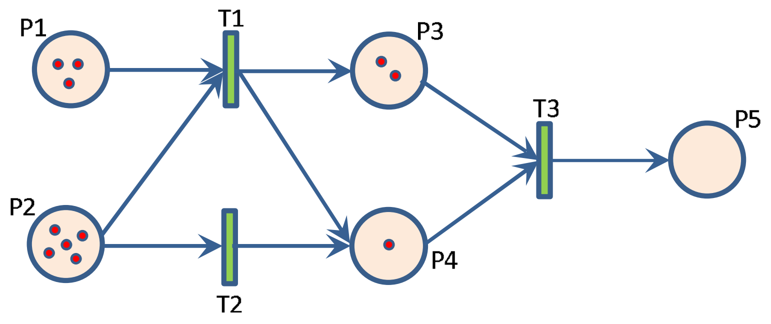

FIGURE 2.1: A Simple Petri net with 5 Places and 3 Transitions.

Table 2.1 shows a possible sequence of transition firings that can occur and the change of the state of system. In step 1, all transitions are enabled since there

TABLE 2.1: A Possible Firing Sequence for Petri net 2.1

Step	Previous State	Firing Transition	Next State
1	$x_0 = [3,5,2,1,0]^T$	T_1	$x_1 = [2,4,3,2,0]^T$
2	$x_1 = [2,4,3,2,0]^T$	T_3	$x_2 = [2,4,2,1,1]^T$
3	$x_2 = [2,4,2,1,1]^T$	T_2	$x_3 = [2,3,2,2,1]^T$
4	$x_3 = [2,3,2,2,1]^T$	T_1	$x_4 = [1,2,3,3,1]^T$
5	$x_4 = [1,2,3,3,1]^T$	T_3	$x_5 = [1,2,2,2,2]^T$
6	$x_5 = [1,2,2,2,2]^T$	T_1	$x_6 = [0,1,3,3,2]^T$
7	$x_6 = [0,1,3,3,2]^T$	T_2	$x_7 = [0,0,3,4,2]^T$
8	$x_7 = [0,0,3,4,2]^T$	$3 * T_3$	$x_8 = [0,0,0,1,5]^T$

are enough tokens in each of their pre places. After transition T_1 is fired, one token is removed from each of places P_1 and P_2 and added to place P_3. In each step, an enabled transition is chosen to fire randomly and the new marking is calculated. Transition T_1 is disabled from step 6, since place P_1 no longer has any tokens. Similarly, transitions T_2 and T_3 become disabled in steps 7 and 8, respectively. When the Petri net reaches states $x_8 = [0,0,0,1,5]^T$, all transitions are disabled and the state of the Petri net can no longer change.

A Petri net is formally a four-tuple $G = (P, T, F, x_0)$ where P is a finite set of places and T is a finite set of transitions [63]. Let Z^+ denote the set of non-negative integers. F is a set of flow relations between places and transitions and vice versa defined as

$$F : (P \times T) \cup (T \times P) \longrightarrow Z^+ \tag{2.2}$$

For $F(a, b) > 0$ there is an arc from a to b with multiplicity (weight) $F(a, b)$.

x is the marking function that gives the number of tokens in each place defined as $x : P \longrightarrow Z^+$, x_0 is the initial marking of the Petri net.

A transition t is enabled at marking x if

$$\forall p \in P : x(p) \geq F(p,t) \tag{2.3}$$

A transition t from a marking x to a marking x' is denoted by $x \xrightarrow{t} x'$, where

$$x'(p) = x(p) - F(p,t) + F(t,p) \qquad \forall p \in P \tag{2.4}$$

Let $A = [a_{ij}]$ is a $\rho \times N$ matrix of integers called the incidence matrix where ρ is the number of transitions and N is the number of places. The a_{ij}'s entry is given by $a_{ij} = a_{ij}^+ - a_{ij}^-$ where a_{ij}^+ is the weight of the arc from transition i to its output place j and a_{ij}^- is the weight of the arc to transition i from its input place j.

The reachability set of a Petri net R_{x_0} is the set of states (markings) that can be reached from a specific initial condition x_0.

Petri nets can be characterized by some important structural and behavioural properties. Examining these properties is usually a key step in the analysis of Petri nets. Some of these important characteristics are [63]:

☐ Reachability: Reachability is fundamental for studying the dynamic behaviour of Petri nets. It is concerned with whether a certain state can be reached from an initial state.

☐ Boundedness: Starting from an initial marking, if the number of tokens in all the places of a net remain bounded, the net is bounded. If the net is bounded for any initial marking, it is called structurally bounded.

☐ Liveness: Liveness describes the possibility for a transition to be enabled and fire infinitely often. If each transition of a net is live, the net is live.

☐ Reversibility: Reversibility describes the possibility of a net being able to go back to a previous state. If a Petri net can reach its initial marking again from any reachable marking, the net is reversible.

☐ Invariants: A T-invariant is a nonzero vector $u \in (Z^{+})^{\rho}$ which satisfies the equation $u.A = 0$. A T-invatiant represents a set of transitions which altogether (if executable) have a zero effect on the marking. T-invariants show the possible cyclic system behaviour. A T-invariant is called realisable, if a marking is reachable, such that all transitions of the T-invariant are able to fire in some order. Analogously, a P-invariant is defined as a non-zero vector $x \in (Z^{+})^{N}$ which satisfies the equation $A.x = 0$. A P-invariant characterises a token conservation rule for a set of places, over which the weighted sum of tokens is constant, independent from any firing of transitions.

2.3 Chemical Reactions

A chemical reaction is a process in which a set of chemical substances is transformated to another. A simple bimolecular reaction is represented as:

$$aA + bB \xrightarrow{k_1} cC + dD \tag{2.5}$$

where A and B are the reactants and C and D are the products and k_1 is the reaction rate, and the lower case letters represent the coefficients of the balanced equation. Normally an elementary reaction involves no more than two (or at most 3) reactant molecules.

2.3.1 Multiple Reversible Reactions

The following formula can be used to represent ρ elementary reactions that have a forward and reverse step, i.e. the reactions are reversible [64].

$$\sum_{j=1}^{N} v_{ji}^{f} c_j \rightleftharpoons \sum_{j=1}^{N} v_{ji}^{r} c_j \qquad , \qquad i = 1, 2, ..., \rho \tag{2.6}$$

where v_{ji}^{f} (v_{ji}^{r}) represents the absolute value of the stoichiometric coefficient of species c_j in the i^{th} forward (reverse) (or f (r)) reaction.

2.3.2 Law of Mass Action

The rate of an elementary chemical reaction is proportional to the product of the concentrations of the reactants. Let k_i^f (k_i^r) be the rate constant for the i^{th} forward (reverse) reaction. Hence, the rate of progress of the i^{th} reaction (the change in concentration over the change in time) will be [64]

$$q_i = k_i^f \prod_{j=1}^{N} c_j^{v_{ij}^f} - k_i^r \prod_{j=1}^{N} c_j^{v_{ij}^r} \tag{2.7}$$

where c_j is also used to represent the concentration of species c_j. The net rate $R_j = \frac{dcj}{dt}$ of production of species j is

$$R_j = \sum_{i=1}^{\rho} v_{ji} q_i \tag{2.8}$$

Here v_{ji} is the net stoichiometric coeffiecient for species j in reaction i.

$$v_{ji} = v_{ji}^r - v_{ji}^f \tag{2.9}$$

A very common method of analyzing chemical reactions is using Ordinary Differential Equations (ODEs) to calculate the concentrations of each molecule over time. So, in order to solve a system of chemical reactions, the above relations will be translated to the following set of ODEs [64]

$$\frac{dc_j(t)}{dt} = \sum_{i=1}^{\rho} v_{ji} q_i = \sum_{i=1}^{\rho} v_{ji} (k_i^f \prod_{j=1}^{N} c_j(t)^{v_{ij}^f} - k_i^r \prod_{j=1}^{N} c_j(t)^{v_{ij}^r}) \tag{2.10}$$

2.3.3 Equilibrium (Steady State)

If reaction i is at equilibrium, then its rate of progress is $q_i = 0$. This means that the forward and reverse reaction rates are equal [64].

$$k_i^f \prod_{j=1}^{N} c_j^{v_{ij}^f} = k_i^r \prod_{j=1}^{N} c_j^{v_{ij}^r} \Longrightarrow \frac{k_i^f}{k_i^r} = \frac{\prod\limits_{j=1}^{N} c_j^{v_{ij}^f}}{\prod\limits_{j=1}^{N} c_j^{v_{ij}^r}} = \prod_{j=1}^{N} c_j^{v_{ij}} = K_i^{EQ} \qquad (2.11)$$

Here K_i^{EQ} is the equilibrium constant which is a function of temperature.

2.4 Sensitivity Analysis

Sensitivity analysis is the study of how changes in the output of a system or mathematical model are apportioned to the changes in its inputs. When the system under study has several input variables (which is usually the case in chemical reaction networks), sensitivity analysis is essential for model analysis.

Sensitivity analysis can be useful for a range of purposes, including the following [65].

□ Understanding the relationships between the different input and output variables in the system and identifying the sensitive or important variables.

□ Searching for errors in the model: Detecting the unexpected relationships between inputs and outputs.

☐ Simplifying the model: removing parts of the model (connections with inputs) that do not have any significant effect on the output.

☐ Reducing uncertainty in the model: in some models, some inputs cause significant uncertainty in the output. Such inputs should be studied and focused on if robustness is to be increased.

There are several commonly used methods of sensitivity analysis. The choice of method is usually affected by a number of problem constraints or settings. Some of them include the following:

☐ Computational expense: Sensitivity is usually measured by sampling-based methods which rely on generating and exploring a mapping from the inputs to the outputs (which requires running the model several times). Such approaches can pose problems when a single run of the model takes a relatively significant amount of time [38].

☐ The curse of dimensionality: Sensitivity analysis requires exploring the multidimensional input space. The volume of the input space can increase exponentially by increasing the number of inputs. This can pose serious problems when the model has a large number of inputs [66].

☐ Nonlinearity: Some sensitivity analysis methods are only accurate when the model is linear with respect to its inputs [66].

☐ Model interactions: Sometimes the simultaneous perturbation of two or more inputs causes a greater change in the output compared with that of perturbing each of the inputs alone. Such interactions cannot be recognized by one-at-a-time (OAT) perturbation methods when the model is nonlinear [66].

Therefore designing a sensitivity analysis method that is suitable for a specific application can be of great importance. Various sensitivity analysis methods have been applied to biological systems. They can provide useful information about how the biological system changes in response to different inputs and about the inputs that are the key factors in the system. One of the methods that has been widely applied to chemical reaction network systems evaluates sensitivity based on the trajectory of the output in the solution region. This method has also been implemented to perform sensitivity analysis in SimBiology toolbox of MATLAB [1].

2.4.1 Sensitivity Analysis Based on Trajectories of the Solution

This section reviews the sensitivity analysis method proposed in [36] which is applied for sensitivity analysis in MATLAB's SimBiology.

Given the $(N \times 1)$ vector of initial conditions (the initial concentrations of the proteins) $c(0) = c_0$ and the $(r \times 1)$ set of parameter values p_0 (other parameters of the protein pathways such as reaction rates), which together form a vector q_0, the time-varying concentration sensitivity coefficients are defined as the elements of the $N \times (N + r)$ matrix function $S_q^c(.)$ given by [36]:

$$S_q^c(t) = \frac{\partial c(t,q)}{\partial q}|_{q=q_0} = lim_{\Delta q \to 0} \frac{c(t, q_0 + \Delta q) - c(t, q_0)}{\Delta q} \qquad \forall t \geq 0 \tag{2.12}$$

These time varying sensitivity coefficients can be interpreted similar to the standard steady state sensitivity coefficients of MCA. Using this method, the response to a perturbation along an entire trajectory is considered rather than at a particular steady state. As time tends to infinity, each trajectory will converge to its steady state, and the sensitivity coefficient $S_q^c(t)$ will converge to the steady state response of MCA [36].

Note that the response coefficient $S_q^c(.)$ is a function of the particular parameters p_0 and a particular initial condition c_0.

In order to obtain a single sensitivity coefficient for each parameter, a time integral of $S_q^c(.)$ can be measured throught the simulation time [36].

$$s_q^c = \int_0^T S_q^c(t)dt \tag{2.13}$$

The sensitivity analysis method used in this thesis is based on and similar to this approach. More details on how the sensitivities are calculated using the Petri net model simulations will be provided in Chapter 3.

2.5 Conclusion

In this chapter, a basic background on systems biology, Petri net models and sensitivity analysis was provided. The current method that is used for performing sensitivity analysis in protein networks is based on the solution of the ODEs of the system which is a complicated and computationally expensive task due to the usually large size of the equations in the equavalent ODE model of the protein networks. In the rest of this thesis, we will show that discrete event system models (more specifically, Petri nets) can be used to calculate the same solution trajectories as ODEs (apart from the notion of time). Using these solutions, we are able to perform sensitivity analysis for the Petri net models and obtain similar results to those of ODEs. Due to the discrete and stochastic nature of the solutions, we are able to use parallel processing to increase the computational speed.

Chapter 3

Solving Ordinary Differential Equations Using Petri Nets

In this chapter, we will explain how a set of ordinary differential equations can be solved using Petri nets. We will then use the proposed method to study the behaviour of a general protein reaction network using a Petri net. A simple example is used to clarify the method. A more detailed case study is provided in the next chapter.

3.1 Problem Statement and Motivation

Consider the following set of $\rho \times N$ ODEs with the following polynomial form:

$$\frac{dc_j(t)}{dt} = \sum_{i=1}^{\rho} a_{ji} \prod_{k=1}^{N} c_k(t)^{b_{kji}} \qquad j = 1, 2, ..., N \tag{3.1}$$

in which a_{ji} and b_{kji} are constants. The parameters b_{kji} are integers. Without loss of generality, it is assumed that each equation in (3.1) has ρ terms and that a_{ji}s are integers. This specific form is used for the ODEs because it is similar to the form of the ODEs that model protein reaction networks (with N chemicals and ρ reactions). In a chemical reaction network, a_{ji}s and b_{kji}s represent the rates of the reactions and the coefficients of the balanced equations. Later it will be shown that the results obtained here apply to larger sets of ODEs.

The solution of equation (3.1) can be approximated with Euler method as the following (for small Δt_ns):

$$\Delta c_j(t_n) = c_j(t_{n+1}) - c_j(t_n) = \sum_{i=1}^{\rho} a_{ji} \prod_{k=1}^{N} c_k(t_n)^{b_{kji}} \Delta t_n \qquad j = 1, 2, ..., N \tag{3.2}$$

3.2 Constructing a Petri net Model from a Set of Ordinary Differential Equations

A system described by equation (3.1) can be modeled by a Petri net model G with (at most) $\rho \times N$ transitions and N places. Each term c_j in the equations is modelled with a place x_j in the Petri net. The Petri net equivalent of a set of ODEs has a transition for each $\prod_{k=1}^{N} c_k^{b_{kji}}$ term in the ODEs. If there are two (or more) equal terms in the ODEs, a single transition can be used to model them. Each transition is connected to a place x_j with an arc with weight a_{ji} if it is equivalent to the i^{th} term in the j^{th} equation. The direction of the arc is determined by the sign of a_{ji}. If a_{ji} is positive, the place will be a post place for the arc. If a_{ji} is negative, the place will be a pre place for the arc in the Petri net. The initial state of the Petri net (i.e. the number of tokens in each place of the Petri net) is chosen proportional to the initial value of the chemical concentrations, i.e. $x_j = \lfloor \gamma c_j \rfloor$ ($\lfloor\rfloor$ is the floor function). We refer to γ as the *scaling factor*.

3.2.1 Simple Example

In order to illustrate the concepts, the theories in each section are described using a simple running example. Consider the following set of ODEs with polynomial form:

$$\begin{aligned} \frac{dc_1(t)}{dt} &= -c_1(t)c_2(t) \\ \frac{dc_2(t)}{dt} &= -c_1(t)c_2(t) + c_3(t) \\ \frac{dc_3(t)}{dt} &= c_1(t)c_2(t) - c_3(t) \end{aligned} \tag{3.3}$$

Following the procedure outlined in the previous section, we construct the following Petri net.

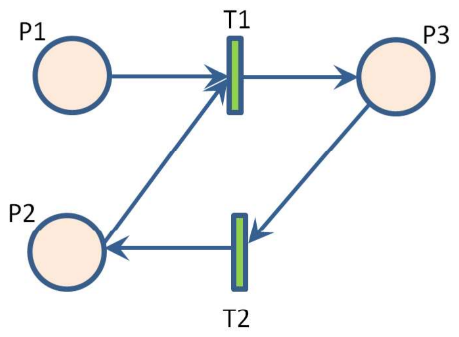

FIGURE 3.1: A Simple Petri net with 3 Places and 2 Transitions.

As we can see in the equations, there are two distinct terms $c_1(t)c_2(t)$ and $c_3(t)$ in the ODEs and as a result, two transitions in the Petri net model. Transition

T_1 models term $c_1(t)c_2(t)$ and transition T_2 models term $c_3(t)$. Since a_{11} is a negative number (-1), place $c_1(t)$ is an input place for transition T_1. The other arcs can be explained similarly.

As we can see in the equations, $\frac{dc_2(t)}{dt} = -\frac{dc_3(t)}{dt}$ which means that the absolute value of the changes in the values of $c_2(t)$ and $c_3(t)$ are equal. In the equivalent Petri net model, when T_1 fires, a token is removed from x_2 and added to x_3. Similarly, when T_2 fires, a token is removed from x_3 and added to x_2. So, the token change of x_2 and x_3 will also be equal.

In the Petri net model, the token count of a place changes if and only if a transition that has an input or output arc connected to that place fires. So, for each step during the evolution of the Petri net,we can write:

$$\begin{aligned} x_j(n+1) = x_j(n) + \sum_{i=1}^{\rho} w_{ji} f(T_{ji}, n) \qquad , \qquad j = 1, 2, ..., N \\ \Rightarrow \Delta x_j(n) = x_j(n+1) - x_j(n) = \sum_{i=1}^{\rho} w_{ji} f(T_{ji}, n) \end{aligned} \tag{3.4}$$

in which $x_j(n)$ is the token count of the j^{th} place at the n^{th} step during the evolution of the Petri net, and w_{ji} is the weight of the arc that connects the j^{th} place to the i^{th} transition. T_{ji} is the transition modeling the i^{th} term in the j^{th} equation in the ODEs. If a place is an output place for the firing transition T_{ji}, w_{ji} will be a positive integer since the place gains tokens after T_{ji} fires. If the place is an input place, w_{ji} will be a negative integer since the place loses tokens after transition firing. The w_{ji}s are constant and will be determined later. $f(T_{ji}, n)$ is a function that is equal to 1 if transition T_{ji} of the Petri net fires in step n and 0 otherwise.

The Petri net that is used here is a stochastic Petri net, i.e. in each step n of the evolution of the Petri net, each of the enabled transitions may fire with a probability. The token count of the places $\mathbf{x}$ and the firing function $\mathbf{f}$ are thus random variables and are shown in bold from here on. Now, if each transition T_{ji} fires with probability $P(T_{ji,n}) = P(\mathbf{f}(T_{ji,n} = 1))$ in step n, then the function $\mathbf{f}$ will be equal to 1 with probability $P(T_{ji,n})$ and 0 with probability $(1 - P(T_{ji,n}))$. Hence, the expected value of $\mathbf{f}$ can be calculated as

$$E(\mathbf{f}(T_{ji}, n)) = P(T_{ji,n}) \times 1 + (1 - P(T_{ji,n})) \times 0 = P(T_{ji,n}) \tag{3.5}$$

Now, we can use the results of equation (3.5) to calculate the expected value of the token count for each place in the Petri net model.

$$X_j(n+1) = X_j(n) + E(\sum_{i=1}^{\rho} w_{ji}\mathbf{f}(T_{ji}, n)) \qquad , \qquad j = 1, 2, ..., N \tag{3.6}$$

X_j is the expected value of $\mathbf{x}_j$ $(X_j = E(\mathbf{x}_j))$. In order to compare the results of the Petri net model with the ODE equations, we can use the expected value of the change in the token count in each step:

$$\begin{aligned} \Delta X_j(n) = E(\Delta \mathbf{x}_j(n)) = E(\sum_{i=1}^{\rho} w_{ji}\mathbf{f}(T_{ji}, n)) \qquad &, \qquad j = 1, 2, ..., N \\ = \sum_{i=1}^{\rho} w_{ji} E(\mathbf{f}(T_{ji}, n)) = \sum_{i=1}^{\rho} w_{ji} P(T_{ji,n}) & \end{aligned} \tag{3.7}$$

By comparing equations (3.1) or (3.2) and (3.7), we can easily note a similarity between the two sets of equations. It is the probability functions $P(T_{ji,n})$ that will determine the result of the Petri net solution. In order to be able to use Petri net results instead of ODE simulations, it is desired to assign the probabilities in such a way that the result of the Petri net network simulations will be similar to the results of ODEs. Note that there is no notion of time in Petri nets. So, the Petri net results $X_j(n)$ contain the information about the state, not the time the state is reached.

Given the similarity between equations (3.2) or (3.1) and (3.7), let us define $\pi_{ji}(x)$ and w_{ji} according to

$$\pi_{ji}(x) = \begin{cases} P(T_{ji}|\mathbf{x}=x) = \dfrac{\prod_{k=1}^{N} x_k^{b_{kji}}}{\sum_{i=1}^{\rho}\sum_{j=1}^{N}\prod_{k=1}^{N} x_k^{b_{kji}}} & if\ T_{ji}\ is\ enabled\ at\ x \\ 0 & otherwise \end{cases} \tag{3.8}$$

$$w_{ji} = a_{ji}$$

The probabilities of course have to be normalized so that,

$$\sum_{i=1}^{\rho}\sum_{j=1}^{N} \pi_{ji}(x) = 1 \tag{3.9}$$

Using the results of equation (3.8), we can write

$$\begin{aligned} P(T_{ji,n}) &= \sum_{x\in R_n} P(T_{ji,n}|\mathbf{x(n)}=x)P(\mathbf{x(n)}=x) \\ &= \sum_{x\in R_n} \pi_{ji}(x)P(\mathbf{x(n)}=x) \\ &= E(\pi_{ji}(\mathbf{x(n)})) \end{aligned} \tag{3.10}$$

Here, R_n is the set of states reachable with n transition firings (from x_0). Using eqution (3.7), we can write

$$E(\Delta \mathbf{x}_j(n)) = \sum_{i=1}^{\rho} a_{ji} E(\pi_{ji}(\mathbf{x(n)})) \tag{3.11}$$

Now, if we write the Taylor expansion of $\pi_{ji}(\mathbf{x})$ around $X = E(\mathbf{x})$, we will have

$$\begin{aligned} &\pi_{ji}(\mathbf{x}) = \pi_{ji,n}(X) + \pi_{ji}^{'}(X)(\mathbf{x} - X) + \frac{1}{2}\pi_{ji}^{''}(X)(\mathbf{x} - X)^2 + ... \\ &\Rightarrow E(\pi_{ji}(\mathbf{x})) = E(\pi_{ji}(X)) + E(\pi_{ji}^{'}(X)(\mathbf{x} - X)) + E(\frac{1}{2}\pi_{ji}^{''}(X)(\mathbf{x} - X)^2) + ... \\ &= \pi_{ji}(X) + \pi_{ji}^{'}(X)E(\mathbf{x} - X) + \frac{1}{2}\pi_{ji}^{''}(X)E((\mathbf{x} - X)^2) + ... \end{aligned} \tag{3.12}$$

Since $E(\mathbf{x} - X) = 0$ and $E((\mathbf{x} - X)^2) = var(\mathbf{x})$, we have

$$E(\pi_{ji}(\mathbf{x})) = \pi_{ji}(X) + \frac{1}{2}\pi_{ji}^{''}(X)var(\mathbf{x}) + ... \tag{3.13}$$

If the variance is small and the higher terms of the expansion equation are negligible, then $E(\pi_{ji}(\mathbf{x})) \simeq \pi_{ji}(X)$ and so

$$\begin{aligned} \Delta X_j(n) &= \sum_{i=1}^{\rho} a_{ji} E(\pi_{ji}(\mathbf{x(n)})) = \sum_{i=1}^{\rho} a_{ji}\pi_{ji}(X(n)) \\ &= \sum_{i=1}^{\rho} a_{ji} \frac{\prod_{k=1}^{N} X_k(n)^{b_{kji}}}{\sum_{i=1}^{\rho}\sum_{j=1}^{N}\prod_{k=1}^{N} X_k(n)^{b_{kji}}} \end{aligned} \tag{3.14}$$

With this probability distribution, it will be shown in the following theorem that the expected state trajectory of the Petri net will be the same as (the scaled version of) the state trajectory of the ODE.

Theorem. Suppose the initial marking of the Petri net is $X_0 = \gamma c_0$. Let $X_j(n)$ $n = 0, 1, ...$ be the expected value of the trajectory of the stochastic Petri net G. Then there exist $t_0 < t_1 < ... < t_n$ such that $\gamma c_j(t_n) = X(n)$ where $c_j(t_n)$ is the numerical solution using Euler's method.

Proof.

From equations (3.2) and (3.14) we can write

$$\begin{aligned}\Delta c_j(t_n) &= \sum_{i=1}^{\rho} a_{ji} \prod_{k=1}^{N} c_k(t_n)^{b_{kji}} \Delta t_n \qquad j = 1, 2, ..., N \\ \Delta X_j(n) &= \sum_{i=1}^{\rho} a_{ji} \frac{\prod\limits_{k=1}^{N} X_k(n)^{b_{kji}}}{\sum\limits_{i=1}^{\rho}\sum\limits_{j=1}^{N}\prod\limits_{k=1}^{N} X_k(n)^{b_{kji}}} \qquad j = 1, 2, ..., N\end{aligned} \tag{3.15}$$

By assumption, we select the initial state of the Petri net $X(0) = \gamma c(0)$. We prove using induction. Suppose $X(n) = \gamma c(t_n)$ for some t_n and prove that $X(n+1) = \gamma c(t_{n+1})$ for some $t_{n+1} > t_n$. From equation (3.15), we have

$$\begin{aligned}\Delta X_j(n) = \sum_{i=1}^{\rho} a_{ji} \frac{\prod\limits_{k=1}^{N} X_k(n)^{b_{kji}}}{\sum\limits_{i=1}^{\rho}\sum\limits_{j=1}^{N}\prod\limits_{k=1}^{N} X_k(n)^{b_{kji}}} = \sum_{i=1}^{\rho} a_{ji} \frac{\prod\limits_{k=1}^{N} (\gamma c_k(t_n))^{b_{kji}}}{\sum\limits_{i=1}^{\rho}\sum\limits_{j=1}^{N}\prod\limits_{k=1}^{N} (\gamma c_k(t_n))^{b_{kji}}} \\ = \sum_{i=1}^{\rho} a_{ji} \frac{\gamma^{\sum_{k=1}^{N} b_{kji}} \prod\limits_{k=1}^{N} (c_k(t_n))^{b_{kji}}}{\sum\limits_{i=1}^{\rho}\sum\limits_{j=1}^{N} \gamma^{\sum_{k=1}^{N} b_{kji}} \prod\limits_{k=1}^{N} (c_k(t_n))^{b_{kji}}}\end{aligned} \tag{3.16}$$

From comparing the results of equation (3.16) and the first equation of (3.15), we can see that if $\Delta t_n = \frac{\gamma^{\sum_{k=1}^{N} b_{kji}}}{\gamma \sum_{i=1}^{\rho}\sum_{j=1}^{N} \gamma^{\sum_{k=1}^{N} b_{kji}} \prod_{k=1}^{N} (c_k(t_n))^{b_{kji}}}$, then we will have $\Delta X_j(n) =$

$\gamma \Delta c_j(t_n)$ and

$$X_j(n+1) = X_j(n) + \Delta X_j(n) = \gamma c(t_n) + \gamma \Delta c_j(t_n) = \gamma c(t_{n+1}) \tag{3.17}$$

So, the result of the Petri net will be equivalent to the numerical solution of the ODEs with the defined Δt_ns. If the step size Δt_n is sufficiently small, the computed approximation will be close to the exact solution. It has been proven that for a wide class of nonlinear problems, the global truncation error of Euler method (the difference between the exact solution and the approximated value) is proportional to the step size [67]. Euler method has second order local truncation error $O(\Delta t_n^2)$ (the error committed in a single step) and first order global truncation error $O(\Delta t)$. We can decrease Δt_n by increasing the scaling factor γ and get more accurate results.

Remark 1: Note that the ODEs do not necessarily have to be in polynomial form, as long as differential equations are in the form of $\frac{dc_j}{dt} = \sum_{i=1}^{\rho} a_{ji} f_{ji}(c)$ in which $c = [c_1, c_2, ..., c_N]^T$ and $f_{ji}(c)$ is any positive nonlinear function that satisfies $f_{ji}(\gamma c) = \gamma^{m_{ji}} f_{ji}(c)$ in which m can be any real number. The probability distribution in the equivalent Petri net will be in the form of $\pi_{ji}(x) = \frac{f_{ji}(x)}{\sum_{i=1}^{\rho} \sum_{j=1}^{N} f_{ji}(x)}$. Also, c_is should be positive in all the solution region, (or if they are negative, should stay negative in all the solution region). Any set of ODEs with this form can be modelled using the proposed method.

Remark 2: In the steady state, the ODE variables $c_j(t)$ are constant, i.e. $\frac{dc_j}{dt} = 0$. Assuming the quantization error (large *gamma*) to be small, we can also assume $\Delta X_j(n) \simeq 0$. Using equation 3.1 for the ODEs or from combining the results of equations (3.7) and (3.8) for Petri nets, we have:

$$\begin{aligned} \sum_{i=1}^{\rho} a_{ji} \prod_{j=1}^{N} c_{j_s}^{b_{kji}} = 0 \\ \sum_{i=1}^{\rho} a_{ji} \prod_{j=1}^{N} x_{j_s}^{b_{kji}} = 0 \end{aligned} \tag{3.18}$$

c_{j_s} and x_{j_s} denote c_j and x_j in steady-state.

Remark 3: If $\gamma c(0)$ is not an integer (as assumed in theorem), then

$$X(0) = \lfloor \gamma c(0) \rfloor = \gamma c(0) - \delta = \gamma(c(0) - \frac{\delta}{\gamma}) \qquad 0 \le \delta < 1 \tag{3.19}$$

Thus, the expected value of the Petri net solution would be equal to the solution of the ODEs for the initial condition $c(0) - \frac{\delta}{\gamma}$. By increasing γ, this solution converges to the solution of ODEs (times γ) for the initial condition $c(0)$ and the result of theorem will still hold.

3.2.2 Simple Example (Continued)

For the Petri net model of Figure 3.1, we have:

$$\begin{aligned} x_1(n+1) &= x_1(n) - f(T_1, n) \\ x_2(n+1) &= x_2(n) - f(T_1, n) + f(T_2, n) \\ x_3(n+1) &= x_3(n) + f(T_1, n) - f(T_2, n) \end{aligned} \tag{3.20}$$

The probability functions for the firings of T_1 and T_2 in the equivalent stochastic Petri net are (when both T_1 and T_2 are enabled)

$$
\begin{aligned}
P(T_1|x) &= \frac{x_1(n)x_2(n)}{x_1(n)x_2(n) + x_3(n)} \\
P(T_2|x) &= \frac{x_3(n)}{x_1(n)x_2(n) + x_3(n)}
\end{aligned}
\tag{3.21}
$$

When one of the transitions is disabled, the firing probability of the other transition will be equal to 1. The execution of the Petri net stops when both transitions are disabled.

Now, in order to show how a Petri net can be simulated with the probability distributions defined in equation (3.8), we run the Petri net with a relatively small initial marking set: $x(0) = [3, 2, 1]^T$. The scaling factor γ is equal to 1. Table 3.1 shows the results:

TABLE 3.1: Simulation Steps of a Simple Petri net

Step (n)	$x(n) = [x_1(n), x_2(n), x_3(n)]^T$	$P(n) = [P_1(n), P_2(n)]^T$	Transition to Fire
0	$[3, 2, 1]^T$	$[0.857, 0.143]^T$	T_1
1	$[2, 1, 2]^T$	$[0.5, 0.5]^T$	T_2
2	$[2, 2, 1]^T$	$[0.8, 0.2]^T$	T_1
3	$[1, 1, 2]^T$	$[0.333, 0.667]^T$	T_2
4	$[1, 2, 1]^T$	$[0.667, 0.333]^T$	T_1
5	$[0, 1, 2]^T$	$[0, 1]^T$	T_2
6	$[0, 2, 1]^T$	$[0, 1]^T$	T_2
7	$[0, 3, 0]^T$	–	–

Figure 3.2 shows the 8 step Petri net simulation results.

Figure 3.4 shows the expected values (X_js) for the same problem obtained by simulation. As we can see by comparing Figures 3.3 and 3.4, the expected values (X_js) are similar to the results of the ODE simulations, even with a very small

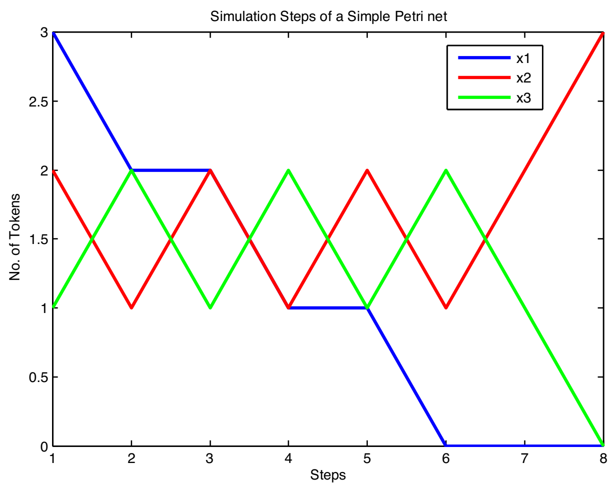

FIGURE 3.2: Simulation Results for the Simple Petri net.

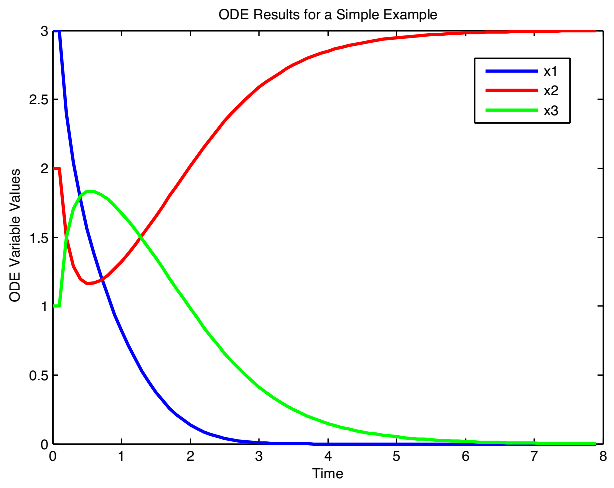

FIGURE 3.3: Simulation Results for the Simple ODE Model.

number of tokens in the Petri net. In order to get finer results, we can increase the scaling factor (e.g. the initial condition can be $x(0) = [300, 200, 100]^T$).

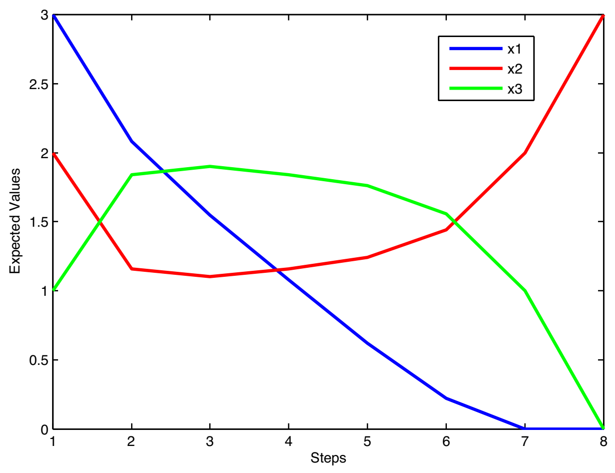

FIGURE 3.4: Expected Values Obtained by Simulation for the Simple Petri net.

3.3 Calculating the Variance

As was mentioned before, the result of the proposed stochastic Petri net is non-deterministic in its nature. In order to be able to have a better discription of the distribution of the solutions, we need to know how far the actual results may spread out from the calculated mean (expected value). In order to have an estimate of the distribution of the results, the variance of the change in the token count is calculated in this section.

$$\begin{aligned} var(\Delta\mathbf{x}_j(n)) &= E(\Delta\mathbf{x}_j(n)^2 - E(\Delta\mathbf{x}_j(n))^2) \\ &= E(\Delta\mathbf{x}_j(n)^2) - E(\Delta\mathbf{x}_j(n))^2 \end{aligned} \tag{3.22}$$

We have already calculated $\Delta X_j(n)$ in (3.7). For $E(\Delta \mathbf{x}_j(n)^2)$, using equation (3.4), we have:

$$
\begin{aligned}
E(\Delta \mathbf{x}_j(n)^2) &= E((\sum_{i=1}^{\rho} w_{ji}\mathbf{f}(T_{ji}, n))^2) \\
&= E(\sum_{i=1}^{\rho} w_{ji}^2 \mathbf{f}(T_{ji}, n)^2 + \sum_{i=1}^{\rho} \sum_{k=1, k\neq i}^{\rho} w_{ji}w_{jk}\mathbf{f}(T_{ji}, n)\mathbf{f}(T_{jk}, n))
\end{aligned}
\tag{3.23}
$$

Now, since we know that in each step during the evolution of the Petri net, only one transition can fire, we know that if $k \neq i$, one of the terms $f(T_{ji}, n)$ and $f(T_{jk}, n)$ will always be zero. So, the term $f(T_{ji}, n)f(T_{jk}, n)$ and subsequently the second term in (3.23) will always be equal to zero. Also, since $f(T_{ji}, n)$ is always equal to 0 or 1, $f(T_{ji}, n)^2 = f(T_{ji}, n)$. Using these results, we will have:

$$
\begin{aligned}
E(\Delta \mathbf{x}_j(n)^2) &= E(\sum_{i=1}^{\rho} w_{ji}^2 \mathbf{f}(T_{ji}, n)) \\
&= \sum_{i=1}^{\rho} w_{ji}^2 E(\mathbf{f}(T_{ji}, n)) \\
&= \sum_{i=1}^{\rho} w_{ji}^2 P(T_{ji,n})
\end{aligned}
\tag{3.24}
$$

So, from (3.22), (3.7) and (3.24), we will have:

$$
var(\Delta \mathbf{x}_j(n)) = \sum_{i=1}^{\rho} w_{ji}^2 P(T_{ji,n}) - (\sum_{i=1}^{\rho} w_{ji} P(T_{ji,n}))^2 \tag{3.25}
$$

3.3.1 Simple Example (Continued)

Figure 3.5 shows the variance values for the simple example. We expected the variance to be high since the quantization of the Petri net was relatively coarse.

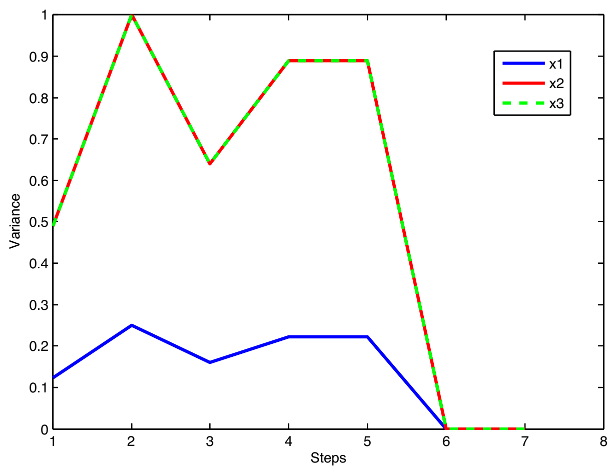

FIGURE 3.5: Variance of the Solution for the Simple Petri net.

3.4 Using Petri Nets to Model Chemical Reaction Networks

This section illustrates how the results of section 3.1. can be applied to chemical reaction networks. A network of chemical reactions with N proteins and ρ reactions is represented as

$$\sum_{j=1}^{N} v_{ji}^{f} c_j \rightleftharpoons \sum_{j=1}^{N} v_{ji}^{r} c_j \qquad i = 1, 2, ..., \rho \tag{3.26}$$

The set of ODEs modeling this system are:

$$\frac{dc_j(t)}{dt} = \sum_{i=1}^{\rho} v_{ji} q_i = \sum_{i=1}^{\rho} v_{ji} (k_i^f \prod_{j=1}^{N} c_j(t)^{v_{ji}^f} - k_i^r \prod_{j=1}^{N} c_j(t)^{v_{ji}^r}) \qquad j = 1, 2, ..., N \tag{3.27}$$

The Petri net equivalent of a set of chemical reactions assigns a transition to each reaction. For the reversible equations, 2 transitions will be used to model the forward and reverse reactions. Note that due to the nature of chemical reaction networks, it will not be necessary to use $\rho \times N$ transitions to model the networks (since there are multiple repetitions in the terms of the equations). So, the equivalent Petri net model will have N places and 2ρ transitions (T_i^f and T_i^r represent the transitions modeling the forward and reverse part of the i^{th} reaction). For each step during the evolution of the Petri net, we can write:

$$\begin{aligned} x_j(n+1) &= x_j(n) + (\sum_{i=1}^{\rho} v_{ji}^r f(T_i^f, n) - \sum_{i=1}^{\rho} v_{ji}^f f(T_i^f, n)) \\ &\quad - (\sum_{i=1}^{\rho} v_{ji}^r f(T_i^r, n) - \sum_{i=1}^{\rho} v_{ji}^f f(T_i^r, n)) \\ &= \sum_{i=1}^{\rho} (v_{ji}^r - v_{ji}^f)(f(T_i^f, n) - f(T_i^r, n)) \qquad , \qquad j = 1, 2, ..., N \end{aligned} \tag{3.28}$$

In which $x_j(n)$ is the token count of the j^{th} place modeling the concentration of the j^{th} protein. $f(T_i, n)$ is a function that will be equal to 1 if transition T_i of the Petri net fires in step n and 0 otherwise. v_{ji}^f and v_{ji}^r are the weights of the arcs that connect transition i to place j equivalent to the parameters of the mass action equation in (3.26).

The token count of a place can change if the transitions modeling the forward or reverse reactions fire. The terms $(\sum_{i=1}^{\rho} v_{ji}^r f(T_i^f, n) - \sum_{i=1}^{\rho} v_{ji}^f f(T_i^f, n))$ and $(\sum_{i=1}^{\rho} v_{ji}^r f(T_i^r, n) - \sum_{i=1}^{\rho} v_{ji}^f f(T_i^r, n))$ account for the change in the number of tokens if transition T_i^f or T_i^r fires (each equation in (3.28) models the effect of two transitions equivalent to the forward and reverse reaction).

Suppose a place is the input place of a transition. It will have v_{ji}^f less tokens after that transition fires if it is a forward transition, and v_{ji}^r less tokens if it is a reverse transition. Similarly, for an output place to which the transition transferes tokens by its firing, it will have v_{ji}^{f} more tokens after that transition fires if is a forward transition and v_{ji}^{r} more tokens if it is a reverse transition.

Now, if each transition T_i fires with probability $P(T_{i,n})$ in step n, then the function f will be equal to 1 with probability $P(T_{i,n})$ and 0 with probability $(1 - P(T_{i,n}))$. Hence, the expected value of f can be calculated as:

$$E(f(T_i, n)) = P(T_{i,n}) \times 1 + (1 - P(T_{i,n})) \times 0 = P(T_{i,n}) \tag{3.29}$$

We use equation 3.29 to calculate the expected value of the concentrations for each protein in the Petri net model

$$X_j(n+1) = X_j(n) + E(\sum_{i=1}^{\rho}(v_{ji}^{r} - v_{ji}^{f})(f(T_i^{f}, n) - f(T_i^{r}, n))) \tag{3.30}$$

For the expected value of the change in the concentrations in each step we have $(v_{ji} = v_{ji}^{r} - v_{ji}^{f})$

$$\Delta X_j(n) = \sum_{i=1}^{\rho} v_{ji}(P^{f}(T_{i,n})) - P^{r}(T_{i,n})) \tag{3.31}$$

As was mentioned before, it is the definition of the probability functions that will determine the result of the Petri net solution. For example, if we assign equal values to the probability functions (i.e. uniform distribution), then the Δx_js will be functions of the difference between the number of arcs that are entering and exiting their equivalent places. So, the concentrations will change with a constant rate untill a place is empty of tokens and the transitions that depend on its tokens to fire are disabled. As we know from the evidence, that is not the case. The more of the reactants there is, the more product will be made. So, we need to assign the probability functions in a way that the transitions whose input places have more tokens, would be more likely to fire than the ones that have less tokens.

From equations (3.26) and (3.31), we can see that the following probability functions (for enabled transitions) will lead to the same results for the Petri net networks as the ODEs.

$$
\begin{aligned}
P^f(T_i) &= \frac{k_i^f \prod_{j=1}^{N} x_j^{v_{ji}^f}}{\sum_{i=1}^{\rho}(k_i^f \prod_{j=1}^{N} x_j^{v_{ji}^f} + k_i^r \prod_{j=1}^{N} x_j^{v_{ji}^r})} \\
P^r(T_i) &= \frac{k_i^r \prod_{j=1}^{N} x_j^{v_{ji}^r}}{\sum_{i=1}^{\rho}(k_i^f \prod_{j=1}^{N} x_j^{v_{ji}^f} + k_i^r \prod_{j=1}^{N} x_j)^{v_{ji}^r})}
\end{aligned}
\tag{3.32}
$$

As we can figure out from equation (3.32), with such probability distributions, the probability of firing of a transition will depend on the current concentrations of its input places as well as the rate of the reactions. The simulation results for both probability distributions will be provided in Chapter 4 for comparison.

3.4.1 Algorithm 1

Using the results of equation (3.32), we can now propose an algorithm for calculating the trajectories of the protein concentrations using the Petri net model.

□ **Algorithm 1.**

Step1. Set the initial marking $x(0)$ using the initial protein concentration values and the scaling factor γ. Set $i = 1$;

Step2. Using Petri net equations, find the enabled transitions.

Step3. Calculate the probabilities for the transition firings of the enabled transitions using equation (3.32).

Step4. Choose a random transition to fire using the probability distribution calculated in the previous step. Fire the selected transition and calculate the new marking $x(i)$. Set $i = i + 1$;

Step5. Return to step 2 and continue until there is no enabled transition left or another preset condition is satisfied.

The algorithm stops when a preset condition such as a maximum number of steps is satisfied or all the transitions are disabled in the final marking. Adding the preset condition helps in cases that the Petri net has a periodic behaviour (rather than a constant steady state). In such cases, one can either use a maximum for number of steps to stop the algorithm or add another condition (e.g. maximum for the number of times a certain state is visited).

3.5 Sensitivity Analysis

One of the main questions that arises after building a model of a biological system is how the system can be manipulated to achieve a desired behaviour (control objective). To address this question clearly, first we need to determine the inputs and outputs of the system so that the control objective can be defined.

Biological systems are naturally stabled by multiple feedback loops that regulate the entire system and ensure that biological entities maintain homeostasis. In case of a malfunction of one or a series of such regulatory mechanisms, diseases may occur. In such cases, control strategies (drugs) may be considered to force the system back to its normal behaviour.

In order to help design drugs that control the behaviour of a system, we need to know the cause-effect relationships between system elements, in the case of protein interactions, which protein(s) have a higher influence on the desired output protein(s). So, the inputs and outputs of the system will be proteins or places in the Petri net model. The question of the degree of influence translates into a sensitivity analysis in such networks. The researcher would like to know perturbing which protein(s) can result in a more significant change in the output protein concentration. Such proteins will offer possible targets for drug design.

As was explained in Chapter 2, one of the commonly used methods of sensitivity analysis for chemical reaction networks is calculating the sensitivities throughout the trajectories of the solution. As was mentioned in section 2.4.1., the sensitivities for the ODE model can be calculated using the following equations

$$\begin{aligned} S_q^c(t) &= \frac{\partial c(t,q)}{\partial q}|_{q=q_0} = lim_{\Delta q \to 0} \frac{c(t, q_0 + \Delta q) - c(t, q_0)}{\Delta q} \qquad \forall t \geq 0 \\ s_q^c &= \int_0^T S_q^c(t) dt \end{aligned} \tag{3.33}$$

Using a similar notion, we define the sensitivity measures for the equivalent Petri net model. Here, sensitivity is defined as the average range (along an entire trajectory) over which the tokens representing the output protein fluctuate given certain excitation at the initial condition of the Petri net.

$$s(x^i, x^o, x_0) = \frac{\frac{1}{No}\sum_{k=1}^{No} |x^o_{per}(k) - x^o_{ref}(k)|}{|x^i_{per}(0) - x^i_{ref}(0)|} \tag{3.34}$$

where $s(x^i, x^o, x_0)$ is the sensitivity of output x^o to input x^i while the initial condition is x_0. x^o_{per} and x^o_{ref} are vectors of the trajectories of the perturbed and reference outputs, respectively. No is the length of the trajectories (the number of steps for which both trajectories reach steady state or a maximum limit). The simulation results for the Apoptosis network will be provided in Chapter 4. The results of a sensitivity analysis method performed based on ODEs in MATLAB's SimBiology will also be provided for comparison.

3.5.1 Algorithm 2

Using Algorithm 1 and the results of equation (3.34), we can now propose an algorithm for calculating the sensitivities of a protein selected as the output of the system to other proteins in the Petri net. Algorithm 2 calculates the sensitivity of the output x^o with respect to input x^i with the initial condition $x(0) = x_0$.

□ **Algorithm 2.**

Step1. Set the initial marking $x(0)$ and calculate the unperturbed trajectory x_{ref} using Algorithm 1. Set δ *.

Step2. Replace the initial state of the Petri net with $x_0 + \delta e_i$ where e_i is a vector whose i^{th} element is equal to 1, and all its other elements are equal to zero. Calculate the perturbed trajectory x^o_{per} for input i using Algorithm 1.

Step3. Use equation (3.34) to calculate the sensitivity measure $s(x^i, x^o, x_0)$ for input x^i. If $i < N$, set $i = i + 1$ and return to Step 2.

* δ is the amount of perturbation of the input. It should be set according to the value of the initial markings. Choosing a very small δ (compared with the standard deviation) will lead to insignificant results in the sensitivity analysis since the trajectories are calculated stochastically and are never exactly the same. Increasing δ should result in more significant changes in the output of the system. On the other hand, if δ is chosen too large, it will change the state of the Petri net and the trajectory entirely and it can no longer be considered as a perturbation. Usually, 1 or 2 percent of the average initial marking of the places is a reasonable value for δ.

The length of the trajectories (No) should be chosen so that both trajectories reach the steady state.

3.6 Scaling Effect

The token counts in the Petri net places are always positive integers while the values of x_{ji} in the ODEs are (non-negative) real numbers. So, the equivalent Petri net model only works for studying ODEs that only have solutions in the positive region (or satisfy the conditions explained above). The accuracy of the solution of the Petri net model is determined by its scaling factor γ, which basically determines how many tokens represent a concentration unit in the ODE equation. While this scaling is inevitable due to the fact that Petri nets are discrete tools in their nature, determining a good scaling factor can get us any desirable precision at the cost of increasing computation time. So, there is a trade off between lower error and shorter computation time. In order to show the effect of scaling on the solutions, the simulations have been performed with various scaling factors and will be provided in Chapter 4.

The initial condition of the Petri net only needs to be proportionate to the initial condition set of the ODE system,

$$x_0 = \lfloor \gamma c_0 \rfloor \tag{3.35}$$

in which x_0 and c_0 are the initial conditions for the Petri net and ODE system respectively, and γ is a positive integer, $\lfloor \gamma c_0 \rfloor$ is the floor of γc_0, i.e. the largest positive integer that is lower than γc_0. This leaves us with a degree of freedom in choosing a reasonable scaling factor that would result in desirable accuracy as well as acceptable computation time. The effect of scaling of concentration values will be examined in Section 4.5.

3.7 Parallel Computing

One of the methods that can be used to improve the accuracy of the solution without significantly increasing the computation time is parallel computing. We can use parallel computing to simulate several independent Petri nets with the same parameters and average the results of these simulations in order to get a more accurate solution. For example, instead of increasing the scaling factor, we can run several parallel simulations and average the results to obtain a much better solution as well as avoiding a major increase in computation time. The simulation results will be provided in Chapter 4 for comparison.

3.8 Summary

In this chapter, a method was presented to solve a system of ordinary differential equations of polynomial form by building and simulating their "equivalent stochastic" Petri net models. Next, it was shown how this approach can be applied to the simulation of chemical reactions. These Petri net models may later be used for performing sensitivity analysis. This approach will be applied in the next chapter to illustrate the results on a case study. It will be shown that the simulations can be done significantly faster compared to a numerical solution of ODE if parallel processing capabilities are available.

Chapter 4

Case Study

In this chapter, the proposed method of Chapter 3 is applied to the Apoptosis protein network example. The results are then compared with the ODE simulations and the differences and advantages of each method are discussed.

4.1 Cell Apoptosis

The pathway selected as an example in this study is the Caspase Apoptosis pathway [1]. Apoptosis is the process of programmed cell death that occurs in response to a stress and brings about cell suicide. This process is controlled by a diverse range of either extra-cellular or intra-cellular signals that trigger or inhibit Apoptosis (positive or negative regulation).

When the normal functioning of the apoptotic pathway is disrupted in a way that the cell would not be able to undergo Apoptosis anymore, several types of diseases may occur. Such diseases are a result of cells that live past their "use-by-date" and are able to replicate and pass on any faulty machinery to their replicates. These malfunctions increase the likelihood of the cell becoming cancerous or diseased [68, 69].

Excess Apoptosis (losing control of the rate of cell death) can also lead to neurodegenerative diseases (such as Alzheimer, Huntington and Parkinson disease), hematologic diseases, and tissue damage. For example, the progression of HIV is directly linked to excess and unregulated Apoptosis [70].

The *Caspase proteins* are the main building blocks of the Apoptotic pathway. There are two types of Apoptotic Caspases: initiator Caspases such as Caspase 2, 8, 9, and 10, and effector Caspases such as Caspase 3, 6, and 7 [71]. Initiator Caspases are only activated when binded to specific proteins. After activation, the initiator Caspases activate the effector Caspases. Then, the activated effector Caspases degrade a host of intracellular proteins to perform Apoptosis and carry out the cell death program [72].

XIAP (X-linked inhibitor of Apoptosis protein) is a protein that stops apoptotic cell death [73]. XIAP stops apoptotic cell death that is induced either by viral infection or by overproduction of Caspases. XIAP is a member of the inhibitors of apoptosis family of proteins (IAP) [74–76]. It has a domain that inhibits the activity of different types of Caspases. XIAP inhibits Caspase 3 activity by binding to the active-site where a protein would normally bind during apoptosis, blocking it access [77, 78]. As a result, the protein will no longer be able to bind and trigger apoptosis.

Caspases 3 and 8 are proteins that interact during the apoptosis procedure. The sequential activation of these Caspases plays a central role in the execution-phase of cell apoptosis. Caspase 3 is activated in the apoptotic cell both by extrinsic and intrinsic pathways. Control of caspase 3 is necessary in this process because if unregulated, Caspase activity would kill cells in an uncorolled way.

Caspase 3^* is a protein that lices certain critical proteins at specific amino acid residuals in the cell. When the cell encounters a trigger, the level of Caspase 3^* activation in the cell goes up, which means that the death signal (Apoptotic signal) is directly proportional to the levels of Caspase 3^* activation in the cell [1].

Figure 4.1 shows the network of Caspase cascade together with XIAP and a possible drug control mechanism that has been obtained by analyzing the differential equations of the chemical reactions and added to the system later [1].

The blue-coloured section that is distinguished as the tissue shows the set of proteins which interact to carry out the procedure of apoptosis. As was explained before, Casp3^* (Caspase 3^*) is the output of the system since the changes in its level are proportional to the rate of apoptosis. All other proteins are considered as the inputs of the system which can be targeted and triggered in order to induce a change in the behaviour of the system. The circles that are connected to the proteins by arrows show the chemical reactions between the proteins, i.e. which proteins interact to produce which protein. The arrows specify the direction of the chemical reaction. The circles with two arrows above them specify the reactions that are reversible, i.e. occur in both ways.

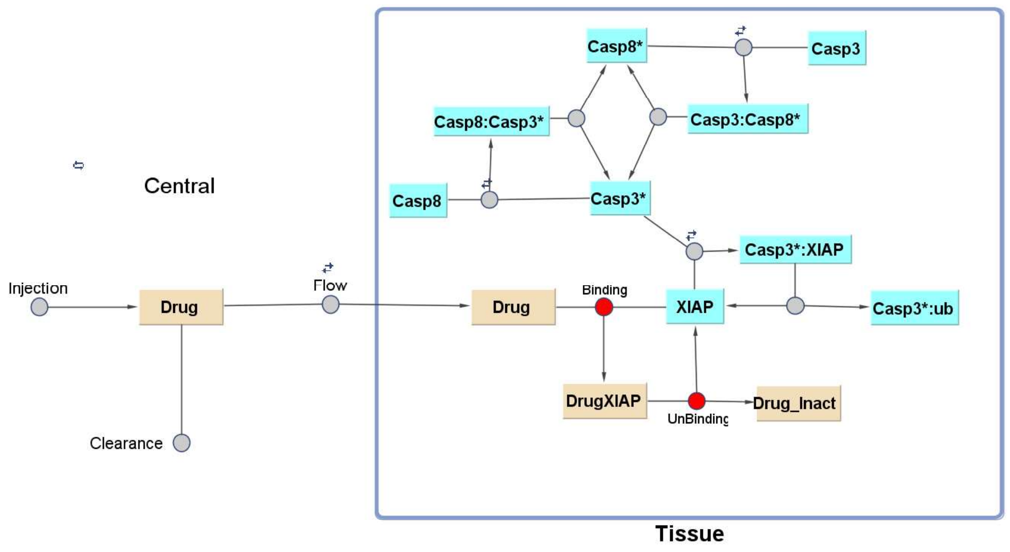

FIGURE 4.1: Apoptosis Pathway Diagram [1].

Considering the importance of biological processes such as Apoptosis, it is of great value to model and analyze their underlying pathways and try to understand the behaviours that arise from the structural properties of such pathways. This information can later be used in the design of control mechanisms with the aim of finding drugs to regulate the behaviour of these processes.

4.2 Modelling Apoptosis Network Using Petri Nets

This section covers the problem formulation of the Apoptosis protein network that has been used throughout the thesis for case study. The aim is to provide a systematic framework for creating discrete-event models (more specifically, Petri net models) based on the ODE based models of protein-protein interactions and calculate protein concentrations and perform sensitivity analysis based on theses Petri net models.

Apoptosis pathway has been extensively studied, and the equations governing the dynamics of Apoptosis can be found in literature [79]. In particular it has been studied using SimBiology for MATLAB. The resulting simulation and sensitivity analysis results will later be used for comparison with those of our proposed technique.

The chemical reactions modeling the Apoptosis network are

$$\begin{aligned}
&\mathit{Reaction}\ 1: && [Casp3^*] + [Casp8] \longleftrightarrow [Casp8 : Casp3^*] \\
&\mathit{Reaction}\ 2: && [Casp8 : Casp3^*] \longrightarrow [Casp8^*] + [Casp3^*] \\
&\mathit{Reaction}\ 3: && [Casp8^*] + [Casp3] \longleftrightarrow [Casp3 : Casp8^*] \\
&\mathit{Reaction}\ 4: && [Casp3 : Casp8^*] \longrightarrow [Casp8^*] + [Casp3^*] \\
&\mathit{Reaction}\ 5: && [Casp3^*] + [XIAP] \longleftrightarrow [Casp3^* : XIAP] \\
&\mathit{Reaction}\ 6: && [Casp3^* : XIAP] \longrightarrow [XIAP] + [Casp3^* : ub]
\end{aligned} \tag{4.1}$$

All reactions have mass action kinetics. [A] is the concentration of protein A (for example in moles per liter). Each reaction i has a forward reaction rate k_i and a reverse reaction rate kd_i if it is reversible. From the above chemical reaction equations, we can write the following ODE equations:

$$
\begin{aligned}
\frac{d[Casp3^*]}{dt} &= kd_5[Casp3^* : XIAP] - k_5[Casp3^*][XIAP] + kd_1[Casp8 : Casp3^*] \\
&- k_1[Casp3^*][Casp8] + kd_4[Casp3 : Casp8^*] + kd_2[Casp8 : Casp3^*] \\
\frac{d[Casp8^*]}{dt} &= kd_3[Casp3 : Casp8^*] - k_3[Casp8^*][Casp3] + kd_4[Casp3 : Casp8^*] \\
&+ kd_2[Casp8 : Casp3^*] \\
\frac{d[Casp3]}{dt} &= kd_3[Casp3 : Casp8^*] - k_3[Casp8^*][Casp3] \\
\frac{d[Casp8]}{dt} &= kd_1[Casp8 : Casp3^*] - k_1[Casp3^*][Casp8] \\
\frac{d[XIAP]}{dt} &= kd_5[Casp3^* : XIAP] - k_5[Casp3^*][XIAP] + kd_6[Casp3^* : XIAP] \\
\frac{d[Casp3 : Casp8^*]}{dt} &= -kd_3[Casp3 : Casp8^*] + k_3[Casp8^*][Casp3] \\
&- kd_4[Casp3 : Casp8^*] \\
\frac{d[Casp8 : Casp3^*]}{dt} &= -kd_1[Casp8 : Casp3^*] + k_1[Casp3^*][Casp8] \\
&- kd_2[Casp8 : Casp3^*] \\
\frac{d[Casp3^* : ub]}{dt} &= kd_6[Casp3^* : XIAP] \\
\frac{d[Casp3^* : XIAP]}{dt} &= -kd_5[Casp3^* : XIAP] + k_5[Casp3^*][XIAP] \\
&- kd_6[Casp3^* : XIAP]
\end{aligned}
\tag{4.2}
$$

These reactions are used to build the Petri net model of the Apoptosis network in Figure 4.2. In order to be able to compare the results of the Simbiology model with Petri net simulation results, the parameters of the system are exported from Simbiology and the same parameters are used to build the Petri net model. Table 4.1 lists the transitions along with the corresponding reactions in the chemical reaction network in Equation (4.1). The purple arcs in Figure 4.1 belong to the transitions that model the reverse reactions.

TABLE 4.1: Mappings of the Transitions of the Petri net and the Reactions of the Protein Network.

Reaction	Transition(s)
Reaction 1	T_5, T_6
Reaction 2	T_4
Reaction 3	T_1, T_2
Reaction 4	T_3
Reaction 5	T_7, T_8
Reaction 6	T_9

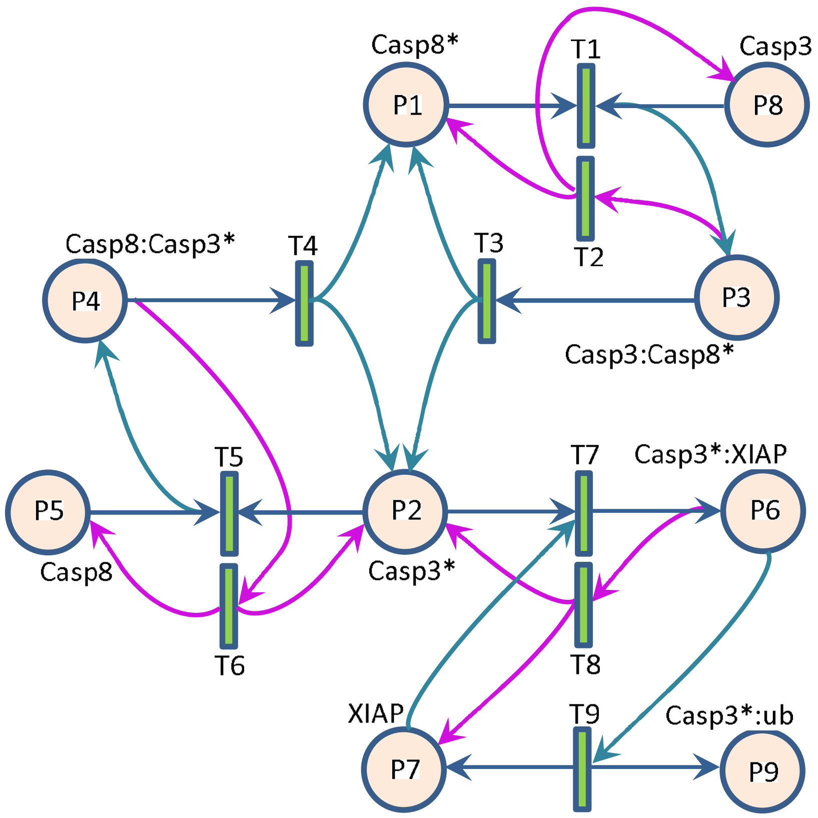

FIGURE 4.2: Apoptosis Pathway Petri net.

Since we want to compare the results of the Petri net model with ODE simulations, the same initial condition set has been chosen for the Petri net model. Note that the token counts in the initial condition of a Petri net are always positive integers. The initial condition of the Petri net only needs to be proportional to the initial condition set of the ODE system. For example, if the initial condition of the ODE system is $[1.3, 2.4, 0.82, 3]^T$, the initial condition of the Petri net can be any of the integer sets of numbers $[130, 240, 82, 300]^T$, $[260, 480, 164, 600]^T$, ... as long as the values are porportional to the values of the equivalent ODE model.

Starting from this initial condition set, the Petri net model is simulated to calculate the changes in the concentration of the proteins from the initial condition to the steady state (or until some other desired condition is satisfied).

4.3 Steady State

From equation (3.18), for the Apoptosis network in the steady state, we can write:

$$\begin{aligned}
&kd_5[Casp3^* : XIAP] - k_5[Casp3^*][XIAP] + kd_1[Casp8 : Casp3^*] \\
&- k_1[Casp3^*][Casp8] + kd_4[Casp3 : Casp8^*] + kd2[Casp8 : Casp3^*] = 0 \\
&kd_3[Casp3 : Casp8^*] - k3[Casp8^*][Casp3] + kd_4[Casp3 : Casp8^*] \\
&+ kd2[Casp8 : Casp3^*] = 0 \\
&kd_3[Casp3 : Casp8^*] - k_3[Casp8^*][Casp3] = 0 \\
&kd_1[Casp8 : Casp3^*] - k_1[Casp3^*][Casp8] = 0 \\
&kd_5[Casp3^* : XIAP] - k_5[Casp3^*][XIAP] + kd_6[Casp3^* : XIAP] = 0 \\
&k3[Casp8^*][Casp3] - kd_3[Casp3 : Casp8^*] - kd_4[Casp3 : Casp8^*] = 0 \\
&k3[Casp3^*][Casp8] - kd_1[Casp8 : Casp3^*] - kd_2[Casp8 : Casp3^*] = 0 \\
&kd_6[Casp3^* : XIAP] = 0 \\
&k5[Casp3^*][XIAP] - kd_5[Casp3^* : XIAP] - kd_6[Casp3^* : XIAP] = 0
\end{aligned} \tag{4.3}$$

From the above equations, we will have:

$$\begin{aligned}
&[Casp3^* : XIAP] = 0\\
&[Casp3 : Casp8^*] = 0\\
&[Casp8 : Casp3^* *] = 0\\
&[Casp3^*][XIAP] = 0\\
&[Casp8^*][Casp3] = 0\\
&[Casp8][Casp3^*] = 0
\end{aligned} \tag{4.4}$$

The results are the same for both the ODEs and Petri net equations. From the equations in (4.4), we can see that the number of the independent equations obtained from the 9 equations in (4.3) is equal to the number of chemical reaction equations. Also, we can see that the steady state values for some proteins are not determined by the reaction rate values and will be zero independent of the change in the reaction rates.

4.4 Ordinary Differential Equations and Petri net Model Simulations

Figure 4.3 shows the results of Petri net simulations performed by calculating the probabilities using Algorithm 1 in Chapter 3. Figure 4.4 shows the results of ODE simulations in MATLAB's SimBiology. The Petri net simulations were stopped at the point where all the transitions were disabled. Since the initial conditions of the ODEs are positive integers, the same values are used for Petri net simulations as well. The inital marking of the Petri net and the initial concentration of the ODE variables are given in Table 4.2[1].

TABLE 4.2: Initial Conditions for Petri net and ODE Simulations ($\gamma = 1$)

Place	Protein Name	Petri net Initial Condition (Tokens)	ODE Initial Condition (Moles)
P1	$Casp8^*$	10000	10000
P2	$Casp3^*$	10000	10000
P3	$Casp3 : Casp8^*$	0	0
P4	$Casp8 : Casp3^*$	0	0
P5	$Casp8$	9000	9000
P6	$Casp3^* : XIAP$	0	0
P7	$XIAP$	8000	8000
P8	$Casp3$	25000	25000
P9	$Casp3^* : ub$	0	0

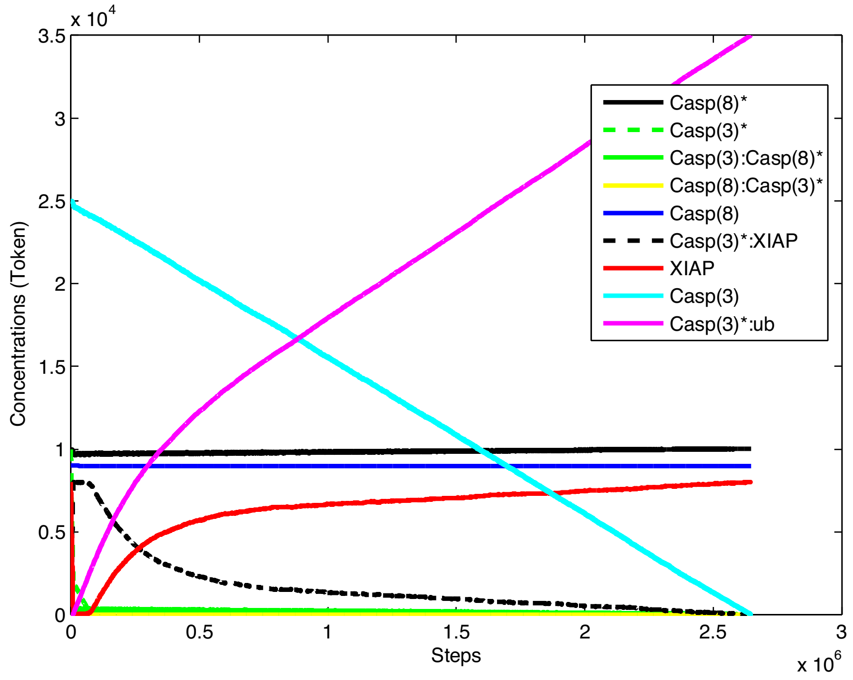

FIGURE 4.3: Protein Concentrations Obtained by Petri net Simulations ($\gamma = 1$).

[1]The values are exported from SimBiology, and the ratio of the values is important (not the exact value of the concentrations).

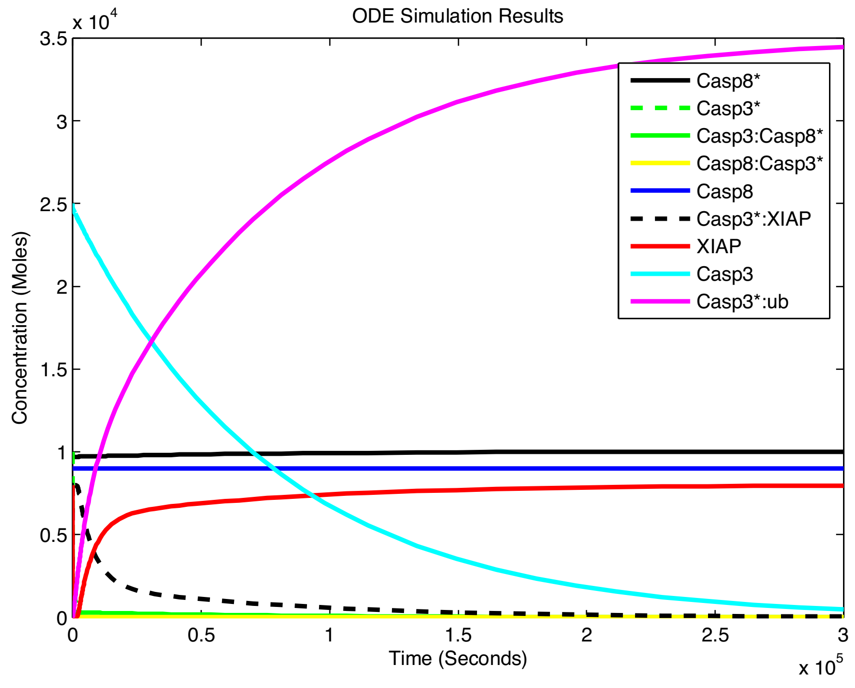

FIGURE 4.4: Protein Concentrations Obtained by ODE Simulations.

By comparing the two figures, we can easily see the similarity. Note that since the steps in the Petri net are not necessarily the same as the ODE steps, (i.e. n is not a linear function of time t) the output signals of the Petri net appear to have a warped time axis. However, the values of the protein concentrations go through the same changes and stablize at the same values (the steady state values are the same), i.e. for every n, there exists t_n such that $x(n) = \gamma c(t_n)$(ignoring the small quantization error). In other words, there is a transformation that can map the time in the ODE (t) to the steps in the Petri net simulations (n). Figure 4.5 shows the relationship between Petri net steps (n) and ODE time (t). Each point in Figure 4.5 represents a Petri net step and ODE time value in which the concentration of the proteins are the same in both models. For instance, the protein concentrations from the ODE at $t = 4 \times 10^4$ are (almost equal to protein concentrations obtained from the Petri net at $n = 2 \times 10^5$).

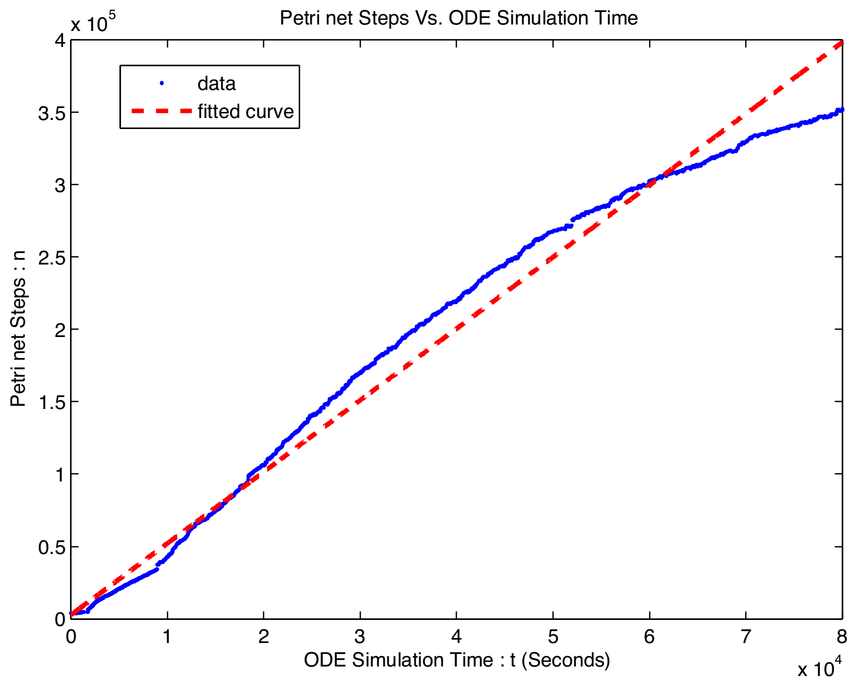

FIGURE 4.5: Comparison of Petri net Simulation Steps and ODE Time During the Transient Behaviour.

To show that the protein concentrations have the same values in Petri net and ODE simulations, we plotted a phase diagram for two of the variables. Figure 4.6 shows the phase diagram for $Casp3^*$ and $Casp3^* : XIAP$ obtained from the above Petri net and ODE simulation results ($\gamma = 1$). As it is clear from the figure, the values have gone through the same trajectories for both models, i.e. have equal phase diagrams.

In order to show how the results may change by changing the probability functions, we have also performed the same simulations with uniform probability distributions for all transitions. Figure 4.7 shows the results of these simulations. As we expected, all the enabled transitions fire with equal probabilities at first

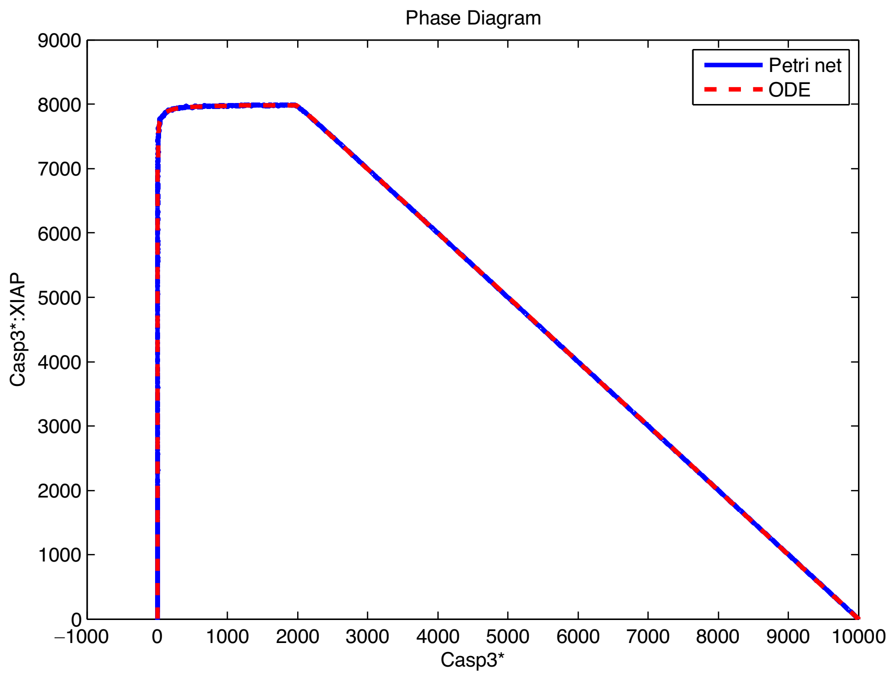

FIGURE 4.6: Phase Diagram of $Casp3^*$ and $Casp3^* : XIAP$ for Petri net and ODE Simulations.

(all the lines have equal slopes since the tokens are removed from/added to the places with constant equal rates) until a transition is disabled and can no longer fire. Then, all the other transitions which are still enabled, fire with new probabilities which are again equal, so the slopes change, but still remain equal until no transition can fire anymore.

Figure 4.8 shows the phase diagram of the same proteins for the ODEs and the Petri net with uniform probability distributions. As we can see in the figure, choosing a different probability distribution will lead to a trajectory completely different from that of the ODE.

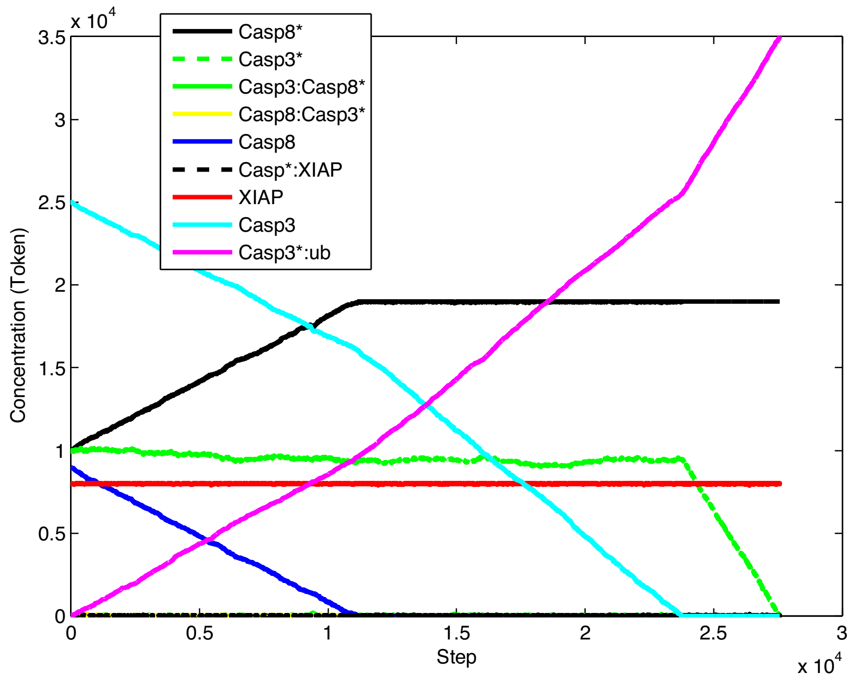

FIGURE 4.7: Protein Concentrations Obtained by Petri net Simulations with Uniform Probability Distributions ($\gamma = 0.1$).

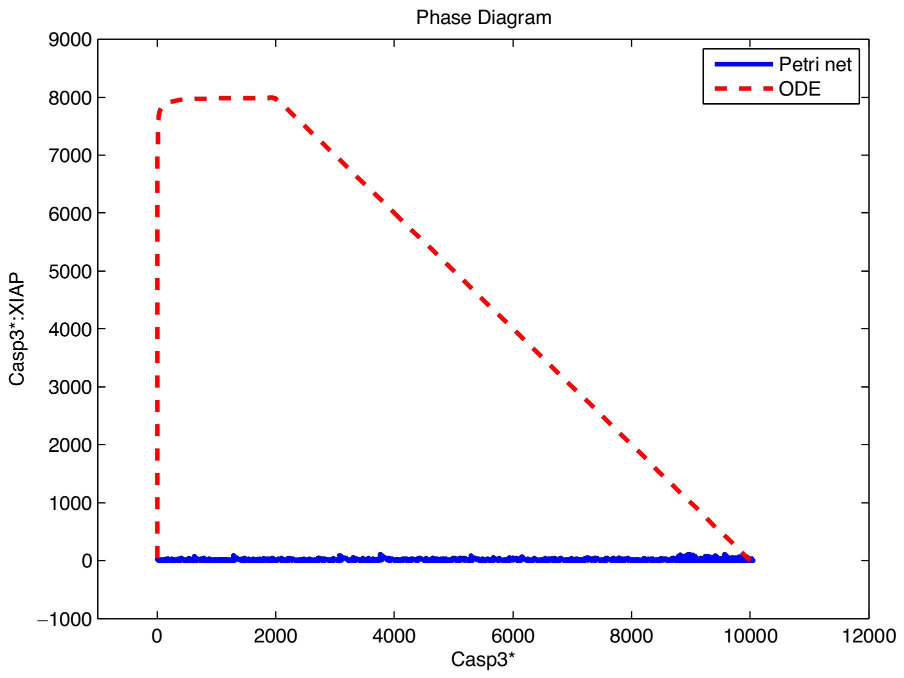

FIGURE 4.8: Phase Diagram of $Casp3^*$ and $Casp3^* : XIAP$ for Petri net with Uniform Probability Distributions and ODE Simulations.

4.5 Scaling Factor

As was discussed in Chapter 3, the Petri net simulations can be performed with different scaling factors that would result in different accuracies and computation times. Figures 4.9, 4.10, 4.11, 4.12, 4.13, and 4.14 show the simulation results for 6 different scaling factors.

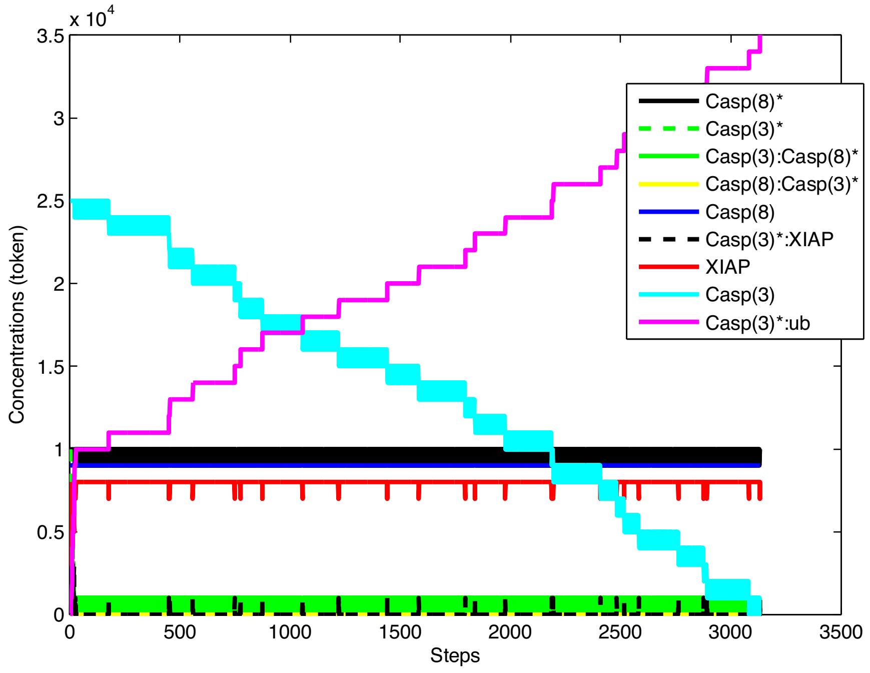

FIGURE 4.9: Effect of Scaling Factor - Petri net Simulation Results ($\gamma = 0.001$).

In order to be able to see the effect of increasing scaling factor better, the phase diagram of the same variables were plotted for all the simulations with different scaling factors. As is clear from Figure 4.15, the phase diagrams get more accurate as scaling factor is increased.

In order to be able to compare the results of the Petri net and ODE solutions directly, we defined an error measure based on the phase diagrams of the solution.

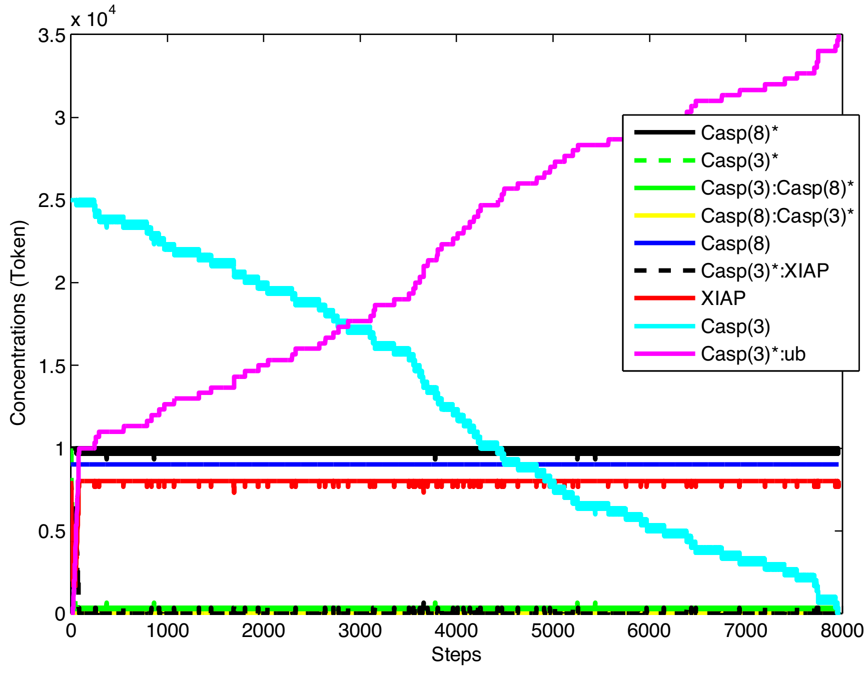

FIGURE 4.10: Effect of Scaling Factor - Petri net Simulation Results ($\gamma = 0.003$).

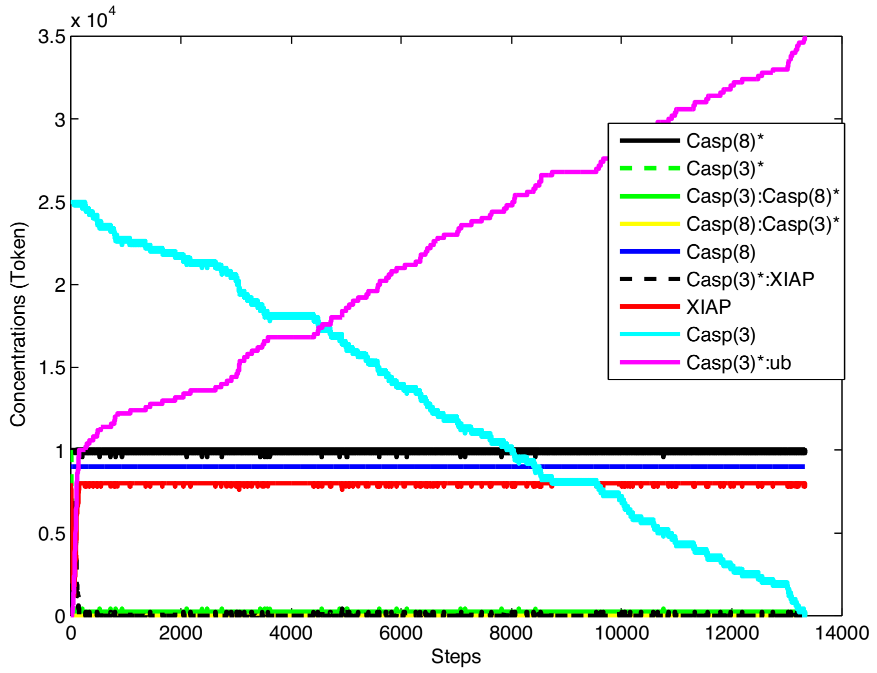

FIGURE 4.11: Effect of Scaling Factor - Petri net Simulation Results ($\gamma = 0.005$).

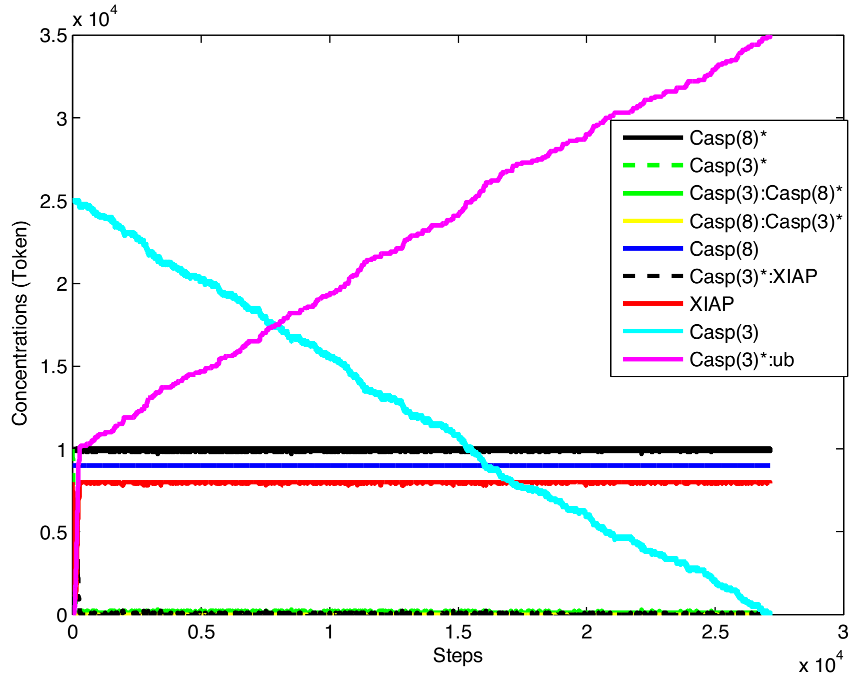

FIGURE 4.12: Effect of Scaling Factor - Petri net Simulation Results ($\gamma = 0.01$).

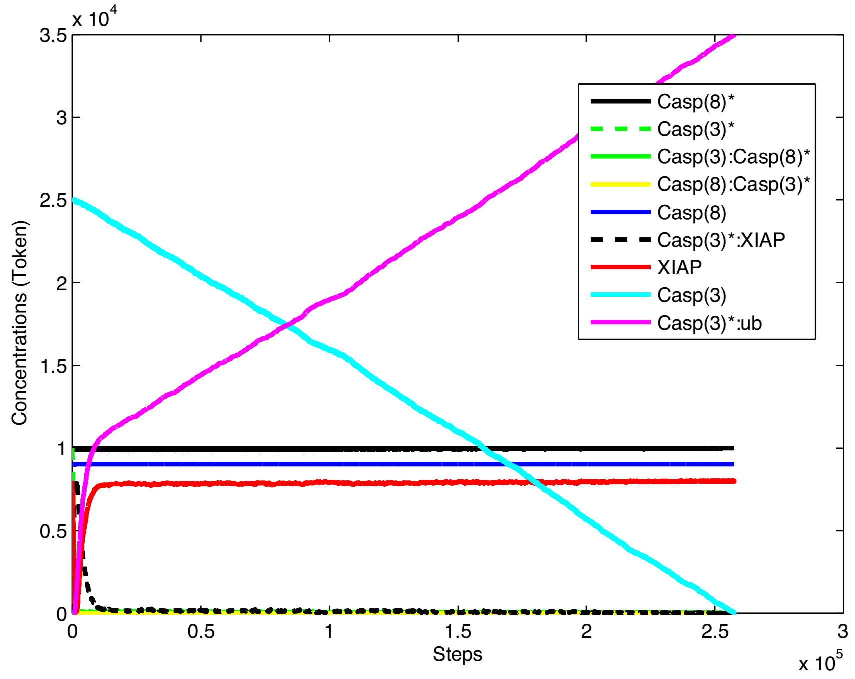

FIGURE 4.13: Effect of Scaling Factor - Petri net Simulation Results ($\gamma = 0.1$).

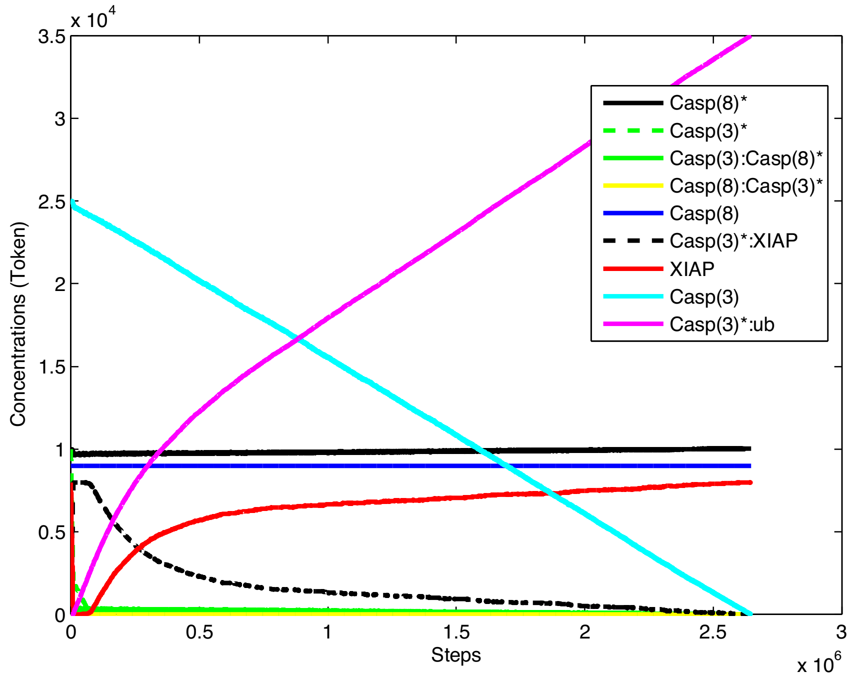

Figure 4.14: Effect of Scaling Factor - Petri net Simulation Results ($\gamma = 1$).

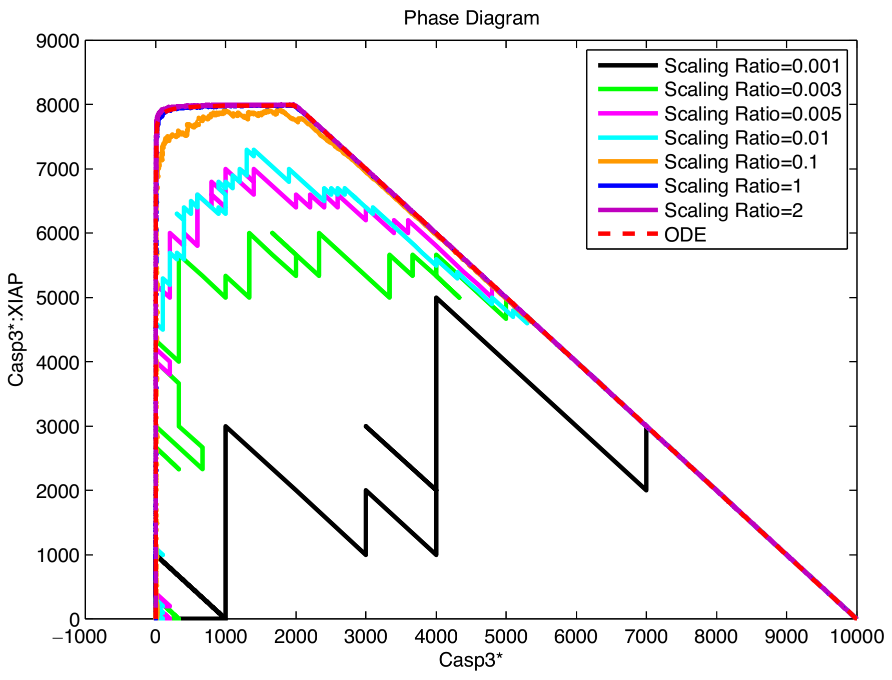

Figure 4.15: Phase Diagram of $Casp3^*$ and $Casp3^* : XIAP$ for Petri nets with Different Scaling Ratios and ODE Simulations.

The phase diagrams of all the variables were calculated based on one of the variables which had a monotonically decreasing curve in the solution ($Casp3^*$). Using these phase diagrams, the root of the mean square of the difference between the solutions of ODE and Petri net is calculated as the measure of error. Table 4.3 shows these root mean squared errors (RMSE) for each of the proteins in comparison with the results of the ODEs. Table 4.4 shows the simulation time for each scaling factor (γ) and the average RMSEs for all the protein concentraions (the simulation time for the ODE results of MATLAB's SimBiology was 1.6729 seconds).

TABLE 4.3: RMSE (Tokens) for Different Scaling Ratios

Scaling Ratio $\gamma \rightarrow$	0.001	0.003	0.005	0.01	0.1	1
Species $\downarrow$	Fig 4.9	Fig 4.10	Fig 4.11	Fig 4.12	Fig 4.13	Fig 4.14
Casp8*	4.3337	4.0313	4.3337	4.1958	3.6300	0.1858
Casp3:Casp8*	4.5069	4.1694	4.5069	4.3718	3.7990	0.1860
Casp8:Casp3*	0.0235	0.0235	0.0235	0.0234	0.0227	0.0132
Casp8	0.2208	0.2208	0.2208	0.2199	0.2120	0.0150
Casp3*:XIAP	112.0822	42.3476	24.5974	22.9377	4.7571	0.0463
XIAP	112.0822	42.3476	24.5974	22.9377	4.7571	0.0463
Casp3	9.4702	8.7924	9.4702	9.3606	8.0021	0.2122
Casp3*:ub	108.6840	38.7770	21.1065	19.2861	6.0777	0.1230
Mean $\rightarrow$	43.9254	17.5887	11.1070	10.4166	3.9072	0.1035

TABLE 4.4: Petri net Simulation Times and Average RMSEs for Different Scaling Ratios (γ)

Figure	Scaling Factor (γ)	Simulation Time (Seconds)	Average RMSE (Tokens)
Figure 4.9	0.001	0.0213	43.9254
Figure 4.10	0.003	0.0639	17.5887
Figure 4.11	0.005	0.0925	11.1070
Figure 4.12	0.01	0.1926	10.4166
Figure 4.13	0.1	1.2374	3.9072
Figure 4.14	1	21.1901	0.1035

4.6 Expected Values and Parallel Computing

In order to find the expected values of trajectories (as described in Section 3.2.) we need to calculate several samples of the trajectory, and find the average result. To speed up the calculations whenever possible, we can use parallel programming. Figures 4.16, 4.17, 4.18, and 4.19, show the simulation results for average trajectories for various scaling factors and number of sample trajectories. Table 4.5 shows the simulation time for each parallel simulation as well as the average RMSEs (of $x_1, x_2, ..., x_N$), and Table 4.6 compares the results of these parallel simulations with the results of the previous simulations. The RMSE values for each protein in each simulation, and the average RMSE for all the proteins in the network are included in the table. As can be seen in the results, using averaging with parallel computing can help improve the results in terms of speed and accuracy.

TABLE 4.5: Simulation Time for Different Parallel Simulations

Figure	Number of Averaged Simulations	Scaling Factor (γ)	Simulation Time per Processor(Seconds)	RMSE (Tokens)
Figure 4.16	8	0.003	0.0639	15.0624
Figure 4.17	1000	0.003	0.0639	13.5829
Figure 4.18	8	0.01	1.2374	3.0510
Figure 4.19	1000	0.01	1.2374	2.3913
Figure 4.10	1	0.003	0.0639	17.5887
Figure 4.11	1	0.005	0.0925	11.1070
Figure 4.12	1	0.01	0.1926	10.4166
Figure 4.13	1	0.1	1.2374	3.9072
Figure 4.14	1	1	21.1901	0.1035

The phase diagram of the $Caspase3^*$ and $Caspase3^* : XIAP$ were plotted for all the above simulations in Figures 4.20 and 4.21. As is clear from Figures 4.20 and 4.21, the phase diagrams improve and get smoother when the results are

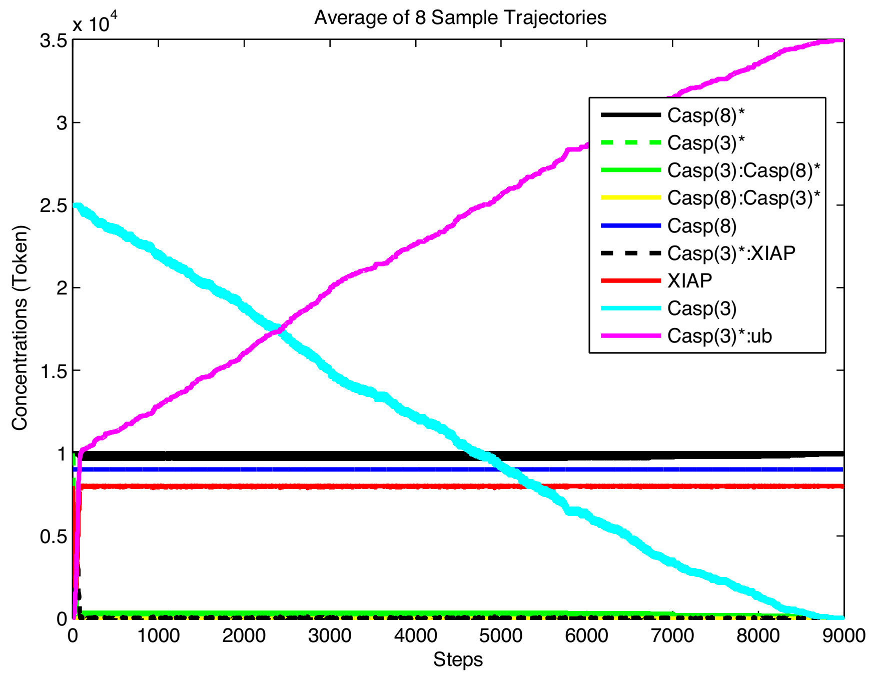

FIGURE 4.16: Average Concentrations Calculated with 8 Parallel Simulations ($\gamma = 0.003$).

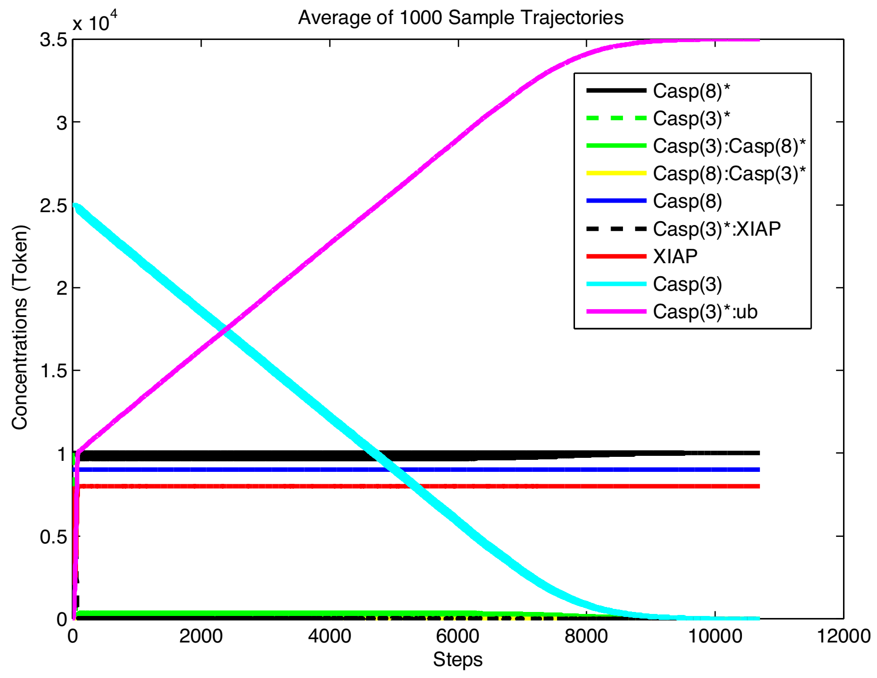

FIGURE 4.17: Average Concentrations Calculated with 1000 Parallel Simulations ($\gamma = 0.003$).

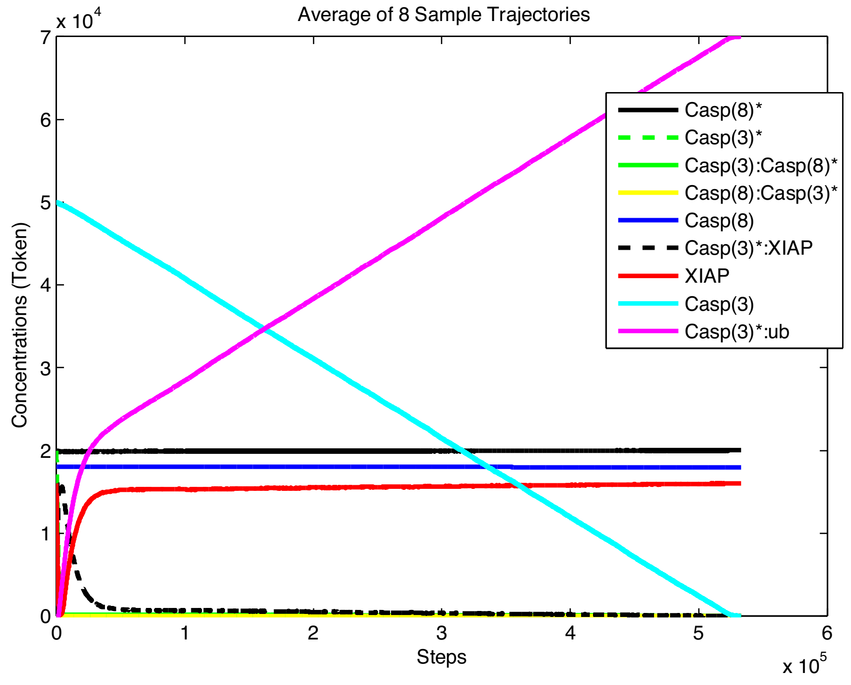

FIGURE 4.18: Average Concentrations Calculated with 8 Parallel Simulations ($\gamma = 0.1$).

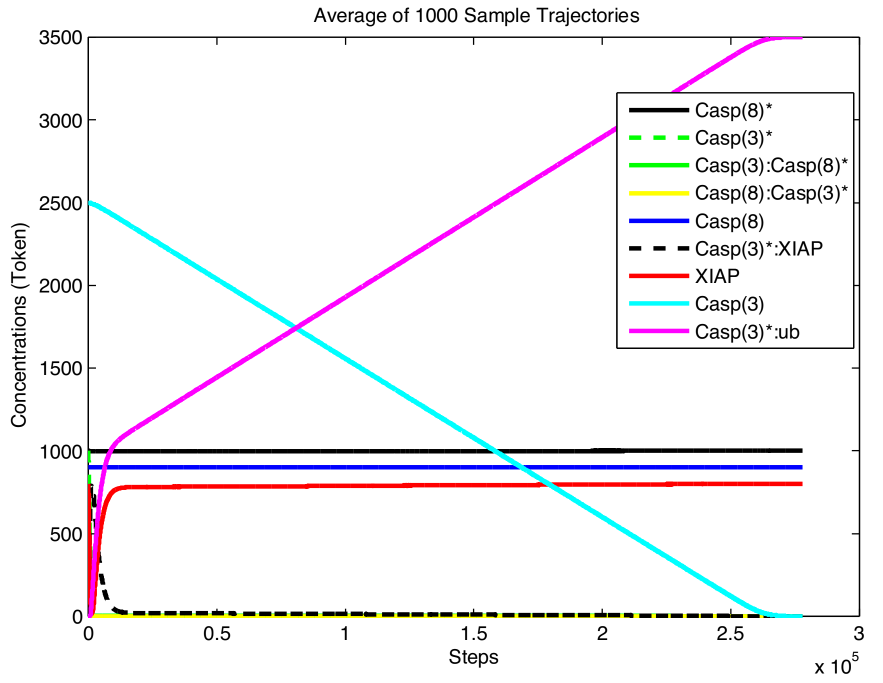

FIGURE 4.19: Average Concentrations Calculated with 1000 Parallel Simulations ($\gamma = 0.1$).

TABLE 4.6: RMSE (Tokens) for Different Scaling Factors and Parallel Simulations

Scaling Factor (γ) $\rightarrow$	0.003	0.003	0.003
No. of Parallel Simulations$\rightarrow$	1	8	1000
Species $\downarrow$	Fig 4.10	Fig 4.16	Fig 4.17
Casp8*	4.0313	3.1774	3.1284
Casp3:Casp8*	4.1694	3.3507	3.2891
Casp8:Casp3*	0.0235	0.0235	0.0219
Casp8	0.2208	0.2214	0.2127
Casp3*:XIAP	42.3476	41.4997	36.3968
XIAP	42.3476	31.4997	26.3968
Casp3	8.7924	8.0482	8.0776
Casp3*:ub	38.7770	32.6785	31.1401
Mean $\rightarrow$	17.5887	15.0624	13.5829
Scaling Factor (γ) $\rightarrow$	0.1	0.1	0.1
No. of Parallel Simulations$\rightarrow$	1	8	1000
Species $\downarrow$	Fig 4.13	Fig 4.18	Fig 4.19
Casp8*	3.6300	3.1803	2.2468
Casp3:Casp8*	3.7990	3.3352	2.3855
Casp8:Casp3*	0.0227	0.0198	0.0195
Casp8	0.2120	0.1838	0.1742
Casp3*:XIAP	4.7571	2.6256	1.9400
XIAP	4.7571	2.6256	1.9400
Casp3	8.0021	7.3782	6.4239
Casp3*:ub	6.0777	5.0592	4.0004
Mean $\rightarrow$	3.9072	3.0510	2.3913

averaged. But since the expected value of the solution of the Petri net model is equal to the Euler approximation of the ODE solution, when the scaling factor is too small, the Δt_n steps in the Euler approximation will be too large, and the equivalent Euler approximation will not be an accurate approximation for the solution of the ODEs. So, the equivalent Petri net solution will be inacurate as well. For example, for $\gamma = 0.003$ even with averaging 1000 simulations, the phase diagrams get smooth but are still inacurate.

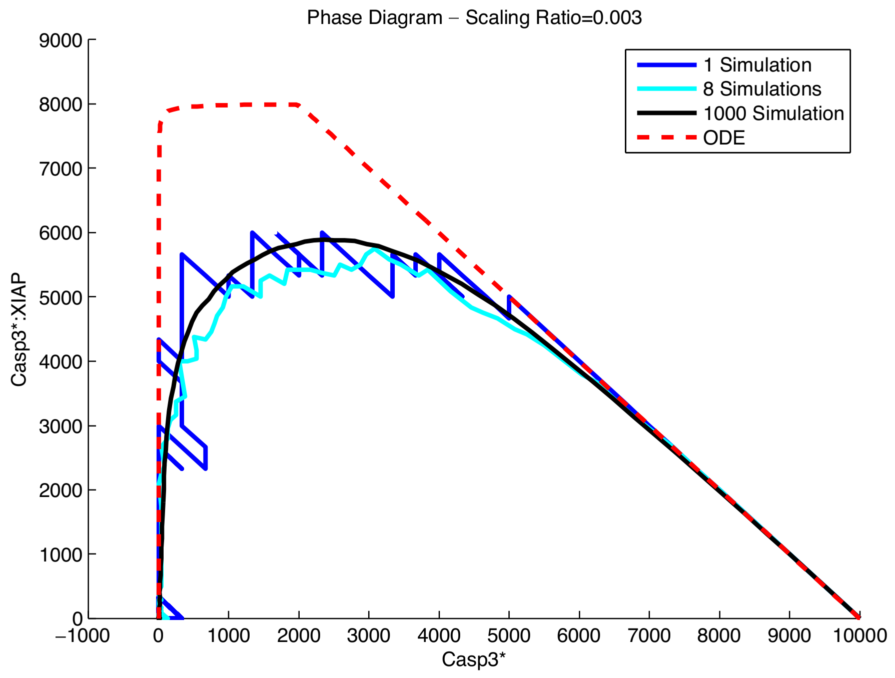

FIGURE 4.20: Phase Diagram of $Casp3^*$ and $Casp3^* : XIAP$ for Petri nets with Averaged Results and Different Scaling Ratios and ODE Simulations.

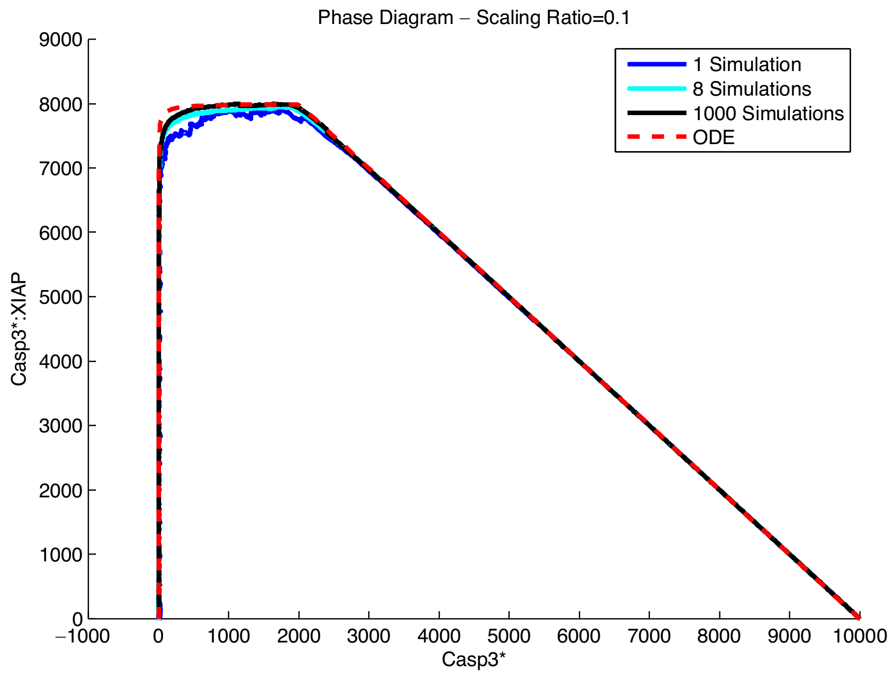

FIGURE 4.21: Phase Diagram of $Casp3^*$ and $Casp3^* : XIAP$ for Petri nets with Averaged Results and Different Scaling Ratios and ODE Simulations.

4.7 Sensitivity Analysis

In this section, the result of sensitivity analysis performed by using ODE simulations from SimBiology and Petri nets are compared. Since the relative magnitude of the sensitivity numbers are important (not the exact values) the accuracy of the trajectories is not of great importance here, and Petri net simulations with relatively lower scaling factors can also be used for performing sensitivity analysis. This way, computation time can be reduced. Figure 4.22 shows the results of sensitivity analysis with ODE simulations obtained from SimBiology. Figures 4.23 and 4.25 show the results of sensitivity analysis performed by Petri net simulations for 2 percent perturbation (in initial condition) and scaling factor equal to $\gamma = 0.1$ and 1 percent perturbation (in initial condition) and scaling factor equal to $\gamma = 1$, respectively.

For large values of γ ($\gamma = 0.1$ or $\gamma = 1$), the trajectory does not change significantly in two different runs. So, we can use the results of a single run to calculate the sensitivities. For smaller values of γ, calculation of the expected value using several runs is necessary for performing sensitivity analysis.

Note that the values do not necessarily need to be similar, as long as the order is the same, since the scale of the simulations are different. Figures 4.24 and 4.26 show the absolute value of the change in the output for perturbation in different inputs. As we can see in the results, perturbing XIAP has the highest effect on the output, i.e. the output is most sensitive to XIAP.

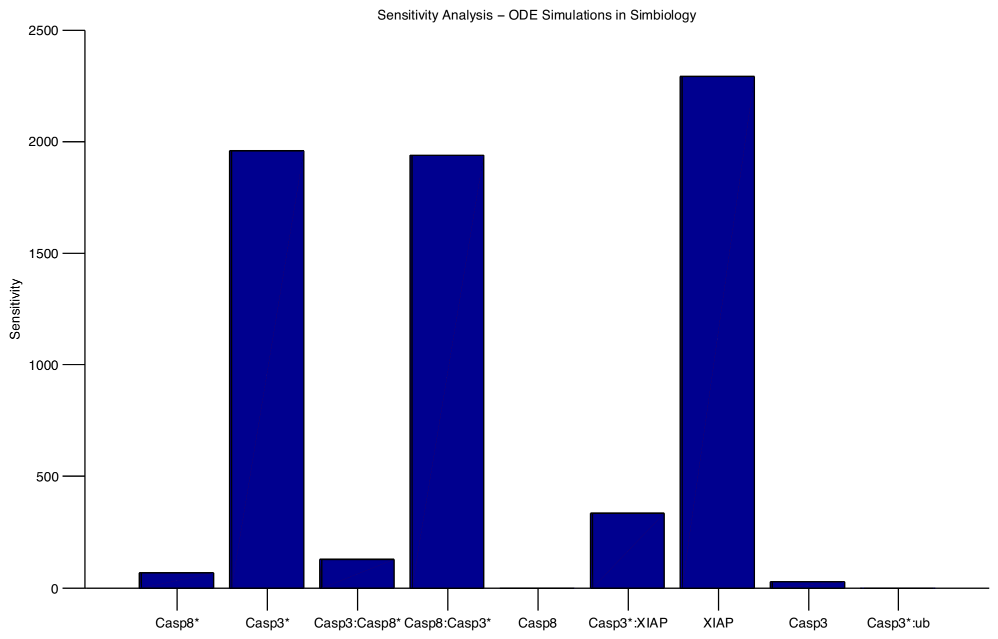

FIGURE 4.22: Sensitivity Analysis Results Obtained from ODE Simulations .

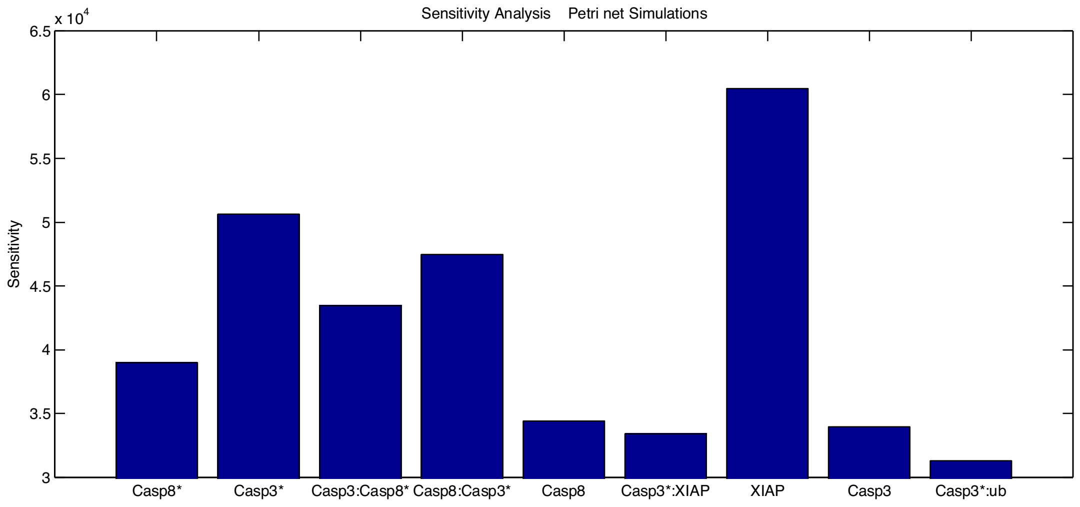

FIGURE 4.23: Sensitivity Analysis Results Obtained from Petri net Simulations with 2% Perturbation in Inputs ($\gamma = 0.1$).

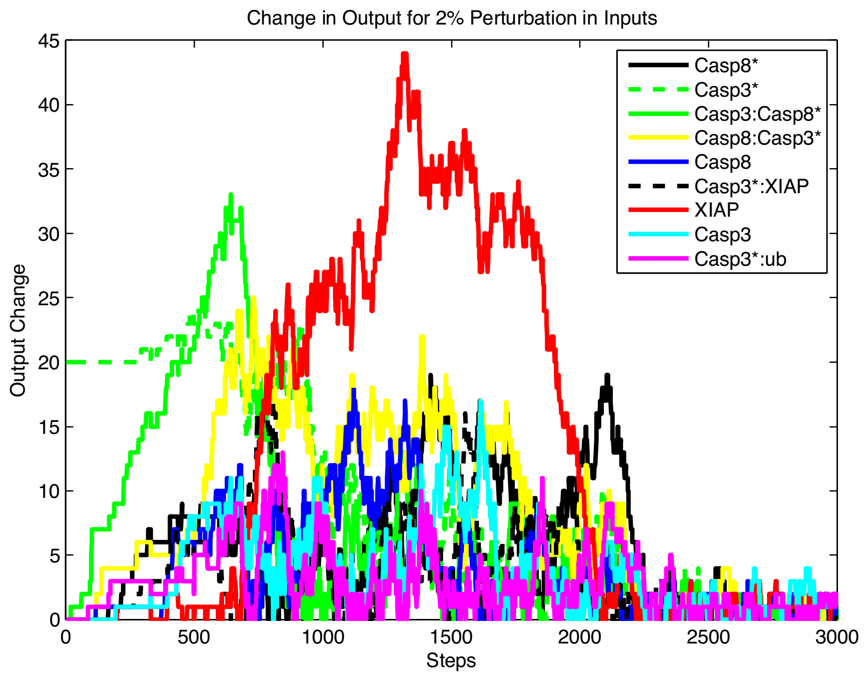

FIGURE 4.24: Change in Output for 2% Perturbation in Input ($\gamma = 0.1$).

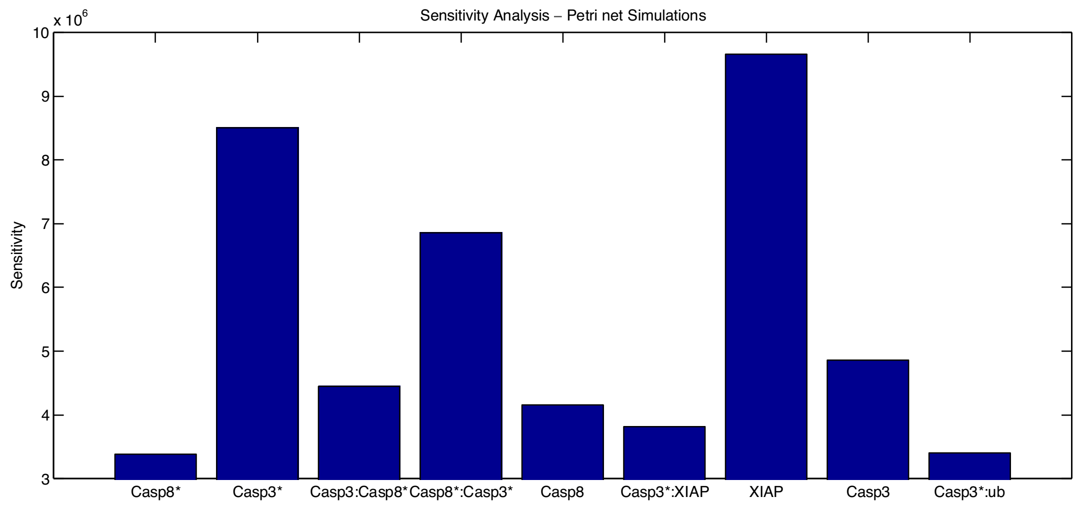

FIGURE 4.25: Sensitivity Analysis Results Obtained from Petri net Simulations with 1% Perturbation in Inputs ($\gamma = 1$).

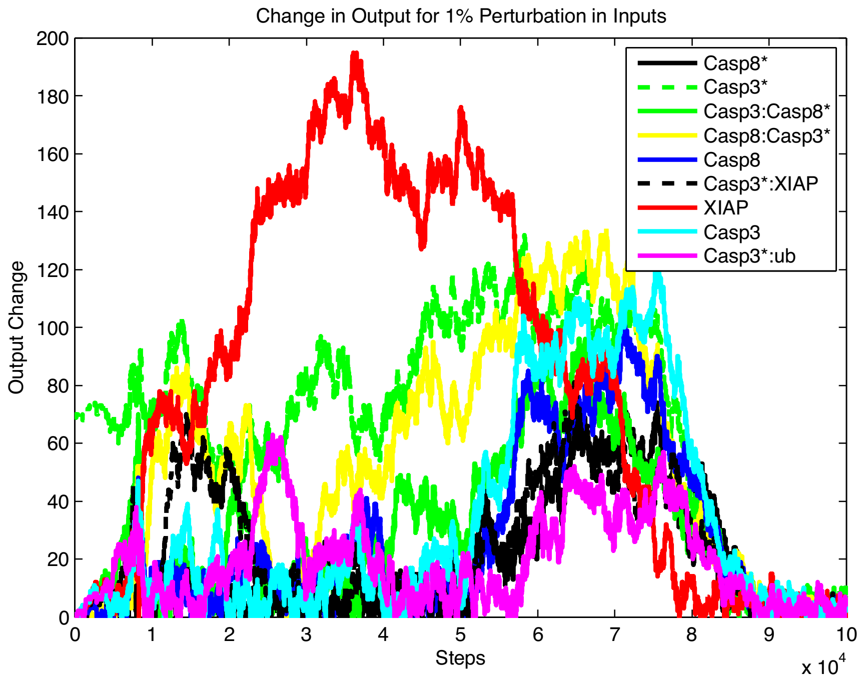

FIGURE 4.26: Change in Output for 1% Perturbation in Input ($\gamma = 1$).

As can be seen in the figures, $Casp3^* : ub$ has the lowest sensitivity value in all results. If we look at the Apoptosis network in Figure 4.2, we can see that $Casp3^* : ub$ is the place in which all the tokens are accomulated and is only an output place. So, perturbing its value has no effect on the rest of the network. Hence, the output of the system ($Casp3^*$) is not sensitive to $Casp3^* : ub$.

4.8 Conclusion

In this chapter, a special type of stochastic Petri nets were used to model the behaviour of the proteins in a simple Apoptosis network. The results of the simulations were compared with the results of ODE simulations, which are one of the most common methods used in analyzing protein networks. The same method was then used to perform sensitivity analysis, and the results were compared with the results of the sensitivity analysis performed using ODE models. As was shown in the results, Petri nets are powerful tools that can be used to model protein networks with desirable accuracy and due to their discrete nature, they can reduce simulation time.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

Discrete event models have been used in modeling biological systems. In this thesis, a type of stochastic Petri nets was proposed to model chemical reactions that are represented with a set of ordinary differential equations. It was shown that the Petri net model calculated correctly the changes in chemical concentrations. It should be noted that in the output sequence generated by the Petri net, there was no linear relationship between the sequence index n and time t. This meant that while the Petri net correctly calculates the concentrations, the timing information is not present in the output of the Petri net. This is equivalent to a distortion (nonlinear scaling) of the time axis. The main advantage of the proposed method however is that it can be easily implemented on a computing system with parallel processors, and hence has lower computational cost. The model was then used to perform sensitivity analysis.

As our case study, a biological system was modeled and used to demonstrate the performance of the proposed method. The same system was used to perform sensitivity analysis using the stochastic Petri net model to determine the best target for designing a drug control mechanism.

The results of the simulations were then compared with the ODE based simulation results obtained from the ODE model in MATLAB's SimBiology toolbox. The simulation results were the same (apart from the fact that the Petri net steps and ODE time values are different in nature). The sensitivity analysis results of the two methods were similar.

5.2 Future Research

This area of research is very new and a lot of interesting topics can be studied based on the research line of this thesis.

Other analyses methods such as stability analysis and reachability analysis can be performed using the proposed method.

One of the disadvantages of ODEs for modeling chemical reaction systems is that ODEs can only model the behaviour of the reactions deterministically if certain assumptions such as well-mixing of the chemicals, constant external conditions, large number of reacting molecules are satisfied [5]. Regular ODE models cannot be adjusted to take into account the changes in the behaviour of the system if such assumptions are not satisfied. Due to the stochastic nature of our proposed model, we can define new probability functions to consider the changes in the reaction rates (which are equivalent to the rates of transition firings in the equivalent Petri net model) if such assumptions do not hold.

Another possible line of research is to focus on sensitivity analysis for systems with multiple inputs and outputs. Sometimes, because of the internal interactions of the model, the perturbation of two or more inputs simultaneously causes variation in the output greater than that of varying each of the inputs alone. In such cases, the sensitivity analysis method should consider such inputs together. Since the Petri net model is relatively fast, it can be used to perform more complicated and computationally expensive simulations to calculate these sensitivities.

Bibliography

[1] A. Samant. Getting started with SimBiology: An online tutorial. *The MathWorks*, page http://www.mathworks.com/company/events/webinars, 2008.

[2] H. De Jong, J. Gouzé, C. Hernandez, M. Page, T. Sari, and J. Geiselmann. Qualitative simulation of genetic regulatory networks using piecewise-linear models. *Bulletin of Mathematical Biology*, 66(2):301 – 340, 2004.

[3] D. Gilbert and M. Heiner. From Petri nets to differential equations, an integrative approach for biochemical network analysis. *Petri Nets and Other Models of Concurrency-ICATPN 2006*, pages 181 – 200, 2006.

[4] R. Schlatter, H. Conzelmann, E. Gilles, O. Sawodny, , and T. Sauter. Analysis of an apoptotic core model focused on experimental design using artificial data. *Systems Biology*, 3(4):255 – 265, 2009.

[5] M. Johnston and D. Siegel. Linear conjugacy of chemical reaction networks. *Journal of mathematical chemistry*, 49(7):1263 – 1282, 2011.

[6] N. El-Farra, A. Gani, and D. Christofides. A switched systems approach for the analysis and control of mode transitions in biological networks. *Proc. of American Contr. Conf.*, 5:3247 – 3252, Jun. 2005.

[7] A. A. Julius, A. Halasz, V. Kumar, , and G. J. Pappas. Finite state abstraction of a stochastic model of the lactose regulation system of escherichia coli. *in Proc. of 45th IEEE Conf. on Decision and Control*, pages 19 – 24, Dec. 2006.

[8] S. Ware and R. Malik. Conflict-preserving abstraction of discrete event systems using annotated automata. *Discrete Event Dynamic Systems*, pages 1 – 27, 2012.

[9] D. Y. Lee, R. Zimmer, S. Y. Lee, and S. Park. Colored Petri net modeling and simulation of signal transduction pathways. *Discrete Event Dynamic Systems*, 8(2):112 – 122, 2006.

[10] H. Genrich, R. Kuffner, and K. Voss. Executable Petri net models for the analysis of metabolic pathways. *International Journal on Software Tools for Technology Transfer (STTT)*, 3(4):394 – 404, Sep. 2001.

[11] M. Chen and R. Hofestadt. A medical bioinformatics approach for metabolic disorders: biomedical data prediction, modeling, and systematic analysis. *Journal of Biomedical Informatics*, 39(2):147 – 159, 2006.

[12] R. Ghosh and C. Tomlin. Symbolic reachable set computation of piecewise affine hybrid automata and its application to biological modelling: Delta-notch protein signalling. *IEEE Proceedings on Systems Biology*, 1(1):170 – 183, Jun. 2004.

[13] A. Halasz, V. Kumar, M. Imielinski, C. Belta, O. Sokolsky, S. Pathak, and H. Rubin. Analysis of lactose metabolism in e.coli using reachability analysis of hybrid systems. *IET Systems Biology*, 1(2):130 – 148, 2007.

[14] M. Herajy and M. Schwarick. A hybrid Petri net model of the eukaryotic cell cycle. *3rd International Workshop on Biological Processes and Petri Nets, Hamburg, Germany*, pages 29 – 43, 2012.

[15] S. Proß, S. J. Janowski, B. Bachmann, C. Kaltschmidt, and B. Kaltschmidt. Pnlib-a modelica library for simulation of biological systems based on extended hybrid Petri nets. *3rd International Workshop on Biological Processes and Petri Nets*, pages 47 – 61, 2012.

[16] T. Ideker, L. R. Winslow, and A. D. Lauffenburger. Bioengineering and systems biology. *Annuals of Biomedical Engineering*, 34(2):257 – 264, 2006.

[17] H. Kitano. Systems biology: A brief overview. *Science*, 295(5560):1662–1664, Mar 2002.

[18] H. Kitano. Foundations of systems biology. *MIT Press*, Mar 2001.

[19] M. A. Savageau and R. Rosen. Biochemical systems analysis: a study of function and design in molecular biology. *Addison-Wesley Reading*, 725, Mar 1976.

[20] S. Ilie, W. H. Enright, and K. R. Jackson. Numerical solution of stochastic models of biochemical kinetics. *Canadian Applied Mathematics Quarterly*, 17 (3):523–554, 2009.

[21] A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, and S. Tarantola. Global sensitivity analysis. *The Primer, John Wiley and Sons*, 2008.

[22] N. Motee, B. Bamieh, and M. Khammash. Stability analysis of a class of biological network models. *American Control Conference (ACC)*, pages 5936–5941, 2010.

[23] W. Materi and D. S. Wishart. Computational systems biology in cancer: Modeling methods and applications. *Gene Regulation System Biology*, (1):91 – 110, Sep 2007.

[24] P. Amodio and L. Brugnano. Parallel solution in time of odes: some achievements and perspectives. *Applied Numerical Mathematics*, 10:–, 2008.

[25] K. R. Jackson. A survey of parallel numerical methods for initial value problems for ordinary differential equations. *IEEE Transactions on Magnetics*, 27 (5):3792 – 3797, 1991.

[26] H. Kacser and J. A. Burns. The control of flux. *Symp. Soc. Exp. Biol.*, 27:65 – 104, 1973.

[27] D. A. Fell. Metabolic control analysis-a survey of its theoritical and experimental development. *Biochem. J.*, 286:423 – 444, 1992.

[28] R. Heinrich and S. Schuster. The regulation of cellular systems. *Chapman and Hall, New York*, 1996.

[29] L. Acerenza, H. M. Sauro, and H. Kacser. Control analysis of time-dependent metabolic systems. *J. Theor. Biol.*, 151:423 – 444, 1989.

[30] O. V. Demin, H. V. Westerhoff, and B. N. Kholodenko. Control analysis of stationary fixed oscillations. *J. Phys. Chem*, 103:10695 – 10710, 1999.

[31] R. Heinrich and C. Redder. Metabolic control analysis of relaxation processes. *J. Theor. Biol.*, 151:343 – 350, 1991.

[32] B. N. Kholodenko, O. V. Demin, and H. V. Westerhoff. Control analysis of periodic phenomena in biological systems. *J. Phys. Chem.*, B 101:2070 – 2081, 1997.

[33] M. C. Kohn, L. M. Whitley, and D. Garfinkel. Instantaneous flux control analysis for biochemical systems. *J. Theor. Biol.*, 76:437 – 452, 1979.

[34] K. A. Reijenga, H. V. Westerhoff, B. N. Kholodenko, and J. L. Snoep. Control analysis for autonomously oscillating biochemical networks. *Biophys. J.*, 82: 99 – 108, 2002.

[35] S. Zahirazami, M. Dadar S. Hashtrudi-Zad, and A. G. Aghdam. Sensitivity analysis in Petri net representation of biological systems. *In American Control Conference*, June 2013.

[36] B. P. Ingalls and H. M. Sauro. Sensitivity analysis of stoichiometric networks: An extension of metabolic control analysis to non-steady state trajectories. *Journal of Theoretical Biology*, 222:23 – 26, 2002.

[37] Z. Zi. Sensitivity analysis approaches applied to systems biology models. *IET Systems Biology*, 5(6):336 – 346, 2011.

[38] J. C. Helton, J. D. Johnson, C. J. Salaberry, and C.B. Storlie. Survey of sampling based methods for uncertainty and sensitivity analysis. *Reliability Engineering and System Safety*, 91:1175 – 1209, 2006.

[39] W. Zhu and L. Petzold. Parallel sensitivity analysis for DAEs with many parameters. *Concurrency: Practice and Experience*, 11(10):571–585, 2009.

[40] J. Martins, P. Sturdza, and J. J. Alonso. The complex-step derivative approximation. *ACM Transactions on Mathematical Software (TMOS)*, 29(3): 245 – 262, Sep 2003.

[41] C. A. Petri. Kommunikation mit automaten. *Schriften des IIM Nr. 2, Bonn: Institut für Instrumentelle Mathematik*, PhD thesis, 1962.

[42] M. Peleg, I. Yeh, and R. B. Altman. Modelling biological processes using workflow and Petri net models. *Bioinformatics*, 18(6):825 –837, 2002.

[43] V. N. Reddy, M. N. Liebman, and M. L. Mavrovouniotis. Qualitative analysis of biochemical reaction systems. *Computers in Biology and Medicine*, 26:9 – 24, Jan 1996.

[44] V. N. Reddy, M. L. Mavrovouniotis, and M. N. Liebman. Petri net representations in metabolic pathways. *Intelligent Systems for Molecular Biology*, 93: 328 – 336, 1993.

[45] M. Heiner, I. Koch, and K. Voss. Analysis and simulation of steady states in metabolic pathways with Petri nets. *Third Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools*, 1:15 – 34, 2001.

[46] J. M. Colom, E. Teruel, and M. Silva. Logical properties of p/t systems and their analysis. *MATCH Summer School (Spain)*, 1998.

[47] P. Baldan, N. Cocco, and M. Simeoni. Comparison of metabolic pathways by considering potential fluxes. *3rd International Workshop on Biological Processes and Petri Nets, Hamburg, Germany*, pages 2 – 17, 2012.

[48] F. Corderoa, A. Horvatha, D. Maninia, L. Napionec, M. D. Pierroa, S. Pavanc, A. Piccoe, A. Veglioc, M. Serenoa, F. Bussolinoc, and G. Balboa. Simplification of a complex signal transduction model using invariants and flow equivalent servers. *Journal of Theoretical Computer Science*, 43(412):6036 – 6057, 2011.

[49] I. Low, I. W. Jin, Y. Yang, and H. Lin. Validation of Petri net apoptosis models using p-invariant analysis. *IEEE International Conference on Control and Automation*, pages 416 – 421, Dec 2009.

[50] H. Genrich, R. Kiffner, and K. Voss. Executable Petri net models for the analysis of metabolic pathways. *International Journal on Software Tools for Technology Transfer (STTT)*, 3(4):394 – 404, 2001.

[51] S. Zahirazami, S. Hashtrudi-Zad, and A. G. Aghdam. Model reduction and consistency in discrete event representation of biological systems. *In American Control Conference*, pages 1703 – 1708, June 2008.

[52] P. Baladin, N. Cocco, A. Martin, and M. Simeoni. Petri nets for modelling metabolic pathways: a survey. *Natural Computing*, 9(4):955–989, 2010.

[53] C. Rohr. Simulative model checking of steady state and time-unbounded temporal operators. *3rd International Workshop on Biological Processes and Petri Nets*, pages 62 – 75, 2012.

[54] D. Thorsley. Diagnosability of stochastic chemical kinetic systems: a discrete event systems approach. *American Control Conference*, pages 2623 – 2630, June 2010.

[55] K. McGarry, M. Loutfi, and A. Moscardini. Stochastic simulation of the regulatory pathways involved in diabetes using Petri-nets. *Proc. of the International Conference on Computer Theory and Applications*, 2007.

[56] G. Ciardo. Toward a definition of modeling power for stochastic Petri net models. *Proc. IEEE Int. Workshop on Petri nets and Performance Models*, pages 54 – 62, 1987.

[57] A. Puliafito, M. Telek, and K. S. Triverdi. The evolution of stochastic Petri nets. 1997.

[58] M.K. Molloy. Performance analysis using stochastic Petri nets. *IEEE Transaction on Computers*, 31(9):913 – 917, Sep 1982.

[59] M. Ajmone Marsan, G. Balbo, A. Bobbio, G. Chiola, G. Conte, and A. Cumani. The effect of the execution policies on the semantics and analysis of stochastic Petri nets. *IEEE Transaction on Sofware Engineering*, 15(7): 832 – 846, 1989.

[60] B. Alberts, A. Johnson, J. Lewis, M. Raff, and K. Roberts. Molecular biology of the cell 4th edition. *National Center for Biotechnology Informations Bookshelf*, 2002.

[61] E. D. Sontag. Molecular systems biology and control. *European journal of control*, 11(4):396 – 345, 2005.

[62] A. H. Wyllie. Apoptosis: An overview. *British Medical Bulletin*, 53(3):451 – 465, 1997.

[63] T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541 – 580, 1989.

[64] F. Horn and R. Jackson. General mass action kinetics. *Arch. Rational Mech*, 47:81 – 116, 1972.

[65] D. J. Pannell. Sensitivity analysis of normative economic models: Theoretical framework and practical strategies. *Agricultural Economics*, 16:139 – 152, 1997.

[66] A. Saltelli and P. Annoni. How to avoid a perfunctory sensitivity analysis. *Environmental Modeling and Software*, 25:1508 – 1517, 2010.

[67] D. F. Griffiths and D. J. Higham. Numerical methods for ordinary differential equations initial value problems. *Springer*, 2010.

[68] E. Cerutti, M. F. Campagnoli, M. Ferretti, E. Garelli, N. Crescenzio, A. Rosolen, A. Chiocchetti, M. J. Lenardo, U. Ramenghi, and U. Dianzani. Co-inherited mutations of fas and caspase-10 in development of the autoimmune lymphoproliferative syndrome. *BMC Immunology*, 8(1):8–28, 2007.

[69] M. Hubeau, F. Ngadjeua, A. Puel, L. Israel, J. Feinberg, M. Chrabieh, K. Belani, C. Bodemer, I. Fabre A. Plebani, S. Boisson-Dupuis, C. Picard, A. Fischer, A. Israel, L. Abel, M. Veron, J. L. Casanova, F. Agou, and J. Bustamante. New mechanism of x-linked anhidrotic ectodermal dysplasia with

immunodeficiency: impairment of ubiquitin binding despite normal folding of NEMO protein. *Blood*, 118(4):926 – 935, 2011.

[70] R. W. G. Watson and J. M. Fitzpatrick. Targeting apoptosis in prostate cancer: focus on caspases and inhibitors of apoptosis proteins. *British Journal of Urology International*, 96:30–34, Dec 2005.

[71] M. Lamkanfi, N. Festjens, W. Declercq, T. Vanden Berghe, and P. Vandenabeele. Caspases in cell survival, proliferation and differentiation. *Cell Death and Differentiation*, 14(1):44 – 55, Oct 2006.

[72] N. N. Danial and S. J. Korsmeyer. Cell death: Critical control points. *Cell*, 116(2):205–219, Jan 2004.

[73] A. D. Schimmer, S. Dalili, R. A. Batey, , and S. J. Riedl. Targeting XIAP for the treatment of malignancy. *Cell Death and Differentiation*, 13(2):179–188, Dec 2005.

[74] P. Liston, N. Roy, K. Tamai, C. Lefebvre, S. Baird, G. Cherton-Horvat, R. Farahani, M. McLean, J. E. Ikeda, A. MacKenzie, and R. G. Korneluk. Suppression of apoptosis in mammalian cells by NAIP and a related family of IAP genes. *Nature*, 379(6563):349–353, Jan 1996.

[75] C. S. Duckett, V. E. Nava, R. W. Gedrich, R. J. Clem, J. L. Van Dongen, M. C. Gilfillan, H. Shiels, J. M. Hardwick, and C. B. Thompson. A conserved family of cellular genes related to the baculovirus IAP gene and encoding apoptosis inhibitors. *The EBMO Journal*, 15(11):2685–2694, Jun 1996.

[76] M. Holcik and R. G. Korneluk. Functional characterization of the X-linked inhibitor of apoptosis (XIAP) internal ribosome entry site element: Role of la autoantigen in XIAP translation. *Molecular and Cellular Biolgy*, 20(13): 4648–4657, July 2000.

[77] Q. L. Deveraux and J. C. Reed. IAP family proteins-suppressors of apoptosis. *Genes and Development*, 13(3):239–252, Feb 1999.

[78] C. S. Duckett, F. Li, Y. Wang, K. J. Tomaselli, C. B. Thompson, and R. C. Armstrong. Human IAP-like protein regulates programmed cell death downstream of bcl-xl and cytochrome c. *Molecular and Cellular Biolgy*, 18(1): 608–615, Jan 1998.

[79] D. Green. Means to an end: Apoptosis and other cell death mechanisms. *Cold Spring Harbor, NY: Cold Spring Harbor Laboratory Press*, 2011.