2018 届研究生硕士学位论文

分类号：_____ 学校代码：10269

密　级：_____ 学　　号：51164500258

華東師範大學

East China Normal University

硕士学位论文

MASTER'S　DISSERTATION

基于改良 SMC 算法以及神经网络的带参数马尔科夫模型验证方法研究

院　　系：计算机科学与软件工程学院

专　　业：软件工程

研究方向：高可信计算理论与技术

指导教师：张敏　副教授

学位申请人：严佳

2018 年 12 月

2018 Master's Degree Thesis **University code: 10269**

Student ID: 51151500126

East China Normal University

Antithetic Path Based SMC and BP Neural Network Method for Parameterized Discrete Time Markov Chain

Department: School of Computer Science and Software Engineering

Major: Software Engineering

Direction: Trustworthy Computing Theory and Technology

Tutor: Assoc Prof. Min Zhang

Candidate: Yan Jia

2018.12

华东师范大学学位论文原创性声明

郑重声明：本人呈交的学位论文《基于改良 SMC 算法以及神经网络的带参数马尔科夫模型验证方法研究》，是在华东师范大学攻读硕士/博士（请勾选）学位期间，在导师的指导下进行的研究工作及取得的研究成果。除文中已经注明引用的内容外，本论文不包含其他个人已经发表或撰写过的研究成果。对本文的研究做出重要贡献的个人和集体，均已在文中作了明确说明并表示谢意。

作者签名：_____ **日期：** 年 月 日

华东师范大学学位论文著作权使用声明

《高中生英语在线自主学习现状的调查研究》系本人在华东师范大学攻读学位期间在导师指导下完成的硕士/博士（请勾选）学位论文，本论文的著作权归本人所有。本人同意华东师范大学根据相关规定保留和使用此学位论文，并向主管部门和学校指定的相关机构送交学位论文的印刷版和电子版；允许学位论文进入华东师范大学图书馆及数据库被查阅、借阅；同意学校将学位论文加入全国博士、硕士学位论文共建单位数据库进行检索，将学位论文的标题和摘要汇编出版，采用影印、缩印或者其它方式合理复制学位论文。

本学位论文属于（请勾选）

（ ）1.经华东师范大学相关部门审查核定的“内部”或“涉密”学位论文*，于 年 月 日解密，解密后适用上述授权。

（√）2.不保密，适用上述授权。

导师签名_____ 本人签名_____

年 月 日

* “涉密”学位论文应是已经华东师范大学学位评定委员会办公室或保密委员会审定过的学位论文（需附获批的《华东师范大学研究生申请学位论文“涉密”审批表》方为有效），未经上述部门审定的学位论文均为公开学位论文。此声明栏不填写的，默认为公开学位论文，均适用上述授权）。

严佳硕士学位论文答辩委员会成员名单

姓名	职称	单位	备注

摘要

模型检测方法是一种形式化方法，可以检验系统是否满足某种性质。它借助于对系统建立起的数学模型和依赖时序逻辑公式描述的性质对系统运行检测算法，并给出系统是否满足该性质的结论。作为一种能够对系统进行精确检测的方法，已经被广泛运用于汽车电子，轨道交通以及航天航空等领域。虽然模型检测方法已经在实际的应用中取得了显著的效果，但是挑战依旧存在。在实际的建模过程中，由于系统中某些信息的不确定性，导致所建立的模型包含非确定参数。由于传统的模型检测方法不允许模型中包含不确定参数，所以无法用传统模型检测方法对该种模型进行验证。

为了对这类具有不确定信息的模型进行建模与验证，本文提出了一种基于BP 神经网络（back propagation neural network）的函数拟合的方法对模型在不确定性参数下的模型检测概率值的函数进行拟合。首先，该方法需要获取模型在若干参数下的模型检测值作为训练数据对神经网络进行训练，在训练的过程中为不同的训练样本赋予不同的可靠性权重，从而使得神经网络的训练更加依赖于那些可靠性高的样本，从而使得神经网络拟合和预测的精度得以提高。

为了提高模型在确定参数下的模型检测的效率，本文提出一种基于方差减小技术的统计模型检测（statistical model checking, SMC）算法。该算法借鉴蒙特卡罗方法中的对偶变量（antithetic variable）技巧，提出了对偶状态和对偶路径的概念，并通过一种基于偏序关系的状态重排序算法对模型中的状态进行重排序，使得满足给定时态逻辑公式的路径在所在状态空间的同一测。基于上述两点抽样产生的对偶路径样本能够使得 SMC 算法的估计值的误差得到显著的减小，从而提高 SMC 算法的准确度。

本文用满足函数来表示从模型的不确定参数空间到模型在此参数下的模型检测值的之间的映射。本文证明了上述定义的满足在一定条件是连续函数。由于无法给出满足函数的具体的解析形式，本文提出使用 BP 神经网络对满足函数进行拟合。为了提高 BP 神经网络的拟合精度，本文提出为不同的样本赋予不同的可靠性权重以表示样本的可靠性，通过为高可靠的样本赋予更高的权重，使得神经网络的训练更多地依赖可靠性高的样本，使得训练和预测的精度得到提高。

SpacNN（Statistical Probabilistic Approximate Model Checker with Neural Network）是基于上述算法实现的统计模型检测工具。通过对网络协议和分布式算法等案例进行分析，发现改进的 SMC 算法可以显著提升模型在确定参数下的估计值的精度，从而间接减小了 SMC 所需的样本大小。实验也发现，对于非确定模型的验证，加权的神经网络拟合方法能够减小神经网络预测未知参数下的模型检测值的平均偏差。

关键词：统计模型检测；带参数马尔科夫链；对偶变量；BP 神经网络；

Abstract

As a formal method, model checking could check whether a system satisfy some kind of property. It use a mathematical method to build a model that representing the system and some temporal logic formula to represent the property and give an answer to whether the system satisfy the property. As a accurate checking method, it is widely used in automobile industry, transportation and space industry. In spite of its wide usage, challenges exist. In many practical practices, many unsure information can not be obtained, thus the model contain unsure parameters. Because the classical model checking method does not allow such a uncertainty, thus new methods need to be proposed.

To deal with such uncertainty situation, we propose a original curve fitting method based on back propagation neural network to predict model's satisfactory probability under unsure parameters. Firstly, we obtain the satisfactory probability under some certain parameters and use those data as training data to feed the network. Through weighing different training data according on their reliability, we make the training process depends more on the data that is more reliable, thus improving the accuracy of the prediction.

To improve the efficiency of model checking under certain parameters, we introduce a antithetic sampling based statistical model checking method. We borrows the idea of antithetic variable in Monte Carlo method and introduce the definition of antithetic state and antithetic path. For antithetic path to work, we introduce a state reordering process to make all satisfying path stay in one side of the path space. After the reordering process, sample of antithetic path can be obtained with smaller variances and thus more accurate prediction.

Also we introduce the definition of satisfactory function to represent the relation from parameters space to probability. We prove the satisfactory function is a continuous function and thus a curve fitting method can be used. We use the BP neural network to do the fitting and introduce the notation of reliablity weight as a measure of sampling's reliablity. By doing this, the accuracy of training and prediction of the neural network can be improved.

SpacNN(Statistical Probabilistic Approximate Model Checker with Neural Network) is a tool implemented using method above. Through using the tool to check two cases from communication protocols and distributive algorithms, we show that the antithetic path based SMC algorithm could improve the accuracy of the prediction of SMC algorithm and network.

keywords: statistical model checking, parameterized discrete time Markov chain, antithetic variable, back propagation neural network

目　录

插图清单

附表清单

第一章 绪论

1.1 研究背景

近年来，随着计算机理论的长足进步以及计算机硬件水平的不断提高，计算机的理论成果在越来越多的领域内得到了运用。在此之中，模型检测由于其成熟的理论基础和工具集得到了来自各个领域的广泛关注。在许多领域内都可以见到模型检验的应用实例，例如在生物领域，[1]运用模型验证技术研究机体感染病毒后的感染过程，又例如，在网络安全领域，模型验证又被用于研究蓝牙通信协议[2]的安全性与可靠性。

常规的模型检验技术倚赖于对系统进行正确的建模。在模型检验开始前，研究者会仔细分析实际系统的运行，通过分析系统所处的所有可能的状态以及状态之间的转移关系对系统建立正确的模型。根据研究问题的不同，可采用不同的模型对系统进行建模。得到了抽象了系统状态和行为的模型之后，我们可以利用逻辑公式对系统所需要满足的性质进行描述。例如，针对离散时间马尔科夫链模型（discrete-time Markov chain），我们可以利用概率计算树逻辑公式[3]（probabilistic computational tree logic）描述系统所满足的性质。得到代表系统的模型以及描述系统性质的时态逻辑公式后，可以通过运行模型检验算法验证系统是否满足相应公式。

概率模型检验技术（probabilistic model checking）[4]是模型检验技术中的一个分支。它主要用于分析系统满足某个性质的概率。使用概率模型检验技术可以回答关于系统的一些定量问题：例如“在一个包含不可靠信道的网络内，在 10 个单位时间内，数据包能从主机 A 到达主机 B 的概率为多少？”。概率模型检测技术最早由[5]提出，成熟的工具有 PRISM 等。经典的概率模型检验算法复杂度依赖于模型的复杂度，所以对于一个复杂度极高的模型，经典的概率模型检验算法存在状态爆炸的问题[6]。

统计模型检验[7][8]（statistical model checking, SMC）是概率模型检验方法

中的一种。它本质上是一种蒙特卡罗方法[9][10]（Monte Carlo Method）。在得到了用于描述系统的模型以及用于描述性质的时态逻辑公式后，该方法从模型中获取若干条有限长随机路径，并统计满足特定时态逻辑公式的路径所占的比例。通过将单条有限长随机路径的验证看成是完成了一次伯努利试验，我们可以将定量求解模型满足时态逻辑公式的概率的问题转化成是一个二项分布的参数估计问题。因为 SMC 算法的时间复杂度不依赖于模型的复杂度，所以经常利用 SMC 算法验证一些复杂度较高的模型，例如网络协议[11][12]，网络安全问题[13][14][15]以及生物系统[16][17][18]等。

但是无论是传统的概率模型检验方法还是统计模型检验算法，其可以正常工作的前提都是系统的模型应该是确定的，即系统中不能包含不确定的参数。但是在实际的运用中，除去一些标准化极高的领域例如例如微电子领域内的验证外，在大多数领域内，研究者需要应对的往往是不确定模型。虽然可以利用参数估计方法[19][20]在一定程度上消除模型的不确定性，但是在大多数的领域内不确定性并不能被完全消除。本文的研究正是在如上背景下提出。

1.2 研究动机

本文的主要研究对象是带参数的马尔科夫链模型。如前所述，带参数的马尔科夫链模型因为包含了一定的不确定性，无法使用常规的 SMC 算法以及模型检验算法。可行的一种解决方法就是对模型的参数的空间进行搜索，但是需要在每个参数点下均进行模型检验，效率较差。通过考察模型的满足函数曲线我们发现，当模型在任意两个状态之间的转移概率均是模型参数的连续函数时，模型满足性质的满足函数是一个定义在参数值域上的连续函数。于是我们提出通过对满足函数曲线进行拟合的方法，首先验证模型在给定参数点下的模型检验值，然后通过拟合出一条误差尽量小的曲线，我们可以近似估计模型在未知点下的模型检验值。

为了提高模型在特定参数点下的模型检验值的准确性，本文借鉴了经典蒙特卡罗方法中的误差减小技术，即对偶变量取样法[21]，提出了针对 SMC 算法的对偶路径取样法。根据经典的蒙特卡罗方法我们可知，估计值的方差越小，估计值的准确度也就越高。对偶变量取样法作为一种方差减小方法，可以有效减小估

计值的方差从而提高估计值的准确度。

1.3 相关工作

针对包含不确定性参数的马尔科夫模型，有不少学者提出了参数估计的方法试图在一定程度上降低模型的不确定性。例如，Opper[22]等人研究了马尔科夫跳变过程（Markov Jump Process, MJP）中的参数不确定性，提出了一种拓展自平均属性（mean-field）的方法对系统中路径的分布进行贝叶斯估计，从而在一定程度上削减参数的不确定性。Andreychenko [23]等人提出了一种针对带参数连续时间马尔科夫模型的退出速率（exit rate）以及初始化参数的最大值的估计方法。同样是针对带参数 CTMC 模型，Bortolussi[24]提出了根据观测数据和模型需要满足的时态逻辑公式学习模型中的不确定参数以及初始化变量。Georgoulas[25]等人提出了将模型检测技术与机器学习相结合的方法，通过提出了一种新型概率程序语言 infer.NET Fun[26]对带参数 CTMC 进行推断和验证。

在[27]中，Bortolussi提出了一种基于贝叶斯估计的方法对带参数的连续时间马尔科夫链（continuous-time Markov chain，CTMC）模型进行模型检验。通过给出模型的满足函数的先验分布，根据观测值进行后验估计，从而对模型在参数空间下进行整体性的模型检测 。基于上述算法，Bortolussi等人实现了相应的验证工具smoothedMC。

1.4 文章结构

本文主要考查带参数的离散时间马尔科夫链模型，解决如何在不进行全局搜索的情况下验证模型在参数空间下的模型检验值。本文首先证明了模型的满足函数在特定条件下是其定义域内的连续函数。为了对满足函数进行拟合，首先需要获取模型在特定点下的模型检验值以此作为BP神经网络的训练数据。为了获取神经网络所需的训练数据，提出了一种基于对偶变量取样法的SMC算法，基于此方法可以有效减小SMC算法估计值的准确性，从而减小SMC算法所需有限长随机路径的条数并有效提高算法效率。本章主要包含六个章节，除本章外，依次

介绍了背景知识，方法框架以及具体算法、工具的实现，案例分析以及总结与展望。各章内容包括：

第二章 背景知识

本章主要介绍了用于系统建模的DTMC，CTMC等模型定义，用于描述模型满足性质的概率计算树逻辑（probabilistic computation tree logic，PCTL）以及连续随机逻辑（continuous stochastic logic，CSL）。另外，本章着重介绍了标准的概率模型检测（statistical model checking）方法，便于引入其改进方法。

第三章 基于对偶路径的 SMC 算法以及 BP 神经网络的验证框架

本章主要介绍基于对偶路径取样法改进的SMC算法以及改进的加权BP神经网络的满足函数拟合算法，并给出了之中一些详细算法的介绍，例如状态重排序算法，对偶路径生成算法，加权BP神经网络的训练算法等。最后通过一个简单的例子描述完整算法框架。

第四章 带参数马尔科夫链验证工具SpacNN

SpacNN（Statistical Probabilistic Approximate Checker with Neural Network）是本文根据第三章中的方法框架实现的带参数马尔科夫链模型的验证工具。本章主要阐述SpacNN工具的软件架构，模块组成，算法实现以及验证的流程。

第五章 案例分析

本章主要对概率广播算法进行了案例分析。通过分别使用标准SMC算法以及基于对偶路径的SMC算法验证了对偶路径在降低估计值方差方面的效果，并利用加权BP神经网络对模型的满足函数进行了拟合，拟合效果符合准确度要求。

第六章 总结与展望

本章主要总结了本文研究过程中的贡献与不足，并讨论了当前工作中

具体存在的不足以及未来的改进方向。

第二章 背景知识

2.1 模型与性质的描述

本文主要研究带参数离散时间马尔科夫链模型（parameterized discrete-time Markov chain）的验证问题，首先给出 DTMC 的定义以及相应的 PCTL 定义。

定义 1. 离散时间马尔科夫链模型　离散时间马尔科夫链模型可以用一个四元组表示，即$D = (S, s_{init}, P, L)$，其中S有限状态空间，$s_{init} \in S$表示初始状态集，$P: S \times S \to [0,1]$表示转移概率矩阵，之中的每个元素P_{ij}表示从状态s_i到s_j的转移概率，并且对于任意的状态s_i，恒有$\sum_{s_j \in S} P(s_i, s_j) = 1$成立。$L = S \to 2^{AP}$表示一个标签函数（labelling function），它将一个系统所处的状态映射到一个原子命题集AP的超集中的一个元素。

对于一个 DTMC 模型$D = (S, s_{init}, P, L)$，我们用一条路径表示系统的一次执行过程。 一条路径的形式化定义可以用以下的状态序列来表示：

$$p = s_0 s_1 s_2 \ldots$$

其中$s_i \in S$并且$P(s_i, s_{i+1}) > 0$对于任意的状态s_i。我们定义一条路径中的第i个状态为$p(i)$。一条路径可以是有限长的也可以是无限长的。我们定义系统D的所有的从状态s出发且有限长路径组成的集合为$Path_{fin}^{D}(s)$。有了模型D有限长路径的定义，我们不加证明地给出一条有限长随机路径在原模型D中被取到的概率为

$$P_s(p_{fin}) = \begin{cases} 1 \; if \; n = 0 \\ P\big(p(0), p(1)\big) \ldots P\big(p(n-1), p(n)\big) otherwise \end{cases}$$

其中n是路径p的长度。

我们使用 PCTL 描述 DTMC 模型所满足的性质。PCTL 本质上计算树逻辑的一个概率扩展。

PCTL 的语法定义如下：

$$\Phi ::= true \mid a \mid \neg\Phi \mid \Phi \wedge \Phi \mid P_{\sim p}[\phi]$$

$$\phi ::= X\Phi \mid \Phi \; U^{\leq k} \; \Phi$$

其中，a表示一个原子命题，$\sim \in \{<, \leq, \geq, >\}$，$p \in [0,1]$并且$k \in \mathrm{N} \cup \{\infty\}$。

另外，我们分别称Φ和ϕ为状态公式和路径公式。我们一般使用状态公式Φ来描述系统D所满足的性质。通常一个状态s满足$P_{\sim p}[\phi]$当且仅当在所有从s出发的路径中，满足ϕ的路径所占的概率的形式满足~p。

以下是状态和路径分别满足公式ϕ和Φ的语义。

定义 2. 令$D = (S, s_{init}, P, L)$是一个离散时间马尔科夫链模型。对于任意状态s，我们定义如下的满足关系

$$s \vDash true\ for\ all\ s \in S$$

$$s \vDash a \iff a \in L(s)$$

$$s \vDash \neg\Phi \iff s \nvDash \Phi$$

$$s \vDash \Phi \wedge \Psi \Leftrightarrow s \vDash \Phi \wedge s \vDash \Psi$$

$$s \vDash P_{\sim p}[\phi] \Leftrightarrow Prob^{D}(s, \phi) \sim p$$

其中

$$Prob^{D}(s, \phi) = \mathrm{P_s}\{\mathrm{p} \in \mathrm{Path^D(s)} | \mathrm{p} \vDash \phi\}$$

表示模型中从s出发的所有路径中满足路径公式ϕ的路径的概率值。

对于任意路径$p \in Path^{D}(s)$，我们定义如下的满足关系：

$$p \vDash X\phi \iff p(1) \vDash \Phi$$

$$p \vDash \phi U^{\leq k} \Psi \Leftrightarrow \exists i \in N.(i \leq k \wedge p(i) \vDash \Psi \wedge \forall j < i.(p(j) \vDash \Phi)$$

即路径p满足公式$X\Phi$，当且仅当路径的第二个状态$p(1)$满足状态公式Φ。路径p满足公式$\Phi U^{\leq k}\Psi$，当且仅当在前k个状态中存在一个状态满足公式Ψ，并且其之前的所有其他状态均满足公式Φ。

2.2. 蒙特卡洛方法

蒙特卡罗方法[28][29][30]，又称随机抽样方法，不同于一般数值计算方法，其主要使用频率近似概率的思想对未知参数进行估计。其一般工作流程为，将一个待计算值转化为某个概率模型的参数使得此概率模型的某个统计量正好是待求问题的解。因此，要将一个问题用蒙特卡罗方法求解一般需要三个步骤，即构

造一个概率模型，实现从该概率分布取样，建立统计量对问题求解。

例如，要求解函数$f(x)$在区间$[a,b]$上的定积分，可以对问题进行下列转化：

$$\begin{aligned}\int_a^b f(x)dx &= (b-a)\int_a^b f(x)\cdot\frac{1}{b-a}dx \\ &= (b-a)\int_a^b f(x)U(x)dx \\ &= (b-a)E_f[U(x)]\end{aligned}$$

可以发现，我们首先构造了一个$[a,b]$上的均匀分布，然后在将$f(x)$在$[a,b]$上的定积分转化为统计量$E_f[U(x)]$，最后我们使用随机取样的方法对统计量$E_f[U(x)]$进行估计，即估计$\int_a^b f(x)dx$。

上述方法的思路可以用下式进行概括：即假设存在一个随机变量X，我们需要求X的期望值，即$E(X)$。蒙特卡罗的思路是从X所满足的一个概率空间中进行随机取样，得到n个样本点，然后使用样本均值估计总体期望，即

$$E(x)\approx\bar{X}=\frac{1}{n}\sum_i X_i$$

根据中心极限定理我们可以知道，假设X服从正态分布$N(\mu,\sigma^2)$并且总体方差σ^2未知。假设X_i为独立同分布的随机变量，那么当样本大小足够大时，随机变量$\bar{X}$近似服从于参数为$n-1$的t分布。于是，我们有如下的置信区间：

$$(\bar{X}\pm\frac{S}{\sqrt{n}}t_{\alpha/2}(n-1))$$

其中，S为样本方差，即$S=\sum_i(E\bar{X}-X_i)^2/(n-1)$，$n$为样本大小，$z_{\alpha/2}$为

根据中心极限定理我们有估计值的方差$Var(\bar{X})$与样本方差之间的关系，

$$Var(\bar{X})\approx\frac{\sigma^2}{n}\approx\frac{S^2}{n}$$

因此上述置信区间可以改写为

$$(\bar{X}\pm\sqrt{Var(\bar{X})}t_{\alpha/2}(n-1))$$

从上式可以发现，置信区间的大小仅仅取决于估计值的方差以及置信水平，而与系统复杂度无关。

根据以上讲解，蒙特卡洛方法主要具有以下优点：

1. 对于复杂系统来说算法结构简单。蒙特卡罗方法适用的唯一条件就是能从系统中取样即可，并且估计精度的大小仅仅依赖于抽样次数，这对于解决复杂系统具有优势。

2. 收敛速度不受问题维度的影响。蒙特卡罗方法的收敛指的是随着样本的大小趋近于无穷大，估计值与真实值的误差小于某个阈值的概率将趋近于 1。此收敛不依赖于问题的维度。因此蒙特卡罗方法特别适用于解决高维问题。

不过蒙特卡罗方法也存在一些缺点，例如被估计值较小时，需要较大的取样数进行精确估计。

2.3 统计模型检验算法

统计模型检验算法[31][32]（statistical model checking, SMC）不同于 PRISM 等成熟模型检验工具使用的数值计算方法，它采用蒙特卡罗方法对模型满足某一时态逻辑公式的概率进行估计。SMC 算法需要从系统中获取多条系统执行的路径，并统计这些路径的验证结果，最后通过假设检验的方法对模型满足公式的概率进行推断并得出结论。

假设如下描述骰子模型的 DTMC 模型，

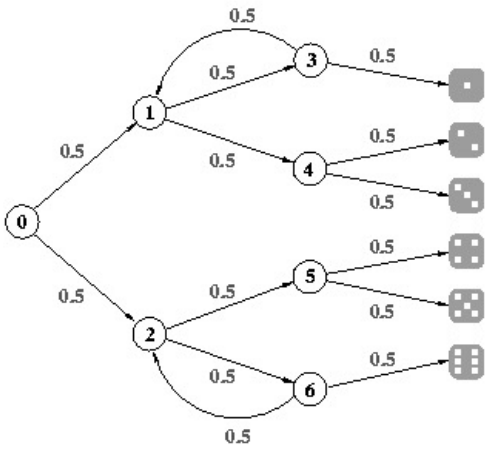

图 1 筛子模型状态转移图

Fig.1 state transition graph for the die model

我们验证性质，$P_{=?}[true\ U^{\leq 5} s = 4]$，即验证模型在 5 个时间单位内筛子的

值出现过为4的概率值。我们通过模拟系统的运行产生n条长度为5的随机路径，并统计这些路径中包含状态"$s = 4$"的条数k，因此，我们可以得出模型满足性质的概率近似为

$$p = \frac{k}{n}$$

SMC 算法作为一种蒙特卡罗方法，具有蒙特卡罗方法的优点，即适用于复杂系统。而且由于 SMC 算法对内存的需求以及时间复杂度与模型复杂度无关等优点，对于一些复杂度极高的模型，SMC 算法往往是唯一的选择。

2.4 总结

本章讨论了 DTMC 模型定义以及性质描述，并阐述了蒙特卡罗方法与统计模型检验算法的基本情况，为后续的改进算法的阐述提供了理论依据。

第三章 基于改进的 SMC 算法以及 BP 神经网络的带参数马尔科夫模型验证框架

3.1 框架概述

为了解决带参数的马尔科夫模型的验证问题，本章首先提出一种新的统计模型检验算法以解决已知参数下马尔科夫模型的检验问题，然后利用模型在若干参数下的验证数据使用基于神经网络的函数拟合的方法计算出模型在未知参数下的模型检验值。

SMC 算法将模型检验看成一个参数估计问题予以解决，即假设马尔科夫模型M满足某一时态逻辑公式ϕ的概率为p，SMC 算法的任务是计算符合可信度和精度要求的p的估计值$\hat{p}$。

与其他 SMC 算法不同，我们在取样的时候引入了旨在减小估计值方差的对偶变量（antithetic variables）取样法，估计值方差的减小可以提高估计的精度。而在使用常规 SMC 算法时，为了提高精度则需提高样本大小。因而对偶变量取样法间接减少了随机路径取样数，提高了 SMC 算法的效率。

在得到了模型在特定参数点下的模型检验值之后，我们针对带参数模型M_θ的满足函数进行函数拟合。由于事先无法确定满足函数的解析形式，于是我们采用神经网络对满足函数进行拟合。不同于常规的神经网络拟合算法，本文从加权最小二乘法中借鉴了估计权值的思想，通过为模型在不同参数下的概率检测值赋予不同的权值，进而表征模型在不同的参数下模型检验值的可靠性，对可靠性高的检验值赋予更高的权重，从而优化了函数拟合的效果。

下面我们主要分为两个部分来叙述本框架采用的算法：基于改进的 SMC 算法的采样算法，以及基于 BP 神经网络的满足函数拟合算法。拟合算法采用 SMC 算法的样本点作为输入，拟合出一条与样本点尽量接近的曲线，从而预测模型在未知参数下的模型检验值。

3.2 基于对偶变量采样法的 SMC 算法

经典的蒙特卡罗方法通过重复性的取样获取对某个值的估计值。通过中心极限定理[33]（central limit theorem），我们可以给出估计值的置信区间。但是对于同一个待估计的值，却可以有多个估计值，每个估计值的方差有大有小。估计值的可信度随着估计值的方差的增大而降低。常见的减小估计值的方差的方法有对偶变量法，共同变量法以及控制变量法等[34][35]。本文吸取了对偶变量并将其运用到 SMC 算法中。下面我们首先给出对偶变量的定义。

定义 1. 对偶变量 称两个定义在实数域的随机变量Y和Y^*为一对对偶变量如果Y和Y^*服从同一分布且两者是负相关的关系。

关于对偶变量我们有如下定理[36]。

定理 1. 设N为一个偶数，$(Y_1, Y_1^*), \dots, (Y_{N/2}, Y_{N/2})$为相互独立的对偶变量对，其中对于任意的整数$k \in [1, N/2]$，$Y_k$和$Y_k^*$均服从于某一特定分布$Y$，则

$$\hat{l}^a = \frac{1}{N}\sum_{k=1}^{\frac{N}{2}}\{Y_k + Y_k^*\}$$

是$l = EY$的无偏估计，并且其方差为

$$\begin{aligned} & Var\left(\hat{l}^a\right) \\ &= \frac{N/2}{N^2}\left(Var(Y) + Var(Y^*) + 2Cov(Y, Y^*)\right) \\ &= \left(Var(Y) + Cov(Y, Y^*)\right)/N \\ &= \frac{Var(Y)}{N}\left(1 + \varrho_{Y,Y^*}\right) \end{aligned}$$

其中，ϱ_{Y,Y^*}为Y和Y^*之间的相关系数。

从以上定理我们可以看出，由于Y和Y^*之间的对偶性使得$\varrho_{Y,Y^*} \leq 0$恒成立，所以

$$Var\left(\hat{l}^a\right) \leq Var(Y)/N = Var(\hat{l})$$

恒成立，即通过对偶变量取样法得出的估计值的方差要小于标准蒙特卡罗方法的估计值。

下面通过一个简单的例子来说明对偶变量是如何减小估计值方差的 。

假设存在随机变量$X = f(U)$，$f(U)$的表达式为

$$f(U) = \begin{cases} 1, if\ U > p \\ 0, if\ U < p \end{cases}$$

其中，参数p未知。现在需要对p进行估计，即运用蒙特卡罗方法计算出一个尽可能准确的p的估计值$\hat{p}$。

我们将f看成是一个黑盒对参数p进行估计。标准蒙特卡罗方法的思路是产生n个独立且服从于均匀分布$U(0,1)$的随机变量$U_1, \dots, U_n$，然后通过函数f得到X的n个样本$X_1, \dots, X_n$，则p的估计值$\hat{p}$为

$$\hat{p} = \sum_{i=1}^{n} X_i / n$$

我们可以得出$\hat{p}$的方差为

$$Var_{\hat{p}} = Var\left(\sum_{i=1}^{n} X_i / n\right)$$

由于X_i相互之间彼此独立，所以$Cov(X_i, X_j) = 0$对任意i, j恒成立，上式可以写成

$$= \frac{1}{n^2}\left(\sum_{i=1}^{n} Var(X_i)\right) = Var(X)/n$$

采用对偶变量的蒙特卡罗方法和上述方法略有不同。假设样本大小n为偶数，即$n = 2k$。我们首先产生k个独立且服从于均匀分布$U(0,1)$的随机变量$U_1, \dots U_k$，剩下的k个随机变量则取作$U_1^a = 1 - U_1, \dots, U_k^a = 1 - U_k$，$U_i^a$的上标$a$表示该随机变量是$U_i$的对偶变量。类似地，我们给出$\hat{p}$的表达式：

$$\hat{p}^a = \sum_{i=1}^{k} (X_i + \dots + X_k + X_1^a + \dots + X_k^a) \Big/ 2k$$

类似的我们给出$\hat{p}^a$的方差

$$Var(\hat{p}^a) = \frac{Var(X)}{n}\left(1 + \varrho_{X,X^*}\right)$$

其中，ϱ_{X,X^*}等于

$$\varrho_{X,X^*} = 2\sum_{i=1}^{k} Cov(X_i, X_i^a)$$

可以看出，由于X_i和X_i^a按照上述方法获取并且当两者不相等时，恒有$Cov(X_i, X_i^a) \le 0$即$\varrho_{X,X^*} \le 0$恒成立，所以我们有下式成立

$$Var(\hat{p}^a) \leq Var(\hat{p})$$

下面分析方差的减小将如何影响样本的大小。

根据中心极限定理 ，均值为μ，方差为σ^2的独立同分布的随机变量$X_1, \dots, X_n$的算数平均$\bar{X} = \frac{1}{n}\sum_{k=1}^{n} X_k$，当$n$充分大时近似服从均值为$\mu$，方差为$\sigma/n$的正态分布。

另外关于样本方差S^2和均值$\bar{X}$我们还有以下定理[37]：

定理 1 设$X_1, \dots, X_n$分别是来自于总体$N(\mu, \sigma^2)$的样本，$\bar{X}$，S^2分别是样本的期望和方差，则有

$$\frac{\bar{X} - \mu}{S/\sqrt{n}} \sim t(n-1)$$

上述定理表示$\frac{\bar{X}-\mu}{S/\sqrt{n}}$近似服从参数为$n-1$的 t 分布。

所以，我们有

$$P\left\{-t_{\alpha/2} < \frac{\bar{X} - \mu}{S/\sqrt{n}} < t_{\alpha/2}\right\} = 1 - \alpha$$

即

$$P\left\{\bar{X} - \frac{S}{\sqrt{n}} t_{\alpha/2} < \mu < \bar{X} + \frac{S}{\sqrt{n}} t_{\alpha/2}\right\} = 1 - \alpha$$

于是我们得到μ的一个置信水平为$1-\alpha$的置信区间为

$$(\bar{X} \pm \frac{S}{\sqrt{n}} t_{\alpha/2})$$

因为S^2为σ^2的无偏估计，而估计值$\bar{X} \sim N(\mu, \sigma^2/n)$，于是我们有

$$\frac{S}{\sqrt{n}} = \frac{\sigma}{\sqrt{n}} = \sqrt{Var(\bar{X})}$$

所以上述置信区间也可以写作

$$(\bar{X} \pm \sqrt{Var(\bar{X})} t_{\alpha/2})$$

分析上述置信区间我们可以发现，估计值$\bar{X}$的方差的减小能有效减小置信区间的宽度，这也从理论上反映出估计值的方差的减小能提高估计值的精度。进一步，假设估计值的方差减小为原来的$\frac{1}{k}$，则为达到同样的精度，假设估计值的方差不变，样本大小n必须增到原来的k倍，因此我们有以下结论，即

结论： 假设现在要求大小为$2d$的置信区间为未知参数p进行估计，如果使

用标准蒙特卡罗方法需要大小为n的样本，而使用对偶变量法使得估计值$\hat{p}^a$的方差是标准估计值$\hat{p}$的$\frac{1}{\mathrm{k}}$倍，则使用对偶变量法需要的样本大小为$\frac{n}{k}$。

下文将阐述如何在 SMC 算法中运用上述思想。首先我们给出一条有限长的随机路径的对偶路径的定义，接着我们证明，通过利用对偶路径改进传统的 SMC 算法，可以有效减小估计值的方差，从而提高估计的精度。

通过回顾在第二章中描述的 SMC 算法可知，马尔科夫模型中的一条随机路径唯一对应于一个由随机数组成的向量U。 借助于这一点，我们给出适用于 DTMC 和 CTMC 的对偶路径的定义。

定义 1 马尔科夫模型的对偶路径 对应于任意一条马尔科夫模型产生的随机路径$p = Path(U)$，其对偶路径为$p^a = Path(1 - U)$。

以下是完整的产生对偶路径的算法。

算法 2. 基于对偶变量的 DTMC 模型随机路径产生算法

输入：模型 DTMC，路径长度 d

输出：长度为 d 的两条对偶路径

```
if d < 0 then
        return empty path
else
        path1 = path()
        path2 = path()
        for i = 1 to d do
                rnd = uniformRandom() // 产生[0,1]上服从均匀分布的随机数
                antiRnd =  1 – rnd
                nextState = chooseNextState(rnd)
                nextState2 = chooseNextState(antiRnd)
                path1.append(nextState)
                path2.append(nextState2)
        end for
```

return path1, path2

以上算法的输入同算法 1，但是不同于算法 1 只返回一条随机路径，上述会返回两条互为对偶路径的随机路径。观察以上算法可以发现，算法 2 在为$path1$产生下个状态而生成随机数rnd的同时，会将$1-rnd$作为第二条随机路径选择下个状态的依据，从而产生了两条互为对偶路径的随机路径。

通过对 SMC 算法的阐述我们知道，马尔科夫模型的验证问题本质上是一个 0-1 分布的参数的估计问题。假设模型M满足LTL公式的概率为p，我们的任务是计算出一个尽可能接近p的估计值$\hat{p}$。

SMC 算法的思路可以用下式来表示

$$\hat{p} = \sum_{i=1}^{n} check(path)/n$$

其中，$check$函数为

$$check(path) = \begin{cases} 1, if\ path \vDash \phi \\ 0, if\ path \nvDash \phi \end{cases}$$

通过对 0-1 分布参数估计的问题的阐述我们发现，只要保证两条路径p_i, p_i^a的验证结果$check(p_i)$和$check(p_i^a)$不同就可以减少估计值$\hat{p}$的方差。然而通过观察对偶路径的定义，我们发现上述定义无法保证互为对偶路径的两条路径的验证结果不同，此问题可以通过定义状态之间的一种偏序关系得以解决。

定义 2 状态之间的偏序关系 对于马尔科夫模型M，LTL 公式ϕ，对于M中的任意状态S，我们定义$s(S)$为所有经过S且满足公式ϕ的路径的总条数，$c(S)$为所有经过S的路径的总条数。对于任意M中两个状态S和T，我们称S和T满足偏序关系R当且仅当

$$\frac{s(S)}{c(S)} \leq \frac{s(T)}{c(T)}$$

下面借助一个简单的模型阐述上述偏序关系的使用。

假设模型M的状态转移图如下图所示：

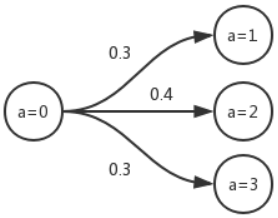

图 2 模型M状态转移图

Fig.2 State transition graph for M model

假设我们的任务是求模型M满足公式$true\ U \leq 5\ a = 2$的概率，根据 DTMC 有限长随机路径产生算法， 只需产生 n 个服从 0 到 1 之间的均匀分布的随机数即可，假设产生的随机数为$U_1, \ldots, U_n$。可以发现，当路径的最后一个状态为 a=2 时，该路径满足公式ϕ。同样是根据算法 1[DTMC 有限长路径产生算法]，我们发现，系统产生的两条对偶路径的路径的验证结果$check(p_i)$和$check(p_i^a)$并不一定不同，于是我们使用上述定义的偏序关系重新排列初始状态的所有使能状态（enabled state）。

要对状态进行排序，首先需要知道每个状态S的$s(S)$和$c(S)$，所以我们首先需要产生一定数量的路径并进行验证，然后利用这些路径对模型中的状态进行成功路径和失败路径的计数，最后根据$\frac{s(S)}{c(S)}$对状态进行排序，完整算法见下图所示。

算法 1 状态重排序算法

输入：n 条随机路径$p_1, \ldots, p_n$组成的数组 paths 以及相应的验证结果$r_1, \ldots, r_n$组成的数组 results，模型 M

输出：None

```
for p, r in zip(paths, results) //  同时遍历路径数组和验证结果数组
        for state in p
                if r == true:
                    state.s_cnt += 1 // s(S)加一
                state.c_cnt += 1 // c(S)加一
        end for
end for
```

```
for state in M
        states = next(state) // 获取状态 state 的全部使能状态
        sort(states, key=state.s_cnt/state.c_cnt) // 根据 state.s_cnt/state.c_cnt 对状态进行排序
end for
```

我们利用上述算法对上述模型进行状态重排序，实验对模型M总共产生了 10 条路径，其中有 4 条为满足路径，其余的 6 条为失败路径，经过计数后可以发现状态 1, 2, 3[换一种表述?]的$s(S)/c(S)$值依次是 0，1 和 0，于是排序后，状态 2 将排在初始状态全部使能状态的最后。

可以发现，经过使能状态重排序后，模型产生的两条互为对偶路径的路径的验证结果$check(p_i)$和$check(p_i^a)$不同的可能性大为提升，从而可以提升估计值$\hat{p}$的方差。

以下是完整的基于对偶路径的 SMC 算法。

```
算法 3. 基于对偶路径的 SMC 算法
输入：DTMC 模型，bounded LTL 公式φ，置信区间参数α，d
输出：DTMC 模型在初始状态下满足特定 PCTL 公式的概率
n = 0 // 抽样总个数
positive = 0 // 验证结果为 true 的路径总数
samples =  computeSamples(α, d)
length = getLength(φ) // 解析 bounded LTL 公式，获取取样路径长度
if samples % 2 != 0 then
        samples += 1
end if
for i = 1 to samples / 2
        path1, path2 = genRandomPathAnti(length) // 产生长度为 length 的两条随机路径
        if checkLTL(path1, φ) == true then
```

```
                positive += 1
        end if
        if checkLTL(path2, φ) == true then
                positive += 1
        end if
        n += 2
end for
```

可以看出，上述算法在传统 SMC 算法的基础上，通过采用对偶算法生成随机路径，除了可以减小随机数的生成数量外，还可以提高算法估计的准确性。

3.3 估计值加权的神经网络函数拟合方法

理论表明，神经网络可以以任意精度拟合任意函数[38]。所以，在很多传统函数拟合方法无法适用的场景，就可以采用神经网络作为拟合特定曲线的手段。例如，在生产生活中已知变量之间存在相互关系，但是无法确定其解析表达式时，则无法采用传统的最小二乘法对曲线进行拟合。

在本节中，我们首先会阐释神经网络的基本原理，先后阐释用于分类问题和回归问题的神经网络。接着，我们会引入在加权最小二乘法中引入的估计权值的概念，并阐述在神经网络的训练算法中如何引入这一概念帮助提高神经网络的拟合性能，最后我们给出完整算法并用例子说明。

在阐述本章将要介绍的算法之前，回顾我们在第二章中介绍的理论背景。本质上来讲，系统中的每条路径均对应一个实数值，这个实数值反应了这条路径在所有路径中被抽中的概率。而一个 PCTL 公式的验证结果可以用所有满足相应 LTLφ公式的路径的概率之和来表示，即

$$P(D_\theta \vDash \phi) = \sum_{path} p(path)\ if\ path \vDash \varphi$$

根据定理 1[神经网络可以拟合任何连续函数]，任意一个连续函数都可以被一个包含任意多个神经元的三层 BP 神经网络以任意精确度拟合。于是，我们首

先定义一个带参数的离散马尔科夫模型的满足函数，接着证明该满足函数在定义域内是连续函数。

定义 1. 满足函数 给定带参数的离散马尔科夫模型M_θ，PCTL 公式ϕ，假设θ的取值范围为Dim_θ，定义定义域为Dim_θ，值域为$[0,1]$的函数S

$$S_\phi(\theta) = p(\phi \vDash M_\theta)$$

即，给定Dim_θ内的一个点θ_i，我们定义了满足函数的值为带参数模型M_θ在参数取θ_i时模型满足公式ϕ的概率值。

下面证明上述定义的满足函数在定义域内是连续的。

定理 2. 给定一个带参数的马尔科夫模型M_θ，假设模型中任意两个状态之间的转化概率t_i关于θ均是连续函数的话，那么其对应的满足函数$S_\phi(\theta)$在定义域内是连续函数。

证明：由前所述，模型M_θ满足 PCTL 公式ϕ的概率可以用满足相应 LTL 公式φ的路径概率之和来表示。假设用$PATH_\varphi$来表示该集合。可以发现，$PATH_\varphi$与参数θ的具体取值无关。假设用$\Pr(PATH_\varphi)$表示该集合的路径概率之和。所以，我们只要证明$\Pr(PATH_\varphi)$是关于θ的连续函数即可。

又由于$\Pr(PATH_\varphi) = \sum_{path} \Pr(path)\ with\ path \vDash \varphi$，所以我们只要证明任何一条随机路径被抽中的概率值$\Pr(path)$是$\theta$的连续函数即可。

由第二章中的理论背景可知，

$$\Pr(path) = p_1 p_2 \dots p_l$$

其中l是随机路径$path$的长度，p_i是某个转化发生的概率。由已知条件，p_i关于θ是连续的，由连续函数的性质可知，$\Pr(path)$关于θ是连续的，即结论得证。

结合定理 1 和定理 2 可知，我们可以使用 BP 神经网络拟合模型M_θ对应的满足函数。

3.3.1 用于分类和回归的 BP 神经网络

下面通过一个简单的分类问题说明神经网络的基本工作原理。

考虑识别手写体数字的问题[39]。下图展示了一些手写体数字的例子。

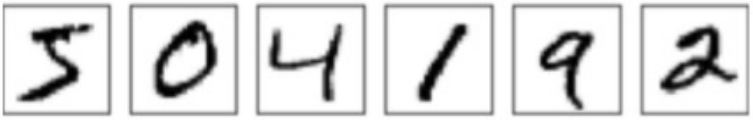

图 3 手写体数字

Fig.3 Handwriting figures

在阐释 BP 神经网络之前，我们先给出 S 型神经网络的定义。

下图表示一个 S 型神经元。

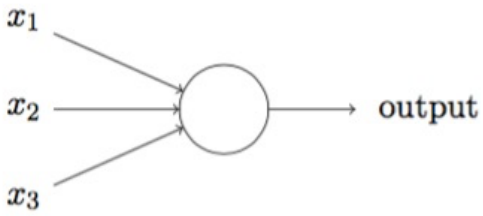

图 4 S 型神经元

Fig.4 Logistic neural

上述 S 型神经元接受x_1, x_2, x_3作为输入，$output$是其输出。$output$的表达式如下所示：

$$output = \frac{1}{1 + e^{-(\sum_{i=1}^{3} w_i x_i) - b}}$$

其中，w_i表示 S 型神经元的权重，b表示其偏置。其中，$\frac{1}{1+\exp\{-x\}}$（即逻辑函数，也叫 S 型函数）的形状酷似英文字母’s’，这也是 S 型神经元命名的由来。S 型神经元是 BP 神经网络的基本组成单元。

一个典型的 BP 神经网络由三层神经元组成，分别为输入层，隐藏层以及输出层。输入层和输出层包含的神经元的个数与具体的输入以及问题的类型有关，隐藏层的神经元的个数属于超参数，一般是事先给定或者根据问题规模进行选择。

使用神经网络解决特定问题一般大致分为两步：训练与预测。前者是神经网络的训练阶段，最常见的算法是随机梯度下降算法（stochastic gradient descent, SGD）。后者包含了一个逐层前向传播的过程，所以一般也称为前向传播算法。下面依次叙述上述算法。

神经网络的训练问题可以这样描述，假设用ω表示神经网络$\mathcal{N}$的所有神经元的权重组成的向量，b表示所有神经元的偏置，我们定义如下的代价函数：

$$C(\omega, b) = \frac{1}{2n}\sum_{x} \| y(x) - a \|^2$$

其中，n表示样本的个数，x表示每个训练样本的输入，$y(x)$表示x对应的训练样本的输出，a表示神经网络在输入为x的情况下的输出值，$\| v \|$表示向量v的模。我们一般把C称之为二次代价函数。

有了代价函数，我们这样叙述 BP 神经网络的训练问题，即给定一组训练数据$(x_1, y_1), \dots, (x_n, y_n)$，我们需要找到使得代价函数$C$最小的ω和$b$。

如何找到合适的ω以及b使得C最小？这里经典的 BP 神经网络使用随机梯度下降。一个多元函数$f(x_1, \dots, x_n)$的梯度∇f的定义如下：

$$\nabla f = \left(\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n}\right)^T$$

从梯度的定义可以看出 ，多元函数的梯度的每个分量是函数关于每个分量的偏导数。

关于梯度有如下显而易见的定理：

定理 1. 对于多元函数$f(x_1, \dots, x_n)$来说，当自变量的增量$\Delta x = -\eta \nabla f$时，$\nabla f \cdot \Delta x \leq 0$。

上述定理可以由模的非负性证明。根据微积分，当函数的自变量发生了一个很小的变化时，如$\Delta x_1, \dots, \Delta x_n$，函数值的变化量可以用下式近似表示：

$$\Delta f \approx \sum_{i=1}^{n} \frac{\partial f}{\partial x_i} \cdot \Delta x_i$$

结合上述两式，我们可以得出当$\Delta x = -\eta \nabla f$时，函数值增量$\Delta f \leq 0$恒成立。所以，通过不断计算函数$f$在当前点的梯度，从而计算出自变量的增量，就可以不断地使得函数值减小，直至达到最小值。

那么如何运用梯度下降算法求解式[?代价函数]呢？观察式[?代价函数]，我们可以发现C是ω和b的函数，对于任意的ω_k与b_l ，我们均使用下式进行更新：

$$\omega_k \to \omega_k' = w_k - \eta \frac{\partial C}{\partial \omega_k}$$

$$b_l \to b_l' = b_l - \eta \frac{\partial C}{\partial b_l}$$

根据定理 1 以及其推论，我们有$C(\omega', b') \leq C(\omega, b)$恒成立。我们只需反复运用上述两式对$C$中的每个$\omega_k$以及$b_l$进行迭代，直至达到某个迭代停止条件即可。

虽然从理论上分析梯度下降算法有较好的效果，但是从实践运用上来看仍存在不少问题，其中最严重的就是算法的运行效率问题。从上述两式可以看出，在每次迭代过程中，算法都要遍历训练样本中的每个x进而求平均值，这样的效率不高。如果每次迭代只以少样样本作为迭代，效率可大大提高，而这也正是随机梯度下降的思路。

完整的随机梯度下降算法如下所示。

算法 1. 随机梯度下降算法

输入：训练样本$(x_1, y_1), \ldots, (x_n, y_n)$

输出：代价函数C的最小值

initialize ω and b

repeat until convergence {

randomly choose m samples from $(x_1, y_1), \ldots, (x_n, y_n)$

$\omega_k = \omega_k - \frac{\eta}{m} \sum_{i=1}^{m} \frac{\partial C_{X_i}}{\partial \omega_k}$

$b_l = b_l - \frac{\eta}{m} \sum_{i=1}^{m} \frac{\partial C_{X_i}}{\partial b_l}$

}

return ω and b

在上述算法中我们可以发现，随机梯度下降算法的每次迭代中，主要的工作就是计算代价函数关于神经网络中的权重和偏置的偏导数，BP 神经网络使用反向传播（back propagation）算法[40]对神经网络中每个神经元的权重和偏置进行更新。反向传播算法本质上使用导数的链式法则对神经网络进行逐层求偏导，与经典的神经网络训练速度相比，效率得到了极大的提高。

神经网络的前向传播算法主要用于预测与分类，给定神经网络的输入，利用前向传播算法对神经网络进行逐层求值，从而得出最后的输出。具体的，假设神

经网络中第$l-1$层中第k^{th}个神经元到第l层中第j^{th}个神经元的链接上的权重 为ω_{jk}^{l}，第l层上第j^{th}个神经元的偏置为b_j^l，第l层上第j^{th}个神经元的激活值为a_j^l，那么第l层上的第j^{th}个神经元的激活值就可以用下式表示：

$$a_j^l = \sigma(\sum_k \omega_{jk}^{l-1} a_k^{l-1} + b_j^l)$$

当采用向量表示每一层的各个神经元的偏置，用矩阵表示层和层之间链接的权重时，我们可以将上式写成

$$a^l = \sigma(\omega^l a^{l-1} + b^l)$$

迭代地利用上式，我们就可以由输入层逐层向后计算，从而计算出输出层的值。

至此我们将神经网络的主要的基本原理叙述完毕，概括地说，神经网络使用反向传播算法计算代价函数关于权重和偏置的偏导数，利用随机梯度下降算法对权重矩阵和偏置向量进行迭代更新，从而求解使得代价函数最小的权重矩阵和偏置向量。得到这两个值之后，就可以利用训练好的神经网络对模型在未知输入下的输出进行预测，预测采用前馈算法。对于分类问题，输出层神经元的激活函数一般采用逻辑函数，逻辑函数将$\mathcal{R}$映射到[0,1]，因此，对于二分类问题，可以如下定义：

$$label(y) = \begin{cases} label_{yes}\ if\ y \geq 0.5 \\ label_{no}\ if\ y < 0.5 \end{cases}$$

而对于回归问题，输出层神经元的激活函数一般采用线性函数，即$activition(a_1, a_2, \dots, a_h) = \sum_{i=1}^{h} \omega_i^o \cdot a_i + b$。

3.3.2. 加权神经网络拟合算法

但是，上述算法没有考虑到每个样本的可靠性。由于 SMC 算法本质上是一种蒙特卡罗方法，其得到的结果本身就带有一定偏差。而且，模型在不同参数下运行 SMC 算法获取的数据的偏差也不同。神经网络在训练时应该考虑到训练数据的可靠性。对于神经网络来说，如果已知观测值y_k的可靠性要高于y_{k+1}，那么应该给$\mathrm{y_k}$赋予更大的权重。换句话说，因为y_k的可靠性要高于y_{k+1}，因此其对函数拟合的影响也更大。下面，我们给出训练数据可靠性权重的概念 。

定义 1.可靠性权重 可靠性权重表达了我们对于一个训练数据可靠性的度量，可靠性越高的观测数据，其在神经网络训练中的可靠性权重也越高。

上述定义仅从功能性方面阐述了可靠性权重，并未给出了可靠性权重应该如何定义。从直观上讲，可靠性权重应该由估计值的标准差唯一决定，即估计值的标准差越大，其可靠性程度越低。但是，如果要为每个估计值均计算它的标准差，计算代价又更大。因此本文提出一种分段计算标准差的方法。它首先将x轴进行分段，在每一段中取若干个样本点，然后用直线对这些样本点进行线性拟合，拟合后可以计算出这些样本点到直线的平均偏移量，这个偏移量实际上表达了该段数据点的总体不确定性，平均偏移量越大，说明该段数据点的不确定性也越大。

针对第k个子段B_k中的N_k个数据点，我们使用直线$y_i = a_1 + a_2 x_i + \varepsilon_i \ (\forall x_i \in B_k)$对数据点进行拟合。任意一个观测值$y_i$与这条直线的偏差反映了每一段观测值的不确定性，它可以由下式表示：

$$\sigma_{y,k} = \sqrt{\frac{1}{N_k - 2} \cdot \sum_{x_i \in B_k} (y_i - (\hat{a}_1 + \hat{a}_2 x_i))^2}$$

基于上式，我们可以给出该段B_k内所有样本点的可靠性权重为

$$w = \frac{1}{\sigma_{y,k}^2}$$

下面给出完整的权重计算算法。

算法 2. 样本点权重计算算法

输入： 第k段B_k内所有样本点，$(x_1, y_1), \dots, (x_{N_k}, y_{N_k})$

输出：该段内所有样本点可靠性权重

k = 1.0 // 直线斜率

b = 0.0 // 直线截距

$\hat{k}, \hat{b}$ = leastSquare$(x_1, y_1, \dots, x_{N_k}, y_{N_k})$ // 执行标准拟合直线最小二乘算法，得出斜率和截距的估计值

standardDiv = standardDiv$(x_1, y_1, \dots, x_{N_k}, y_{N_k}, \hat{k}, \hat{b})$

return 1.0 / $standDiv^2$

在实际的运用中，为了保证权重的引入不会影响到神经网络的训练，需要对权重采用归一化措施，即

$$W_i = \frac{W_i}{\sum_i W_i} \ \#\{W_i\}$$

在介绍加权神经网络拟合算法之前，我们需要回顾一下经典的反向传播算法。

首先让我们定义以下符号：假设我们定义从$l-1$层第k个神经元到第l层的第j个神经元的链接的权重为w_{jk}^l，第l层中第k个神经元的偏置为b_k^l，第l层中第k个神经元的激活值为a_k^l，第l层上第k个神经元的带权输入为$z_k^l = \sum_{k=1}^{N_{l-1}} w_{jk}^l a_k^{l-1} + b_j^l$，用向量形式表示即为$z^l = w \cdot a^{l-1} + b^l$。

我们定义第l层上第j^{th} 个神经元的误差δ_j^l为

$$\delta_j^{\mathrm{l}} = \frac{\partial C}{\partial z_j^l}$$

经典的反向传播算法包含四个方程，首先，反向传播算法需要计算输出层的误差，即

$$\frac{\partial C}{\partial z_j^L} = \frac{\partial C}{\partial a_j^L} \ \sigma'(z_j^L)$$

上式的推倒依赖于偏导数的链式法则，其中$\partial a_j^L / \partial z_j^L = \sigma'(z_j^L)$，$\sigma$为逻辑函数。

上式用向量形式重写即为

$$\frac{\partial C}{\partial z^L} = \nabla C_a \odot \sigma'(z^L)$$

其中，$z^L = \left(z_1^L, \dots, z_{N_o}^L\right)^T$，$\nabla C_a = \left(\frac{\partial C}{\partial a_1^L}, \frac{\partial C}{\partial a_2^L}, \dots, \frac{\partial C}{\partial a_{N_o}^L}\right)^T$，$\sigma'(z^L) = (\sigma'(z_1^L, \dots, z_{N_o}^L)$。

然后，我们利用神经网络下一层的误差计算当前层的误差，假设我们需要计算第l层神经元中第j^{th}个神经元的误差，即$\frac{\partial C}{\partial z_j^l}$。由于上一层某个神经元的误差会传递到下一层所有神经元，所以需要对下一层的神经元的误差进行求和。

$$\frac{\partial C}{\partial z_j^l} = \sum_k \frac{\partial C}{\partial z_k^{l+1}} \frac{\partial z_k^{l+1}}{\partial z_j^l} = \sum_k \delta_k^{l+1} \frac{\partial z_k^{l+1}}{\partial z_j^l}$$

由于

$$\frac{\partial z_k^{l+1}}{\partial z_j^l} = w_{kj}^{l+1}\,\sigma'(z_j^l)$$

故上式可以改写为

$$\delta_j^l = \frac{\partial C}{\partial z_j^l} = \sum_k \delta_k^{l+1} w_{kj}^{l+1} \sigma'\left(z_j^l\right)$$

而上式的向量形式为

$$\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$$

通过上述两个公式，我们可以计算出神经网络每一层的误差。而通过误差，我们可以求出代价函数关于神经网络中每个权重以及偏置的偏导数。

具体地，我们有代价函数关于第l层神经元中第j^{th}个神经元的偏置的偏导数为$\partial C/\partial b_j^l$：

$$\frac{\partial C}{\partial b_j^l} = \frac{\partial C}{\partial z_j^l}\frac{\partial z_j^l}{\partial b_j^l} = \frac{\partial C}{\partial z_j^l}$$

而代价函数C关于某个权重w_{jk}^l的偏导数为

$$\frac{\partial C}{\partial w_{jk}^l} = \frac{\partial C}{\partial z_j^l}\frac{\partial z_j^l}{\partial w_{jk}^l} = \delta_j^l a_k^{l-1}$$

综上所述，反向传播算法通过计算输出层的误差然后逐层往后计算每一层神经元的误差，并通过链式法则计算代价函数关于权重和偏置的偏导数。

回顾代价函数C，

$$C = \frac{1}{2n}\sum_x \| y(x) - a \|^2 = \frac{1}{n}\sum_x C_x$$

如前所述，不同的样本具有不同的可靠性权重，因此我们改写上式，即

$$C = \frac{1}{2n}\sum_x W_x \| y(x) - a \|^2 = \frac{1}{n}\sum_x W_x C_x$$

可以发现，可靠性越高的样本，在代价函数中所占的比重也就越大，其对权重和偏置的影响也就越大。而可靠性越低的样本，因为被赋予了更小的权重，对神经网络的影响也就越小。

下面给出完整的加权神经网络梯度下降算法。

算法 4. 加权神经网络随机梯度下降算法

输入：神经网络的权重向量ω以及偏置向量b，n个训练样本$(x_1, y_1), \ldots, (x_n, y_n)$，样本的权重向量$(W_1, \ldots, W_n)$

输出：训练好的神经网络的权重ω以及b

initialize ω and b

repeat until convergence {

randomly choose m samples from $(x_1, y_1), \ldots, (x_n, y_n)$

$\omega_k = \omega_k - \eta \sum_{i=1}^{m} W_i \frac{\partial C_{X_i}}{\partial \omega_k}$

$b_l = b_l - \eta \sum_{i=1}^{m} W_i \frac{\partial C_{X_i}}{\partial b_l}$

}

return ω and b

3.4 完整算法描述

下面通过一个例子描述算法的整个运行过程。

我们以如下模型为例。

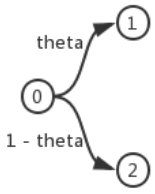

图 5 包含三个状态的带参数马尔科夫模型

Fig.5 Parameterized Markov chain model containing three states

从上述状态转移图我们可以看出，模型在状态 0 下的转移依赖于参数θ，从状态 0 转移至状态 1 的概率为θ，转移到状态 2 的概率为$1-\theta$，状态 1 和状态 2 均为吸收状态。模型需要验证的公式为

$$P_{=?}(true\ U^{\leq 5}\ s = 1)$$

表示系统在 5 个时间单位内到达状态 1 的概率。

由于系统的行为取决于参数θ的具体值，所以上述模型是一个带参数的马尔科夫模型。这里，我们对参数θ在$[0,1]$内进行验证。

首先我们对$[0,1]$区间进行分段，假设将其均匀分为 10 段。依据算法[?加权随机梯度下降]，我们需要在每一段内取若干个样本点，运行 SMC 算法，假设每段取 100 个样本点。对每个参数点θ，我们运行基于对偶路径的 SMC 算法，得到 100 个样本点，$(x_1, y_1), \ldots, (x_{100}, y_{100})$。以下是我们在区间段$[0, 0.1]$内得到的 100 个样本点。

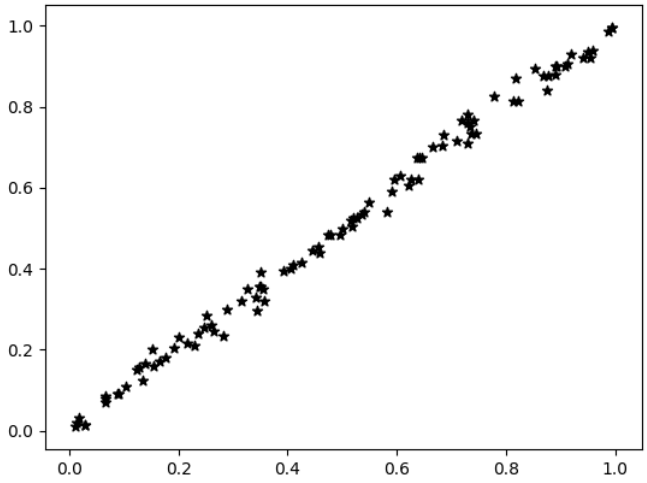

图 6[0,1]区间内 100 个样本点分布图

Figure.6 100 samples residing in interval of [0, 1]

接着，我们需要为这 100 个样本点确定可靠性权重，将这 100 个样本点作为输入，运行线性拟合的最小二乘算法，得到直线$y = \hat{k}x + \hat{b}$，求出样本到直线的平均偏差σ，求出每个样本点的可靠性权重$W = \frac{1}{\sigma^2}$。重复为每个区间运行上述算法，计算出每个样本区间的可靠性权重。

下表为各个区间段的可靠性权重。

表 1 各区间段的可靠性权重

区间	权重
[0.0, 0.1)	2.005

[0.1, 0.2)	0.362
[0.2, 0.3)	0.880
[0.3, 0.4)	0.368
[0.4, 0.5)	1.287
[0.5, 0.6)	1.251
[0.6, 0.7)	0.4253
[0.7, 0.8)	0.865
[0.8, 0.9)	0.509
[0.9, 1.0)	2.045

观察图 5 以及表 1 可以发现，区间内的样本点越集中意味着区间中的样本点的可靠性越高，代表着这些样本点的权重也应该越高。得到了 1000 个样本点以及每个样本点对应的可靠性权重后，将这些信息输入给神经网络进行训练，就可以得到训练好的神经网络模型。下图是训练好的神经网络预测结果和利用 PRISM 软件对模型进行求解的对比图。

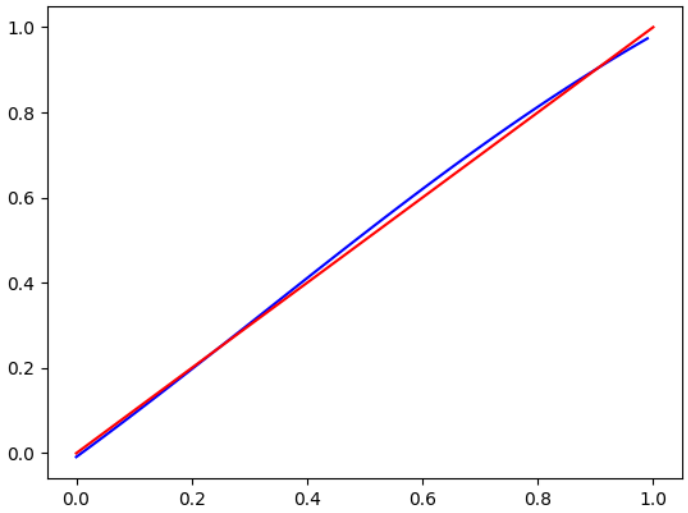

图 7 预测结果与 PRISM 结果对比图

Fig.7 Comparison between predictions and results from PRISM

从上图可以看出，神经网络预测的结果与 PRISM 软件通过解析计算出的结

果相差无几。

可以认为 PRISM 工具的验证的结果为真实值，通过计算可以得出， 神经网络和 PRISM 在未知点（θ依旧位于[0，1]之间）的预测结果的误差为 0.0096，满足精确度要求。

3.5 小结

本章分别阐述了基于对偶路径改进的 SMC 算法以及基于加权的神经网络训练算法，并结合具体案例给出了完整的算法描述。

第四章 带参数模型检测预测工具 SpacNN

4.1. 工具概述

SpacNN（Statistical Probabilistic Approximate Model Checker Using Neural Network）是根据上述理论基础开发的一款带参数离散型马尔科夫模型检测工具。该工具使用 python2.7 开发，集成了编译，解析，检测，训练以及预测等功能。支持在 Mac OS X 以及 Windows 平台下对带参数离散马尔科夫模型进行验证，下图为工具的操作界面。

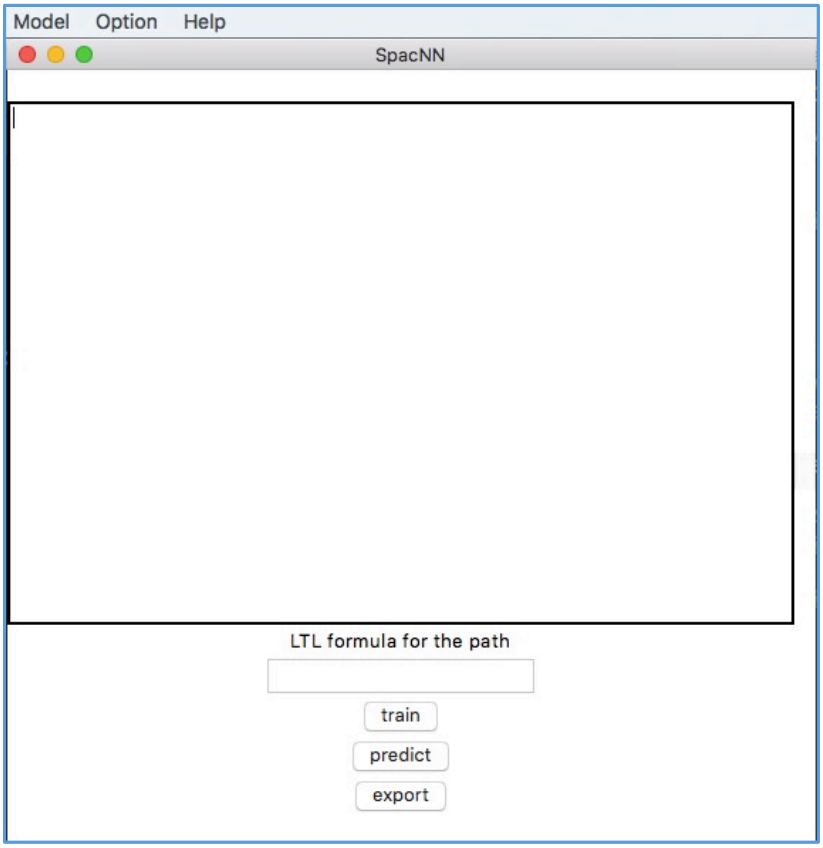

图 8 SpacNN 主界面

Figure.8 main front page of SpacNN

SpacNN 的主要特性有：

1. 支持 PRISM 定义的模型语言，也就是说，SpacNN 支持 PRISM 所支持的模型文件格式。

2. 支持 LTL 公式的部分子集，同样支持 PRISM 语言所定义的 LTL 公式语言。

3. 实现了第三章中提到的改进的 SMC 算法以及加权算法，支持对神经网络的训练以及利用神经网络对模型在未知参数下进行模型检验。

4.2. 工具实现

本工具主要使用 python2.7 进行开发，底层使用 python 核心库以及 numpy，matplotlib 等第三方库用于矩阵运算以及图形绘制。软件的架构图如下所示。

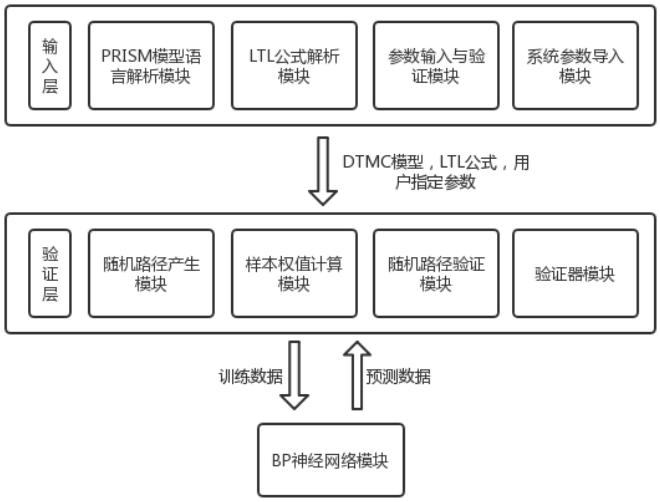

图 9 SpacNN 架构图

Figure.9 Structure of SpacNN

可以将 SpacNN 软件的架构分为两部分来看，第一部分 checker，主要定义了模型的底层表示，相关功能类以及工具类。其负责的主要功能主要有：模型的定义，随机路径的产生，有限长路径的验证，权值的产生等等。第二部分 regressor，定义了使用加权反向传播算法的神经网络，向上层提供训练以及预测功能。

下面将分模块依次介绍各个模块的主要功能。

4.2.1. 输入层

输入层的各个模块，主要负责和用户的交互工作，并提供作为检验层输入的输出。在用户打开软件的时候，启动程序首先启动 UI 程序，提供页面给用户操作，考虑到软件的专业性，软件界面在设计之初就考虑到了用户使用的友好性。软件在影响到结果的关键步骤都设置了提示消息，确保用户对软件提供的关键功

能不会遗漏。并且，软件在简洁性也做了相应的设计，使得界面只保留必要的功能，确保用户可以在拥有最少相关知识的情况下使用软件。

软件需要用户提供两方面的输入，分别是模型文件和模型应该满足的时态逻辑公式。用户分别通过选择.prism 格式的文件和以及手动输入公式的方式输入上述两项。在用户完成输入后，PRISM 模型语言解析模块和 LTL 公式解析模块会对用户的输入进行解析。

用户必须确保所输入的 PRISM 文件中不包含语法错误。PRISM 模型语言解析模块会获取到 PRISM 文件的文本，在从文本中去除注释后，会对文本进行逐行解析。解析程序会从用户输入的文本中获取模型类型，模块名，变量，常量以及转化关系等信息，并保存在相应的数据结构中。软件使用$ModulesFile$类保存此信息，如下所示。

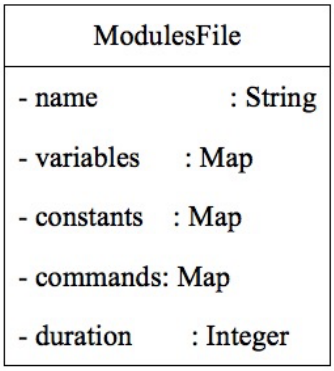

图 10 ModulesFile UML 结构图

Figure.10 UML for class ModulesFile

如上所示，其中$duration$用于指定在产生随机路径时的路径长度。

与 PRISM 文件的输入方式不同，用户在输入 LTL 公式时需要手动输入。同样的，在用户输入 LTL 公式后，LTL 公式解析模块对将 LTL 公式解析成一棵抽象语法树（abstract syntax tree，AST），在获取到 LTL 公式对应的 AST 后，LTL 公式解析模块会将该树的层次遍历结果作为输出提供给验证层的随机路径验证模块进行验证。例如，对于输入”true U<=10 failure”，解析模块的输出结果为”U[0, 10], true, failure”，可以看到遍历结果之间用逗号隔开。

在用户完成上述两项输入之后，用户可以对验证过程中的参数进行设定，如下图所示。

图 11 验证参数设定截图

Figure.11 setting parameter of SpacNN page

其中，隐藏层神经元个数，矫正率以及学习速率属于神经网络模块所需的参数，另外训练样本取样数表示随机路径产生模块产生随机路径的数目。

最后，在程序加载之前，软件会提前到系统参数文件中读取软件中各个参数的默认值，用户也可以修改该默认值，避免每次打开软件时均要修改。经过输入层的处理后，系统给下层提供了模型类对象，LTL 公式以及用户设定的参数。

4.2.2 验证层

验证层的主要工作有负责从某个状态产生指定数量的随机路径，验证产生的随机路径，负责分段计算所产生的样本的可靠性权值，下面分条分析。

其中，随机路径产生模块负责产生指定数目的随机路径。随机路径的产生实质上是产生一系列随机数的过程。对于离散时间马尔科夫链模型来说，在决定下个状态的时候，需要产生一个服从于$[0,1]$范围内均匀分布的随机数。对于连续时间马尔科夫链模型来说，除了决定下个状态之外，在任意状态的停留时间也需要通过随机数来产生。

产生的随机路径通过随机路径验证模块进行验证。模块的返回结果是一个布尔值，表示该条路径是否满足给定公式。验证算法采用一种递归验证的策略。

验证器模块负责管理随机路径的产生与验证，是验证层负责对外的门面类。上层通过该模块传递验证层所需的参数。在对参数进行必要的有效性验证后，验证器模块会调用随机路径产生模块与验证模块，并对结果进行统计。

由于需要采用不同的随机路径产生算法，所以在实现上对算法也做了一定的抽象，程序单独抽象出了随机路径产生算法接口类，好处是保证了程序的鲁棒性以及灵活性。算法接口类的示意图如下所示。

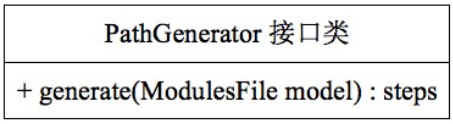

图 12 随机路径产生算法接口类

Figure.12 interface for class generating random path

最后，样本权重计算模块负责计算每个样本对应的可靠性权重。在验证层完成所有的工作后，会将每个样本的参数值，模型检验值以及对应的可靠性权值打包成数组传递给 BP 神经网络模块用于回归分析。

4.2.3 BP 神经网络模块

BP 神经网络位于软件处理的最后一层。输入层主要负责接受用户的输入并对输入进行解析和预处理。验证层主要负责对模型以及公式进行验证并产生拟合所需的训练数据，BP 神经网络模块则主要负责消费这些数据对神经网络模型进行训练，并预测模型在未知点满足公式的概率。

4.2.4 小结

本章主要介绍了软件的组织架构并对各个模块进行了功能性分析，从输入和输出对模块之间的关系进行了梳理，下一章将会结合案例分析 SpacNN 的使用。

第五章 案例分析

在本章中，我们使用第四章中介绍的模型检验工具 SpacNN 对一个案例进行分析，以此说明在工具在验证真实案例时的效果。

5.1. 概率广播算法

5.1.1 模型介绍

概率广播算法[41]是一种通信协议（communication protocol），用于描述多个计算机之间的通信过程，属于广播算法的一种。它由简单的泛洪算法改进而来。在一个由采用泛洪（flooding）算法[42]的计算机网络中，每一台计算机在接受到消息后，都会将该信息以广播（broadcast）的形式发送到所有的邻居节点。概率广播算法是在简单的泛洪算法下进行的改进。假设在一个计算机网络中，存在一个消息源节点（source node），它首先将消息通过泛洪（flooding）的方式传递给它的所有的邻居节点，然后进入休眠模式。其他所有的节点在接收到消息后，将随机地进行下列两种行为中的一件：

1. 以$psend$的概率将接收到的消息进行转发（forward），然后进入休眠状态
2. 以概率$1-psend$的概率不发送消息，然后进入休眠状态。

这样做的目的是在保证了网络的稳定性的前提下减少了带宽的浪费。可以看到，在$psend$等于 1 的时候，上述算法等同于泛洪算法。现在考虑一个 3×3 的网络拓扑结构，以下为示意图：

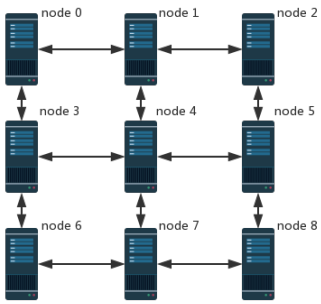

图 13 3×3 网络拓扑图

Figure.13 the topology of a network that is 3×3

如图所示，节点 0 是源消息节点，在起始状态节点 0 会向直接相邻节点发送消息，即节点 0 会向节点 1 和节点 3 发送消息，同理，节点 1 在收到消息后，也只会向节点 0，节点 2 和节点 4 发送消息。节点 0 在发送完消息后随即进入休眠状态。节点 1 在接收到消息后，会选择转发或是不转发这条消息，转发的概率为 $psend$。而不管是转发还是不转发，节点 1 都会进入休眠状态。

另外，我们假设网络中的通信是同步的，即消息的接收和发送都由一个外部时钟控制，即，所有在同一时刻接收到消息的节点将同时选择是否转化消息，并在同一时刻进行转发。

上述结构中的节点 1 的 PRISM 模型如下所示：

```
// 非初始节点在接收到消息后进行转发的概率
const double psend; // 模型参数
// 节点 1 模块
module node1
active1:[0..1] init 1; // 表示节点 1 是否处于激活状态
send1:   [0..1] init 0; // 表示节点 1 是否转发消息
[tick] active1=1 & send1=0 & send0+send2+send4 >=1 -> psend:(active1'=1)&(send1'=1)+(1-psend):(active1'=0)&(send1'=0);
[tick] active1=1 & send1=0 & send0+send2+send4 <1 -> (active1'=1)& (send1'=0);
[tick] active1=1 & send1=1 -> (send1'=0)& (active1'=0);
[tick] active1=0 -> (send1'=0)& (active1'=0);
endmodule
```

可以看到，节点 1 包含两个变量，即$active1$和$send1$，前者表示节点 1 是否处于激活状态，处于激活状态表示节点 1 尚未决定是否转发消息；后者表示节点 1 是否已经转发消息，$send1$为 1 表示节点 1 选择转发消息，为 0 表示节点 1 忽略消息。之后的四条转移关系分别表示如下含义：

1. 当节点 1 处于激活状态且未转发消息时，此时如果接收到任何来自于节点 0，节点 2 或者节点 4 的消息，则以概率$psend$选择转发消息，而以概率$1-psend$选择忽略。

2. 当节点 1 处于激活状态且未转发消息，当此时未接收到来自节点 0，节点 2 或者节点 4 的消息，则继续保持当前状态直到下一个时钟周期。

3. 当节点当前处于激活状态并且已经转发了来自相邻节点的消息时，此时节点 1 进入休眠状态，并置$send1$为 0。

4. 当节点 1 处于休眠状态，保持此状态不变。

当考虑到网络中可能发生的冲突和延时时，上述模型将发生改变。考虑网络中发生的冲突。网络中发生冲突是指在网络中，同时存在多个节点向同一个节点发送消息，对于上述节点 1，即$send0+send2+send4>0$，此时节点 1 将接收不到任何有用的消息，故选择忽略。而当考虑节点的延时时，这里使用简单的无记忆延时模型（memory less delay），即当节点接收到任意一个消息时，它发生延时的概率为某个固定值，即其不受当前节点在此之前接收的消息的延时情况的影响。

考虑到上述两点特性且修改之后的 PRISM 模型如下所示。

```
// 节点 1 在仅接收到来自节点 0 或者节点 3 的一条消息时发生延时的概率
const double pdelay =0.5;
module node1
// 节点 1 所处的状态，0:休眠状态；1:转发状态；2:延时状态
active1:[0..2] init 1;
// 节点 1 是否已转发消息，1:已转发；2:未转发
send1:   [0..1] init 0;
[tick] active1=1 & send1=0 & send0+send2+send4 =1 -> (1-pdelay)* psend: (active1'=1)&(send1'=1) +(1-pdelay)*(1-psend):(active1'=0)&(send1'=0) + pdelay: (active1'=2)&(send1'=0);
[tick] active1=2-> (1-pdelay)*psend: (active1'=1)&(send1'=1) +(1-pdelay)*(1-psend):(active1'=0)&(send1'=0)+ pdelay: (active1'=2)&(send1'=0);
```

```
[tick] active1=1 & send1=0 & send0+send2+send4 !=1 -> (active1'=1)& (send1'=0);
[tick] active1=1 & send1=1 -> (send1'=0)& (active1'=0);
[tick] active1=0 -> (send1'=0)& (active1'=0);
endmodule
```

可以看到，修改之后的节点 1 模型除了休眠状态和激活状态之外，还增加了一个延时状态；除此之外，当节点 1 的邻居节点中有超过一个节点发送消息给节点 1 时，节点 1 将忽略来自两者的消息并保持原状态。当节点 1 只接收到来自节点 0，节点 2 或者节点 4 的一个消息时，将以概率 pdelay=0.5 发生延时，或以概率 1-pdelay=0.5 的概率决定是否转发，其中转发的概率依旧是 psend。最后，当模型处于已发送消息的状态或者是休眠状态时，转化的关系同无冲突和无延时的模型一致。

下面我们将给出运用第三章介绍的方法对概率广播算法进行验证的情况。我们将从以下几个角度对验证结果进行分析：

1. 评估基于对偶路径的 SMC 算法的正确性。
2. 对比常规 SMC 算法和基于对偶路径的 SMC 算法的估计值的方差。
3. 评估基于对偶路径路径在不同的参数下的估计值的方差。
4. 对比加权神经网络和未加权神经网络在曲线拟合方面的差异。

5.1.2 基于对偶路径的算法正确性评估

我们对如下性质进行检测，即“网络在 10 个单位时间内消息发送到节点 8 的概率”。

在 SpacNN 中，上述公式可以如下表示

$$P_{=?}[true\ U^{\leq 10} active8 = 0]$$

注意，从节点 0 发出的消息最终到达节点 8 当且仅当节点 8 的状态为休眠状

态，即$active8 = 0$。

我们分别使用标准 SMC 算法以及基于对偶路径的 SMC 算法对模型满足上述性质的概率进行估计，分别运行 10 次的计算结果如下表所示：

表 2 标准 SMC 算法与基于对偶路径 SMC 算法的正确性

标准 SMC 算法	误差	基于对偶路径的 SMC 算法	误差
0.45667	-0.01866	0.47333	-0.00199
0.52667	0.05134	0.43333	-0.04199
0.46667	-0.00863	0.43333	-0.04199
0.41333	0.01199	0.41333	-0.06199
0.50667	-0.06199	0.49333	0.01800
0.50667	0.03134	0.52	0.04467
0.50333	0.03134	0.5	0.02467
0.46333	0.02800	0.4667	-0.00865
0.49	0.01467	0.44	-0.03532
0.48333	0.00800	0.5	0.02467

通过上式我们可以看出标准 SMC 算法和基于对偶路径的 SMC 算法均可以给出近似正确的结果。

5.1.3 标准 SMC 算法和基于对偶路径的 SMC 算法的估计值方差对比

本节主要对两种算法的估计值的方差进行比较。

首先来看样本内部相关系数。假设样本的大小为偶数，即$n = 2k$。则样本内部相关系数定义为

$$I_S = I([S_1, S_3 \dots, S_{2k-1}], [S_2, S_4, \dots, S_{2k}])$$

由相关系数的概念可知，任意的S_i和S_{i+1}对的结果互不相同的组数越多，上述定义的相关系数的值越小。

下面，我们分别使用标准 SMC 算法和基于对偶路径的 SMC 算法取得 100 个大小为 300 条随机路径的样本对参数$psend = 0.8$的情形下进行分析，统计样

本内部相关系数的直方图如下图所示。

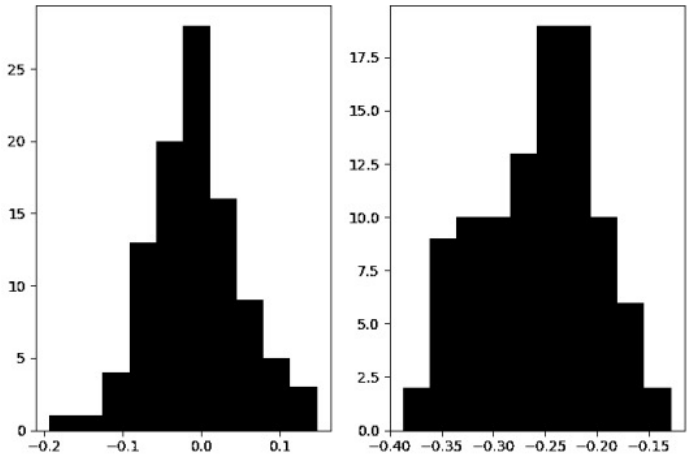

图 14 标准 SMC 算法和对偶路径 SMC 算法的样本内部相关系数直方图

Figure.14 histgram for standard SMC's and antithetic path based SMC's results

从上述直方图我们可以看出，对于标准 SMC 算法所取得的样本，其内部相关系数一般在 0 附近，这意味着其线性相关性基本为 0。而对于对偶路径的 SMC 算法，由于采用了状态重排序算法，使得互为对偶路径的两条路径之间的验证结果相反的概率大大提高，应该其样本内部相关系数得以减小，其均值在-0.30 到-0.25 之间，且所有样本的内部相关系数均小于 0。

下面来看两种算法所产生的估计值的方差的对比。本实验进行 100 次方差计算，每次方差计算产生 10 个样本，每个样本的大小为 20 条随机路径，参数值为 $psend = 0.8$。绘制出的直方图如下所示。

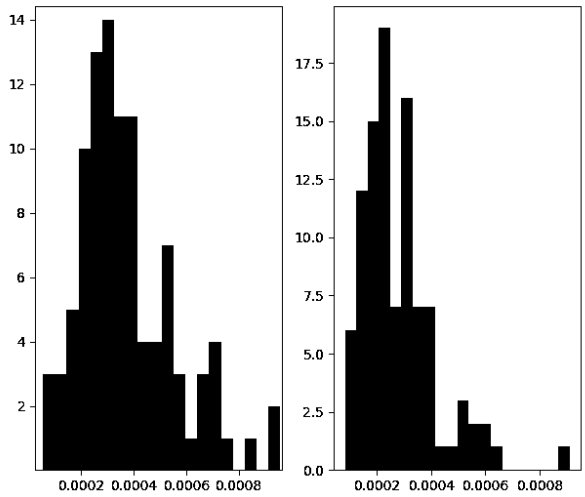

图 15 标准 SMC 算法和对偶路径 SMC 算法的估计值方差直方图

Figure15. histgram for variance of prediction of standard SMC and antithetic path based SMC

可以看出，标准 SMC 算法的估计值的方差总体分布于 0.002 到 0.008 之间，且分布较为均匀，通过进一步的计算可得方差小于 0.004 的比例为 56%；而对于基于对偶变量的 SMC 算法，有将近 92%的数据分布在 0.004 以下的区间，并且，方差数小于 0.002 的频数也远远小于标准 SMC 算法。由此可见，基于对偶路径的 SMC 算法的确可减小估计值的方差，进而提高估计值的准确率。

5.1.4. 加权神经网络与非加权神经网络的拟合效果对比

最后一节将验证使用加权 BP 神经网络对带参数马尔科夫链模型的满足函数进行拟合的结果，并将这一结果与普通 BP 神经网络拟合的结果进行对比，比较两者的差异。

在概率广播模型中，参数$psend$的取值范围为$[0,\ 1]$，这里取参数$[0, 0.02, 0.04, \ldots, 1]$作为训练参数点，在这些参数点上运行对偶路径 SMC 算法，然后根据第三章权重计算方法，首先分段计算各段样本点的可靠性权重，结果如下图所示：

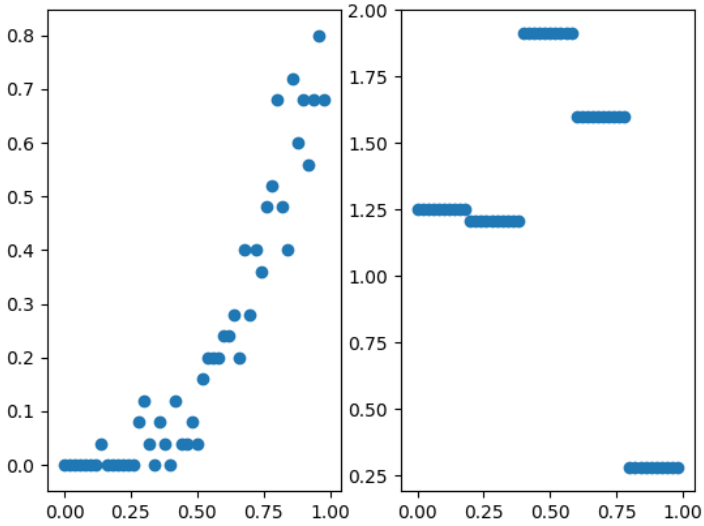

图 16 样本点分段权重示意图

Figure16. reliability weights for sample

上图中左图为样本点的分布图，右为各段样本点的可靠性权重。从上述示意图可以看出，除去[0,0.2]之外，各段的权重大致和各段内点到拟合直线的平均距离成反比。这里对[0, 0.2]区间段进行了特殊处理，即为了防止该区间段的权重过大（即样本点到拟合直线的距离过小），令该区间段的权重为后续区间段的权重的平均值。使用第三章中介绍的训练方法对加权 BP 神经网络进行训练，并利用训练好的神经网络模型对系统在位置参数点下的模型检测值进行估计，得到如下结果：

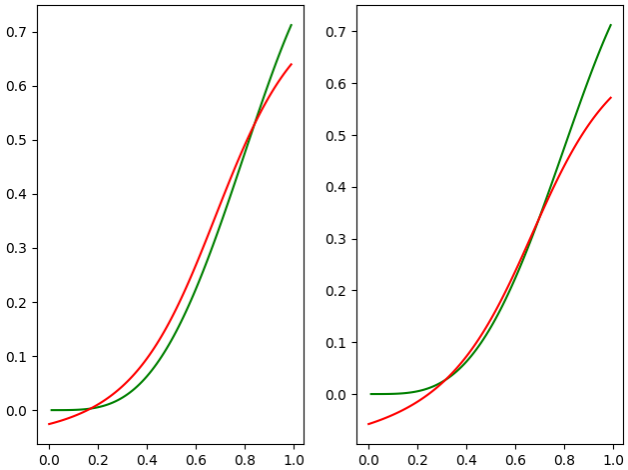

图 17 神经网络拟合满足曲线结果对比图

Figure.17 comparison of BP neural network and weighted BP neural network prediction

上图展示了使用神经网络对训练数据进行拟合的结果，其中绿色曲线展示了使用 PRISM 进行验证的真实数据，红色曲线为拟合出的曲线。其中，左边是未加权的 BP 神经网络的结果，右边是加权神经网络的拟合结果。可以看出，经过加权的神经网络由于在中间段（即[0.25, 0.75]）的权重较大，所以拟合出的曲线在该段更贴合训练数据，即在该段与真实的曲线更接近。

由此我们可以结论，加权神经网络通过分段对训练数据赋予不同的可靠性权重，使得可靠性较高的数据具有更高的权重，从而使得拟合曲线与可靠性高的训练数据更贴合，从而可以有效减小拟合曲线与真实曲线的误差。

为了进一步比较普通 BP 神经网络与加权 BP 神经网络的拟合效果的对比，我们分别使用两种拟合方法对 BP 神经网络进行了 100 次训练，并计算了经过训练的神经网络的预测值与真实值的平均误差，统计成直方图如下所示：

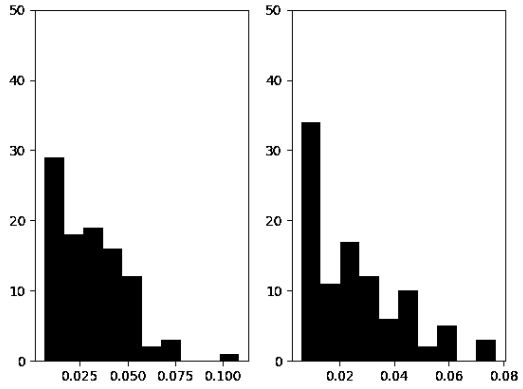

图 18 神经网络拟合平均误差统计对比图

Figure.18 comparison of normal BP neural network and weighted BP neural network regression

观察上图可以发现，加权神经网络的拟合误差更低。通过进一步统计两种拟合误差的分位数可以清晰的看到这一点，以下是两种拟合情况的误差的分位数表:

表 3 普通神经网络与加权神经网络拟合误差分位数表

	25%分位数	50%分位数	75%分位数	90%分位数
未加权	0.01453	0.02794	0.04095	0.05146
加权	0.00998	0.02226	0.03453	0.04698

上表表明，加权神经网络的拟合误差总体更小。

第六章 总结与展望

6.1 总结

本文主要研究了带参数的离散时间马尔科夫链模型的模型检测问题。通过分析模型的满足函数的连续性我们将模型检测问题转化为函数拟合问题。本质上我们需要对满足函数进行拟合。为了拟合满足函数，我们使用改进的 BP 神经网络，为每个样本添加了可靠性权重，并给出了度量可靠性权重的方法。为了估计模型在确定参数下的模型检测值，我们采用统计模型检测方法，并借鉴蒙特卡罗方法中的方差减小技巧，即对偶变量取样，提出了 DTMC 模型中的对偶状态与对偶路径的概念，并证明了在对状态进行一定规则的重排序的基础上，对偶路径之间的验证结果往往不同，进而得出结论即，对偶路径 SMC 方法可以有效减小估计值的方差。

我们根据以上算法实现了带参数离散时间马尔科夫链模型的检测工具 SpacNN，第四章中分别针对工具的架构，实现原理，关键算法和简单使用做了介绍。

在案例研究阶段，我们研究了概率广播协议，并利用 PRISM 建立了相应的模型，使用我们在第三章中介绍的方法，我们论证了算法的正确性，改进的 SMC 算法相对于标准 SMC 方法的优势以及 BP 神经网络的拟合效果。

6.2 展望

带参数的模型在现实生活中无处不在，有的不能用离散时间马尔科夫链模型进行建模。目前我们提出的方法只适用于 DTMC，后续将研究适用于 CTMC 乃至更多模型类型的模型检测工具。

参考文献

[1] Lecca, Paola, and C. Priami. "Cell Cycle Control in Eukaryotes: A BioSpi model." *Electronic Notes in Theoretical Computer Science*180.3(2007):51-63.

[2] Duflot, Marie, et al. "A formal analysis of bluetooth device discovery." *International Journal on Software Tools for Technology Transfer*8.6(2006):621-632.

[3] Kwiatkowska, Marta, G. Norman, and D. Parker. "Stochastic model checking." *International Conference on Formal Methods for PERFORMANCE Evaluation* Springer-Verlag, 2007:220-270.

[4] Kwiatkowska, Marta Z., G. Norman, and D. Parker. "Probabilistic Symbolic Model Checking with PRISM: A Hybrid Approach." *International Conference on TOOLS and Algorithms for the Construction and Analysis of Systems* Springer-Verlag, 2002:52-66.

[5] Bianco, Andrea, and L. D. Alfaro. "Model checking of probabilistic and nondeterministic systems." *International Conference on Foundations of Software Technology and Theoretical Computer Science* Springer Berlin Heidelberg, 1995:499-513.

[6] 侯刚等. "模型检测中状态爆炸问题研究综述." *计算机科学* 40.s1(2013):77-86.

[7] Legay, Axel, B. Delahaye, and S. Bensalem. "Statistical Model Checking: An Overview." 6418.2(2010):122-135.

[8] Sen, Koushik, M. Viswanathan, and G. Agha. *Statistical Model Checking of Black-Box Probabilistic Systems*. *Computer Aided Verification*. Springer Berlin Heidelberg, 2004.

[9] Rubinstein, Reuven Y. *Simulation and the Monte Carlo Method*. *Simulation and the Monte Carlo method.*. John Wiley & Sons, 2008:167-168.

[10] Nicholas Metropolis, and S. Ulam. "The Monte Carlo Method." *Journal of the American Statistical Association* 14.1(1971):151-167.

[11] Kwiatkowska, Marta, et al. *Symbolic Model Checking for Probabilistic Timed Automata*. *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*. Springer Berlin Heidelberg, 2004:430--440.

[12] Kwiatkowska, Marta, et al. "Symbolic model checking for probabilistic timed automata ☆." *Information & Computation* 205.7(2007):1027-1077.

[13] Norman, Gethin, and V. Shmatikov. *Analysis of Probabilistic Contract Signing*. *Formal Aspects of Security*. Springer Berlin Heidelberg, 2003:81-96.

[14] Aizatulin, Mihhail, H. Schnoor, and T. Wilke. "Computationally Sound Analysis of a Probabilistic Contract Signing Protocol." 5789(2009):571-586.

[15] Islam, S, and M. A. Zaid. "Probabilistic analysis of the ASW protocol using PRISM." *Southeastcon* IEEE, 2008:159-164.

[16] Lakin, M. R., et al. "Design and analysis of DNA strand displacement devices using probabilistic model checking." *Journal of the Royal Society Interface* 9.72(2012):1470-1485.

[17] Dannenberg, Frits, et al. *DNA Walker Circuits: Computational Potential, Design, and Verification*. *DNA Computing and Molecular Programming*. Springer International Publishing, 2013.

[18] Kwiatkowska, Marta, G. Norman, and D. Parker. "Using probabilistic model checking in systems biology." *Acm Sigmetrics Performance Evaluation Review* 35.4(2008):14-21.

[19] Brown, Peter F., et al. "The mathematics of statistical machine translation: parameter estimation." *Computational Linguistics*19.2(1993):263-311.

[20] Tarantola, Albert. "Inverse Problem Theory and Methods for Model Parameter Estimation." *Society for Industrial & Applied Mathematics Philadelphia Pa* (2005):xii,342.

[21] Nicholas Metropolis, and S. Ulam. "The Monte Carlo Method." *Journal of the American Statistical Association* 14.1(1971):151-167.

[22] Opper, and Manfred. *Advanced mean field methods :*. MIT Press, 2001.

[23] Andreychenko, Aleksandr, et al. "Parameter identification for Markov models of biochemical reactions." *International Conference on Computer Aided Verification* Springer-Verlag, 2011:83-98.

[24] Bortolussi, Luca, et al. "Continuous approximation of collective system behaviour: A tutorial ☆." *Performance Evaluation* 70.5(2013):317-349.

[25] Dimitriadis, E I, et al. "Parametric Study and Improvement of the Electrical Characteristics of a-SiC/c-Si(p) Based, Thyristor Like Switches, Using Two Dimensional Simulation Techniques." *Active & Passive Electronic Components* 10.3-4(2015):283-312.

[26] Wang, S. S. J., and M. P. Wand. "Using Infer.NET for Statistical Analyses." *American*

Statistician 65.2(2011):115-126.

[27] Bortolussi, Luca, D. Milios, and G. Sanguinetti. "Smoothed model checking for uncertain Continuous-Time Markov Chains ☆." *Information & Computation* 247(2016):235-253.

[28] Glasserman, Paul. *Monte Carlo Methods in Financial Engineering*. Springer, 2004.

[29] Hastings, W. K. "Monte Carlo sampling methods using Markov chains and their applications." *Biometrika* 57.1(1970):97-109.

[30] Manly, Bryan F. J. *Randomization, Bootstrap and Monte Carlo Methods in Biology*. *Randomization, bootstrap and Monte Carlo methods in biology*. Chapman & hall, 1997:856-856.

[31] Gadzhiev, Ch. M. "Checking multivariate model fit from the generalized Wishart-statistic variance." *Measurement Techniques* 36.12(1993):1316-1319.

[32] Legay, Axel, B. Delahaye, and S. Bensalem. "Statistical Model Checking: An Overview." 6418.2(2010):122-135.

[33] 盛骤. *概率论与数理统计:第三版*. 高等教育出版社, 2001.

[34] 徐钟济. *蒙特卡罗方法*. 上海科学技术出版社, 1985.

[35] 尹增谦等. "蒙特卡罗方法及应用." *物理与工程* 12.3(2002):45-49.

[36] 寻广彬等. "非线性最小二乘跟踪的对偶变量变分方法." *力学学报* 48.5(2016):1202-1207.

[37] 李晓莉, and 张雅文. *概率论与数理统计*. 高等教育出版社, 2014.

[38] Hornik, Kurt, M. Stinchcombe, and H. White. "Multilayer feedforward networks are universal approximators. " *Neural Networks* 2.5(1989):359-366.

[39] 吴岸城. *神经网络与深度学习*. 电子工业出版社, 2016.

[40] 焦李成. *神经网络系统理论*. 西安电子科技大学出版社, 1990.

[41] Fehnker, Ansgar, and P. Gao. "Formal Verification and Simulation for Performance Analysis for Probabilistic Broadcast Protocols." *Lecture Notes in Computer Science* 4104(2006):128-141.

[42] Qayyum A, Viennot L, Laouiti A. Multipoint Relaying for Flooding Broadcast Messages in Mobile Wireless Networks[C]// Hawaii International Conference on System Sciences. IEEE, 2002:3866-3875.

致谢

两年半的研究生活转瞬即逝，纵使有万千的舍不得还是不得不说再见。研究生生活是艰苦的，在第一个学期完成学业之余，便投入了科研之中。从最初的懵懂无知，一头乱撞，到如今的按图索骥，游刃有余，不可以不说是自己的努力加上同学的帮助。尤其是我的导师张敏，她不光教我在课题研究的道路上如何沉着应对，更教我在人生的道路上的许多道理。

我还要感谢同门师兄郭延楠以及师姐吕悦，是你们的帮助和指点迷津，让我的科研的道路上越走越好。

最后，要感谢我的恋人金畅和母亲，是你们朝夕的陪伴让我最终完成了研究生生涯。