2018 届研究生硕士学位论文 学校代号: 10269
学号: 51151500019

華東師範大學

基于分布式统计模型检测的异构系统验证方法

院　　系: 计算机科学与软件工程学院
专 业 名 称: 软件工程
研 究 方 向: 可信软件
指 导 教 师: 杜德慧 副教授
硕士研究生: 姜 凯强

2018 年 5 月

2016 MASTER'S DISSERTATION School Code: 10269
Student Number: 51131500003

EAST CHINA NORMAL UNIVERSITY

Statistical Model Checking Based on Abstraction and Learning

Department:	School of Computer Science and Software Engineering
Major:	Software Engineering
Research Direction:	Trustworthy Software
Supervisor:	Associate Professor Dehui Du
Candidate:	Bei Cheng

May, 2016

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《基于抽象和学习的统计模型检测研究》，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

《基于抽象和学习的统计模型检测研究》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，本论文的研究成果归华东师范大学所有。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，并向主管部门和相关机构如国家图书馆、中信所和“知网”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）。

程贝 硕士学位论文答辩委员会成员名单

姓名	职称	单位	备注
缪淮扣	教授	上海大学	主席
胡豪东	高级工程师	中航工业航空动力控制系统研究所	
陈铭松	教授	华东师范大学	

摘　要

信息物理融合系统（Cyber Physical Systems，CPS）是一种更关注计算机与物理环境交互和协作的高级嵌入式系统，自 2006 年此概念被提出以来，已受到了学术界与工业界的高度关注。第一，CPS 应用大都安全攸关或功耗要求严苛，在保证功能的前提下，仍必须满足一定的非功能属性，如吞吐量、能耗等，因此需要验证分析以保证其可信性；第二，CPS 大都是异构的混成系统，融合了连续的物理过程和离散的系统行为，且处于高度不确定的开放环境中，因此使用传统的方法（如模型检测和定理证明）难以完成验证分析。为缓解此问题，人们开始尝试使用统计算法对系统模型的仿真 Trace 进行分析，求得近似结果，并给出结果的误差范围，这种方法被称为统计模型检测（Statistical Model Checking，SMC）。SMC 无需遍历状态空间，但当结果精度要求较高时需要产生大量 Trace（多数仿真软件的 Trace 生成比较耗时），性能因此大大降低，本文即针对 SMC 的性能问题展开深入研究。

首先，对已有 SMC 算法的原理进行了剖析，实现了 4 种 SMC 算法，通过大量实验给出了详细的性能评估。基于实验结论，提出了一个自适应的 SMC 算法框架，以根据不同属性的预估概率，动态地选择合适的 SMC 算法。

其次，为改进自适应的 SMC 中贝叶斯区间估计算法的不足，提出了基于抽象和学习的 SMC 方法，旨在减少统计分析所需的 Trace 数量以提高 SMC 的效率。其中结合已有的抽象和学习理论（如主成分分析、随机文法推断），对随机混成自动机的仿真 Trace 进行了概率等价抽象和简化；并基于抽象 Trace 学习出概率等价的系统行为模型——前缀频率树，同时提出了树的两阶段约减算法，以有效控制树的规模，为更高效的 SMC 验证分析提供了良好的抽象模型。

最后，介绍了我们实现的 CPS 建模分析平台——Modana，基于此平台实现了本文提出的 SMC 改进算法，基于 Modana 平台建模分析了典型的 CPS 系统——智能温控系统；并结合 3 个基准测试案例，对 SMC 算法改进前后的性能和准确度进行了实验性评估。结果证明，本文提出的 SMC 改进方法正确并且有效。

关键词： 信息物理融合系统；随机混成自动机；主成分分析；统计抽象；统计模型检测

ABSTRACT

Cyber Physical Systems (CPS) are advanced embedded systems concerning more the interaction and collaboration between computer and physical environment. Since 2006 when this concept was presented, they have been highlighted by both academic and industrial worlds. First, most CPS applications are safety-critical or limit demanding energy consumption; a number of non-functional requirements (e.g. throughput, energy consumption, etc) need to be met when functional ones have been guaranteed, so that they required to be checked to achieve trustworthy systems. Second, most CPS are heterogeneous hybrid systems which combine continuous physical process and discrete system behavior, and also are exposed to open environment of high uncertainty; so traditional methods (model checking and theorem proving) can hardly finish checking effectively. To mitigate this issue, statistical methods are used to analyze the traces drawn from system simulator, by which an approximate result are obtained with an error bound. This method is known as Statistical Model Checking (SMC) which does analysis without traversing the state space of systems. However, SMC with high precision usually consumes a large number of traces (generating traces is seriously time-consuming for most simulation softwares), which leads to poor performance. This paper intensely studies the performance issue of SMC.

First of all, we gives an insight into the theory of existing SMC algorithms and implement four of them for conducting large numbers of experiments of their performance in detail. Based on our conclusion, an adaptive SMC algorithm framework is presented to automatically choose appropriate SMC algorithms according to the estimated proba-

bility of properties in different cases.

Next, to overcome the shortcoming of Bayesian Interval Estimate in the adaptive SMC, we present an SMC method based on abstraction and learning, aimed at improving the efficiency of SMC via reducing the number of traces for statistical analysis. This method uses the existing related learning theories (e.g. principal components analysis and stochastic grammar inference) to abstract and simplify probabilistically equivalent traces of Stochastic Hybrid Automata. Then we learn the probabilistically equivalent system behavior model, i.e. prefix frequency tree with abstracted traces, and effectively control the size of the tree by two-phase reduction algorithms presented also in this paper. It provides a well abstract model for more efficient verification and analysis with SMC later.

Finally, we introduce Modana Platform for modeling and analysis of CPS, which is implemented by our team. Based on Modana, we further implements the improved SMC presented in this paper. And a typical CPS application - smart heating system is modeled and analyzed. Then we experimentally evaluate the performance and accuracy of both original and improved SMC algorithms with three benchmarks. It turns out that our method is correct and efficient.

Keywords: *Cyber physical systems, Stochastic hybrid automata, Principal components analysis, Statistical abstraction, Statistical model checking*

目录

第一章　绪 论

1.1　研究背景及意义

信息物理融合系统（Cyber Physical System，CPS）是一种复杂的异构系统，具有以下两个特征：1) 面向开放环境，存在大量不确定因素，如天气突变、信号误差、人为失误等，都需要在设计 CPS 时予以考虑，以保障 CPS 在未知环境下的可信性；2) 除了计算机外，还结合了机械、环境、土木、电子、生物、化学、航空等诸多工程领域的模型和方法，且各领域不是简单地关联，而是深度的融合。因此，由于 CPS 的不确定性、异构性及连续性，使 CPS 的建模分析及验证面临巨大的挑战，已有的 CPS 研究已经针对 CPS 的各个性质做了大量的工作，并相应的有了一些工具的支持，例如基于时间自动机理论的 UPPAAL，基于马尔科夫模型的 Prism 以及对物理系统提供较好的建模仿真支持的 Modelica、Simulink 等等。然而，以上工作在解决 CPS 的建模验证及仿真问题上各有优势及不足。本文首先提取出信息物理融合系统的信息部分（cyber part）、物理部分（physical part）以及需求约束（constraint），之后对信息部分和物理部分分别用合适领域的建模工具进行建模，同时将需求约束抽象成验证属性（property），其次我们将针对建好的多个模型设计 master 算法进行协同仿真（Co-simulation），并生成协同仿真的迹（trace），最后将生成的协同仿真的迹输入到我们的验证器（checker）并使用分布式统计模型检测算法进行评估分析。本方法可以很好的结合多种建模仿真工具的的建模优势，从而更好的支持对 CPS 系统的建模、仿真和定量评估。

1.2 国内外研究现状

1.2.1 CPS 的形式化建模

对 CPS 进行验证分析的基础是形式化建模。模型无论在系统设计还是系统分析中都处于核心地位，良好的模型可以为一个复杂的系统提供合适的抽象，并有助于理解系统行为的本质，但 CPS 的内在异构性使其很难拥有一个完美的建模范式。目前常见的模型如下：1. Ptolemy II 是由加州大学伯克利分校的 Edward A. Lee 教授团队完成，着重于解决异构系统建模、设计和仿真问题的强大工具。其基于 Actor 模型，为多种不同的计算模型（Models of Computations，MoCs）提供了一种强语义，以达到在一个完整系统中融合多种 MoCs 的目标。因此，Ptolemy 方法特别适用于大型的异构 CPS。2. 随机混成自动机（Stochastic Hrbrid Automata，SHA）可视作混成自动机的扩展。由于 CPS 与控制论同源，而混成系统又适合于刻画一般控制系统，所以扩展的混成自动机也可用来建模 CPS。在混成自动机中，常微分方程（Ordinary Differential Equation，ODE）可包含在状态中，用来表示系统停留在该状态时所进行的连续变化过程，状态之间的跳转仍然为离散行为。SHA 大体可分为两类，一类是仅仅将随机行为引入离散的状态跳转中，如 Piecewise-deterministic Markov processes；另一类则更加复杂，还将随机过程引入到了连续变化中，如中提出的随机混成系统。特别的，UPPAAL-SMC 将代价时间自动机网（Network of Priced TimedAutomata，NPTA）中的时钟变化率扩展为非常数形式，其表达能力等价于第一类 SHA。3. Petri 网是另一种常见的性能分析模型，适合描述同步并发系统，因其丰富直观的表达风格而得到广泛应用。目前 Petri 网也已有针对 CPS 的特点进行扩展的版本。

1.2.2 分布式技术 (Distributed technology)

分布式计算技术是计算机发展过程中产生的一项科学技术，主要工作原理是通过多台计算机的分布式连接实现数据的综合处理，旨在通过多台计算机的强大的工作能力来分解复杂问题，解决一些计算难题。分布式计算技术的具体特征表现

如下：首先，分布式计算能够合理分配计算内容，实现多台计算机共同工作，节约设备成本，提高工作效率。其中最核心的内容在于能够为计算程序寻找最合适的计算机来完成工作。目前，计算机领域内关于分布式计算的技术已有数百种之多，但多数并没有密切的联系，这种缺乏系统管理和统一行业规定的技术并不利于日后的广泛发展。另外，分布式计算技术主要是通过科学算法的研究，形成一种独特的计算模型，确保其超长的数据处理能力，这种发展规律导致大多数用户只单纯研究如何集结更多闲置计算机来完成实际数据的处理，并没有考虑如当某些计算机丧失处理能力后的数据归属问题。那么，就要求研究者对分布式计算技术进行更加深入、系统的研究，目前，通过虚拟网络运营机制来实现大批量数据的共同处理以及如何实现用户间数据的高速共享以初具规模。如何更大规模的集结剩余计算力量、如何科学系统的管理共享数据资源、如何更大程度的节省计算资源成本成为当今社会研究分布式计算技术的重要课题。

1.2.3 协同仿真 (Co-simulation)

由于 CPS 包含信息和物理两个部分，并涉及各个领域，因此，对于 CPS 的各个部分的建模在不同的领域都有相应的工具及方法支持。如果将 CPS 各个部分联系在一起进行仿真分析通常有两种方法: 1) 开发一个统一的 CPS 建模平台，将 CPS 的所有相关部分都在此平台建模仿真分析 2) 将 CPS 的各个部分在不同的工具中建模，使用协同仿真技术 [22,23] 联合 CPS 的各个部分进行仿真。方法一到目前为止实现起来较为复杂，所以通常采用第二种方法较多，但协同仿真技术在实际的仿真中需要消耗较多的时间，针对这一问题，我们之前提出了 [24] 有效的提高了协同仿真的效率。

1.2.4 统计模型检测 (Statistical Model Checking)

人们利用统计知识来分析系统由来已久，SMC 技术即建立在蒙特卡洛模拟、假设检验等统计方法之上，通过统计分析系统仿真运行的 Trace 来验证系统属性满足的情况。它最早被 Sen 等人提出用来验证黑盒系统 [?]，即 Single Sampling Plan

（SSP）算法的雏形，其难点在于确定算法收敛所需的 Trace 总样本数量 N 以及接受原假设的阈值 C；Younes 等人在博士论文 [?] 中提出一种二叉搜索的算法来近似得到所需的 n 和 c，并指出了 [?] 中验证黑盒系统方法的一些错误。基于 Wald 的 Sequential Probability Ratio Test（SPRT）[?] 原理，Younes 等人还提出了基于对数的 SPRT 实现算法 [? ?]，用以验证系统；该方法可以最小化算法所需 Trace 的样本数量。SSP 与 SPRT 用以解决定性验证问题，回答了“系统 S 满足属性 ϕ' 的概率是否大于或等于某个概率阈值 θ”这个问题，即 $S \models P_{\geq\theta}(\phi')$。与定性算法不同，定量算法可以直接返回 S 满足 ϕ' 的概率 p，如 Approximate Probabilistic Model Checking（APMC）[?]，通过计算 x/n 来计算 p（其中 x 和 n 分别表示所需的 Trace 正样本数量和总数量），并通过 Chernoff-Hoeffding 界来限定结果的误差范围。随后，Zuliani、Clarke 等人基于贝叶斯统计又提出了两个新 SMC 实现算法：Bayesian Hypothesis Testing（BHT）和 Bayesian Interval Estimation（BIE）[? ?]，前者基于贝叶斯假设检验解决定性验证问题，后者基于贝叶斯区间估计解决定量评估问题。

从 SSP、SPRT、APMC 到 BHT、BIE，SMC 算法所需 Trace 数量逐渐减少，效率逐渐提高。为了进一步探索 SMC 技术，许多人将数值方法与统计方法相结合 [? ?]，来进一步提升 SMC 效率或解决一些 SMC 难以应付的问题，如非确定性问题（Non-determinism）。研究 SMC 非确定性算法的还有 Henriques 等人，其博士论文讨论了如何用概率方式近似解决非确定性问题 [?]。SMC 善于验证有界（通常指时间约束，即 time-bounded）的属性，因此传统模型检测中的无界“Until”属性的 SMC 验证方法也是研究热点之一，He、Jennings 等人提出了一种将无界“Until”验证转化为有界“Until”的方法以解决此问题 [? ?]。由于 SMC 基于系统仿真结果，所以也不可避免地引入了仿真领域的问题，比如小概率事件在 Trace 中出现概率极低，使得验证过程需要产生 Trace 的数量过多而效率低下。Jegourel、Legay 等人基于重要性取样（Importance Sampling）和重要性分割（Importance Splitting）技术提出了面向小概率属性验证的 SMC 算法 [? ?]，大大减少了验证所需的 Trace 样本数量；解决类似问题还可以借助于机器学习技术，如 [?] 借助支持向量机预测事件，[?] 借助

贝叶斯推断预测事件，都可以提高 SMC 的效率。除此之外，SMC 还有一些特殊的应用场景，如黑盒系统 [?]、异构系统 [? ?]，允许在系统内部结构和行为未知的条件下分析系统。

Ymer[?] 和 Vesta[?] 是最早实现 SMC 的验证工具。Vesta 采用了极易实现并行化的 SSP 算法的一个变种，Ymer 采用的 SPRT 算法很快也被 Younes 证明同样能够被并行化。Ymer 在实验中的效率高于 Vesta；此外，Vesta 还支持了无界“Until”的验证。目前最流行的支持 SMC 的验证工具则是 UPPAAL-SMC[?] 和 Prism[?]，UPPAAL-SMC 和 Prism 都实现了定性的 SPRT 算法，同时都实现了基于置信区间（Confidence Interval，CI）[?] 的定量算法（文献 [?] 对不同版本的 CI 统计算法进行了对比，并根据需求的不同，给出了方法选择的建议）。UPPAAL-SMC 的建模基于 PTA 或 SHA，使用图形化建模，用户友好，对于时间和连续行为的支持较好；Prism 则使用 Reactive Modules Language（RML）建模，对随机（如马尔科夫链）和非确定性（如马尔科夫决策过程）模型的验证支持较好。Plasma Lab[?] 是新出现的一款纯 SMC 工具，同样支持 RML 建模，并实现了多种 SMC 算法（包括重要性取样和分割等面向小概率事件的算法）。Plasma Lab 允许用户以插件集成的方式为其添加新的模型输入和验证算法，如 Simulink 的集成。

1.3 本文技术路线及主要研究内容

信息物理融合系统是异构系统，本文针对这种异构系统的验证提出了一种解决方案，图1.1为本文的技术路线图，本文的技术路线大致如下：

（1）我们通过对该系统进行分析，提取出该系统的信息部分（Cyber part）和物理部分（Physical part），除此之外我们根据自己需要验证的行为属性定义约束（Constraint）。

（2）使用 SysML[?] 建模语言对提取出的信息部分和物理部分进行建模，将信息系统和物理系统的组件使用 SysML 的 BDD（SysML Block Definition Diagram ）图进行建模，同时使用 SysML 的 IBD（SysML Internal Block Diagram）图来描述系统

中各个组件之间的关联。

（3）SysML 只是用来建模组件内部结构和组件之间的关联，该模型不可直接进行仿真运行，因此我们将 SysML BDD 图建模的模型使用 FMU 进行实现，同时将 SysML IBD 图描述的系统组件关系转化为 FMU 之间相互依赖的接口配置文件，此时，我们只需要再设计好协同仿真的主算法（Master Algorithm）就可以进行异构系统的协同仿真。然而，在进行协同仿真之前，我们首先要保证各个 FMU 之间的协同是正确的，要确保 FMU 之间协同行为的正确性，我们需要验证主算法的正确性及各个 FMU 之间的连接顺序及数据交换的正确性。在本文中，我们基于时间自动机设计了一个协同行为正确性验证的验证器，我们将系统的多个 FMU、协同仿真的主算法以及 FMU 之间的接口配置文件输入到该协同行为的验证器之中即可验证当前系统协同行为的正确性，如果验证通过则说明我们当前的模型即为正确的模型，如果验证不通过，则需要修改当前系统的协同行为，直到得到验证通过的模型之后再输入到仿真器中进行仿真。

（4）我们在进行系统的验证分析时，首先需要验证的系统模型，同时我们还需要验证的属性（Property），我们将（1）中得到的约束进行形式化描述，即可得到验证属性（该验证属性根据约束的不同可以是 BLTL/ALTL/GSCL 等等）。

（5）将通过第三步验证的模型（Verified Model）及第四步得到的验证属性输入到异构系统验证器（co-verification）之中进行验证分析，首先将模型输入到仿真器（Simulator）之中进行仿真或协同仿真 (Co-simulation) 并得到仿真迹（Traces），然后将得到的仿真迹和验证属性输入到模型验证器（Checker）之中来验证该迹是否满足某条特定的验证属性，得到结果满足为 1，不满足为 0，我们将该验证是否满足的结果称为观察值（Observations），多条仿真迹对于一条特定的验证属性会得到多个观察值。最后将得到的观察值输入到统计分析算法中进行统计分析，并得到评估结果。

本文的具体研究内容和贡献点总结如下：

1. 使用 SysML 建模语言建模整个系统的架构，使用 SysML 的 BDD 图建模系统

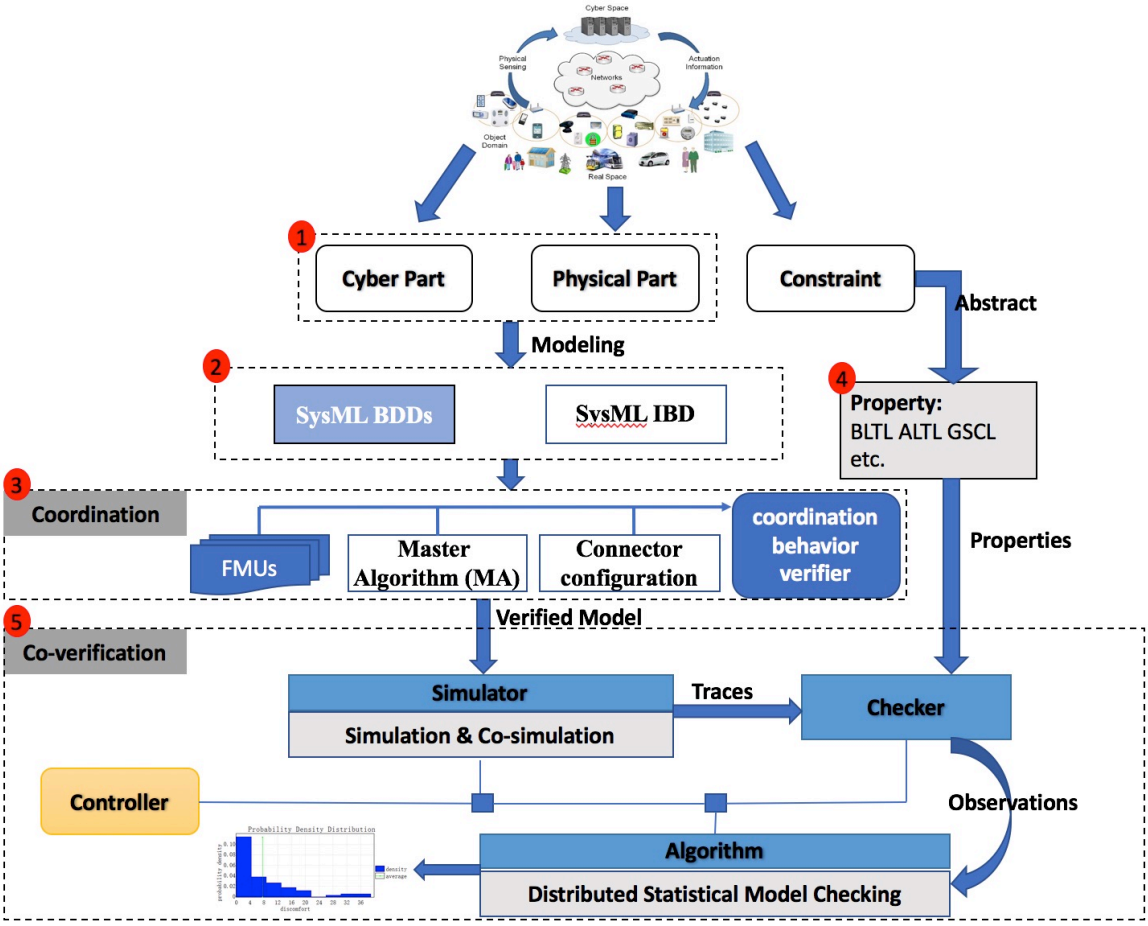

Cyber Space
Physical Sensing
Networks
Actuation Information
Object Domain
Real Space
1
Cyber Part
Physical Part
Constraint
Abstract
Modeling
2
SysML BDDs
SysML IBD
4
Property:
BLTL ALTL GSCL
etc.
3
Coordination
FMUs
Master Algorithm (MA)
Connector configuration
coordination behavior verifier
Properties
Verified Model
5
Co-verification
Simulator
Simulation & Co-simulation
Traces
Checker
Controller
Observations
Algorithm
Distributed Statistical Model Checking

图 1.1: 论文技术路线图

组件，使用 SysML 的 IBD 图来描述系统中各个组件的关联关系。

2. 基于 FMI 标准实现 SysML 描述的系统模型，将 SysML 的 BDD 图建模的系统组件包装成 FMU，并将各个组件的关联关系转化为 FMU 之间的接口配置文件。

3. 使用时间自动机理论验证了基于 FMI 标准的多个组件的协同行为，使用时间自动机将协同仿真的主算法进行形式化描述，并使用 UPPAAL [?] 模型检测器来验证主算法的正确性；提出了一种从 FMU 到时间自动机的映射标准，通过此标准用时间自动机将多个 FMU 进行编码，并用时间自动机之间的通道（channel）来描述多个 FMU 之间的关联关系，最终将多个 FMU 及 FMU 之间的关联关系使用一个时间自动机网络进行了描述，将该时间自动机网络输入到 UPPAAL 之中进行验证从而来验证协同行为的正确性。

4. 将验证属性用 BLTL/ALTL/GSCL 等形式化语言进行描述。

5. 提出了一种基于抽象和学习的分布式统计模型检测算法，大大提高了统计模型检测的效率。

1.4 本文组织结构

本文共分七章，组织结构如下：

第一章介绍了本文的研究背景及意义，并从四个方面阐述了该研究领域的国内外研究现状，其中包括信息物理融合系统的形式化建模、分布式技术、协同仿真及统计模型检测的研究现状。之后，给出了本文的技术路线和主要贡献点。最后，总结了论文组织结构。

第二章介绍了相关预备知识。首先给出了信息物理融合系统的主要概念，并详细讨论了我们本文关注的信息物理融合系统的异构性；之后，给出了概率有界线性时态逻辑和时间自动机的形式化描述，并给出了基于 FMI 标准的协同仿真通用接口。

第三章介绍了基于时间自动机理论来验证异构系统协同行为正确性的方法，首先我们给出了该方法的技术框架，之后我们详细描述了如何用时间自动机理论来验证协同仿真的主算法，以及如何验证整个异构系统的协同行为的正确性。

第四章重点阐述了如何用统计模型检测算法来对异构系统进行验证分析，也是本文的主要内容。首先，介绍了如何基于 FMI 标准对异构协同进行协同仿真并得到仿真迹，然后提出了一种基于抽象和学习的统计模型检测算法来提高统计模型检测的效率。最终，我们将协同仿真和该高效的统计模型检测算法进行结合，以此来对异构系统进行验证分析。

第五章主要介绍工具及程序实现。首先简单介绍了我们自己的 CPS 建模分析平台——Modana，之后给出了基于 Modana 平台的异构系统验证工具（Co-SMC 工具）。最后，给出了 Co-SMC 工具的详细设计及程序实现。

第六章给出了两个案例，通过使用本文提出的方法对这两个案例进行建模、仿真和分析来验证本文提出方法的有效性。

第七章为总结和展望，总结了本文提出的基于分布式统计模型检测的异构系统验证方法，并讨论了其优点和不足，指出了未来要进一步进行研究的工作。

1.5 本章小结

本章首先说明了选题的背景和意义，指出了由于 CPS 的异构性而导致 CPS 系统的验证分析面临巨大挑战；接着介绍了信息物理融合系统的形式化建模、分布式技术、协同仿真及统计模型检测的研究现状；最后给出了本文的技术路线、主要贡献点和组织结构。下一章将介绍本文涉及到的预备知识及概念。

第二章　预备知识与概念

2.1　信息物理融合系统及异构性

CPS 这一概念最早由美国基金委的 Helen Gill 于 2006 年提出，近几年已成为国际学术交流的热点之一，如领域内最重要的 CPS Week，集合了四大 CPS 的学术会议（HSCC、ICCPS、IPSN 及 RTAS）、20 个 Workshops 等一系列平台，以推进 CPS 在学术和工业界的发展。美国国家科学基金会资助了 CPS 在嵌入式、信息安全等多方面的研究，已取得重要成果；2007 年，美国总统科学技术委员会将 CPS 列为八大关键技术之首。欧洲、日本、韩国也开始逐步重视对 CPS 相关领域的基础研究。CPS 在国内的研究也已起步，2010 年，国家 863 计划信息技术领域办公室和专家组在上海举办了“信息物理融合系统发展战略论坛”，并对这项极具前景的技术给予了高度关注。周巢尘、何积丰院士等人对 CPS 不变式等方面的基础研究做出了巨大贡献，为国内各个研究机构（如西北工业大学、南京大学、华东师范大学等）的研究工作奠定了坚实的基础。

2.2　FMI 功能模型接口及 FMU

CPS 中各个组件之间的协同可以使用基于 FMI 标准的协同仿真来实现，FMI 标准最初是在 2008 年开始的 MODELISAR 项目中开发的，并得到大量软件公司和研究中心的支持。FMI 支持模拟由异构组件组成的复杂系统，通过一个协同仿真环境将不同模型与自己的求解器耦合起来。实现了 FMI 标准接口的系统组件被称为 FMU，下面我们给出 FMU 的语法和语义。

定义 2.2.1. ***FMU* 语法**

FMU 的语法可以用一个八元组 $F = (S, U, Y, D, s_0, set, get, doStep)$ *表示,*

- S 表示 FMU 的状态集合。
- U 表示 FMU 的输入变量集合。
- Y 表示 FMU 的输出变量集合。
- $D \subseteq U \times Y$ 表示多个 FMU 之间输入和输出之间的依赖关系集合。$(u, y) \in D$ 表示输出变量 y 直接依赖于输入变量 u 的取值。
- $s_0 \in S$ 表示 FMU 的初始状态。
- $set : S \times U \times \mathbb{V} \to S$ 表示 set 函数对一个输入变量进行赋值。给定当前状态 $s \in S$, 输入变量 $u \in U$, 及一个数值 $v \in \mathbb{V}$, 该函数将返回一个新的状态，此状态的 u 的值为 v.
- $get : S \times Y \to \mathbb{V}$ 表示 get 函数返回某个输出变量的数值。给定一个状态 $s \in S$ 和一个输出变量 $y \in Y$, $get(s, y)$ 返回 s 状态上 y 输出变量的取值。
- $doStep : S \times \mathbb{R}_{\geqslant 0} \to S \times \mathbb{R}_{\geqslant 0}$ 表示 $doStep$ 函数进行了一步仿真。给定当前状态 s, 和一个非负实数 $h \in \mathbb{R}_{\geqslant 0}$, $doStep(s, h)$ 返回一个集合 (s', h')，且
  当 $h' = h$ 时, F 接受并执行步长 h 并且迁移到了一个新的状态 s';
  当 $0 \leqslant h' < h$ 时, F 拒绝步长 h, 只执行步长 h', 并且迁移到一个新的状态 s'.

定义 2.2.2. ***FMU* 语义**

给定一个 FMU $F = (S, U, Y, D, s_0, set, get, doStep)$, FMU 的执行依赖于 $doStep$ 函数, FMU 的执行可以用一个时间输入序列（Timed Input Sequence, TIS）进行描述。

TIS 是一个有限的四元组序列 (t, s, v, v'), $t \in \mathbb{R}_{\geqslant 0}$ 表示当前时刻, $s \in S$ 表示 F 的一个状态, v 是一个输入赋值, $v' : Y \to \mathbb{V}$ 是一个输出赋值。

TIS = $(t_0, s_0, v_0, v'_0), (t_1, s_1, v_1, v'_1), (t_2, s_2, v_2, v'_2), ..., (t_i, s_i, v_i, v'_i), (t_{i+1}, s_{i+1}, v_{i+1}, v'_{i+1}), ...$
定义如下:

- $t_0 = 0$ 时刻的 s_0 状态表示 F 的初始状态。
- 对于任意的 $i \geqslant 1, t_i = t_0 + \sum_{k=1}^{i} h_k$
- 给定当前状态 s_i, set 函数用来将当前状态的输入参数设置为一个特定的数值 v. 之后 F 执行 $doStep$ 函数并且迁移到一个新的状态 s_i'. get 函数用来得到当前状态的所有输出参数值 v_i'.

因此, FMU 的语义可以用一个标签迁移系统进行描述。

2.3　时间自动机

时间自动机 [?] 是一个建模实时系统行为的经典理论模型。它提供了一种用许多实值时钟标注状态转换图的有效方法。在本小节中，我们来回顾一下时间自动机的语法和语义。

定义 2.3.1. 时间自动机语法

时间自动机可以用一个四元组 $A = (L, l_0, E, I)$ 来表示, 其中:

- L 表示时间自动机中有限的位置集合;
- $l_0 \in L$ 为时间自动机的初始位置;
- 约束集合 $G(x)$ 可以用 $g = x \bowtie c \mid g \wedge g$ 来表示, 其中 $x \in X$, $c \in \mathbb{N}$ 且 $\bowtie \in \{<, \leqslant, \geqslant, >, =\}$.
- $E \subseteq L \times G(X) \times Act \times 2^X \times L$ 是包含约束和时钟的一组边的集合, 其中 $Act = Act_i \cup Act_o$. Act_i 是一个输入动作的集合且 Act_o 是一个输出动作的集合。
- $I : L \to G(X)$ 将不变量指定给位置。

定义 2.3.2. 时间自动机语义

时间自动机 $A = (L, l_0, E, I)$ 的语义可以用一个标签迁移系统 $L_A = (Proc, Lab, \{\xrightarrow{\alpha} \mid \alpha \in Lab\})$ 进行描述, 其中:

- $Proc = \{(l, v) \mid (l, v) \in L \times (X \to \mathbb{R}_{\geqslant 0})$ 且 $v \models I(l)\}$, 其中，状态是一个 (l, v) 元组, l 是时间自动机中的位置且 v 是满足 $I(l)$ 的一个时钟变量;
- $Lab = Act \cup \mathbb{R}_{\geqslant 0}$ 是一个标签集合；且
- 迁移关系定义如下:

  $(l, v) \xrightarrow{\alpha} (l', v')$，如果存在一个边 $(l \xrightarrow{g,\alpha,r} l') \in E$, 则 $v \models g$, $v' = v[r]$ 且 $v' \models I(l')$

  $(l, v) \xrightarrow{d} (l, v + d)$，对于所有的 $d \in \mathbb{R}_{\geqslant 0}$, 则 $v \models I(l)$ and $v + d \models I(l)$

对于时间自动机 A 和其中某个位置 l 的可达性问题就是一个判断在迁移系统 L_A. 中是否存在一个从初始状态 (l_0, v_0) 到状态 (l, v) 的路径。为了验证需要，我们定义了时间自动机的符号语义。该定义用到了包含一组时钟的执行序列集合。

对于一个特定的位置 l，特定的时刻 $t \in X$，对于任意的 $x \in X$, 则 $t + x \in X$. 从该时刻位置开始的执行序列如下所示:

$(l, t) \xrightarrow{x_1} (l, t + x_1) \xrightarrow{x_2} (l, t + x_1 + x_2) \xrightarrow{x_3} (l, t + x_1 + x_2 + x_3) \xrightarrow{x_4} ... \xrightarrow{x_i} (l, t + x_1 + x_2 + x_3 + ... + x_i) \xrightarrow{x_{i+1}} ...$

其中 $x_i > 0$ 且无穷序列 $x_1 + x_2 + ...$ 对于 x 是收敛的。

2.4　概率有界线性时态逻辑

概率有界线性时态逻辑（Probabilistic Bounded Linear Temporal Logic，PBLTL）公式可以用来形式化的描述系统的验证属性。首先，我们定义用来验证单条仿真迹的有界线性时态逻辑（Bounded Linear Temporal Logic，BLTL）的语法和语义，然后将其扩展为 PBLTL。

给定一个模型 M，设其状态变量的集合 SV 是一个有限的实数集。在 SV 上的一个布尔谓词约束为 $y \sim v$ 的形式，其中 $y \in SV$，$\sim \in \{\geq, \leq, =\}$，且 $v \in \mathbb{R}$。BLTL 的语法定义如下：

$$\varphi ::= y \sim v \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \neg\phi_1 \mid \phi_1 U^{\leq t} \phi_2$$

其中 $\sim \in \{\geq, \leq, =\}$，$y \in SV$，$v \in \mathbb{Q}$，$t \in \mathbb{Q}_{\geq 0}$。为方便使用，我们可以定义一些辅助的时序逻辑算子，例如“F”算子，$F^{\leq t}\phi = TrueU^{\leq t}\phi$，表示最终在 t 时间内存在 ϕ 满足；又如”G” 算子，$G^{\leq t}\phi = \neg F^{\leq t}\neg\phi$，表示在 t 时间内 ϕ 始终满足。对于一条迹 σ，σ^k 表示这条迹的后缀，即 σ 从第 k 步开始执行的部分（σ^0 则表示原始迹）。我们规定 $V(\sigma, k, y)$ 表示迹 σ 在第 k 步时状态变量 y 的值，t_k 表示第 k 步的时间，t 表示最终的时间约束，则 BLTL 在迹 σ^k 上的语义可以如下定义：

定义 2.4.1. （有界线性时态逻辑的语义）.

- $\sigma^k \vDash y \sim v$ 当且仅当 $V(\sigma, i, y) \sim v$。
- $\sigma^k \vDash \phi_1 \vee \phi_2$ 当且仅当 $\sigma^k \vDash \phi_1$ 或者 $\sigma^k \vDash \phi_2$。
- $\sigma^k \vDash \phi_1 \wedge \phi_2$ 当且仅当 $\sigma^k \vDash \phi_1$ 并且 $\sigma^k \vDash \phi_2$。
- $\sigma^k \vDash \neg\phi_1$ 当且仅当 $\sigma^k \vDash \phi_1$ 不成立。
- $\sigma^k \vDash \phi_1 U^t \phi_2$ 当且仅当存在一个自然数 i 使得

  1) $\Sigma_{0 \leq l < i}\, t_{k+1} \leq t$;

  2) $\sigma_{k+i} \models \phi_2$;

  3) $\sigma_{k+j} \models \phi_1$ 对于每个 $0 \leq j \leq i$。

定义 2.4.2. （概率有界线性时态逻辑）.

一个 *PBLTL* 属性公式 ϕ 表示为 $P_{\geq\theta}(\phi')$ 的形式，其中 ϕ' 是一个 *BLTL* 公式，θ 是一个介于 *0* 和 *1* 之间的阈值。我们定义模型 M 满足 *PBLTL* 属性 $P_{\geq\theta}(\phi')$，表示为 $M \models P_{\geq\theta}(\phi)$，即模型 M 的仿真迹满足 *BLTL* 属性 ϕ' 的概率不小于 θ。而对于定量分析的 *PBLTL* 属性公式，则无需指定阈值 θ，所以可表示为 $P_{=?}(\phi')$。

2.5　本章小结

本章首先给出了信息物理融合系统的定义，并介绍了本文涉及到的一个重要概念——CPS 的异构性，指出对异构系统的验证分析面临巨大的挑战。之后，介绍了实现异构系统各个组件之间协同运行的标准，即 FMI 标准和 FMU 的语法及语义；同时，给出了用于验证组件之间协同行为正确性的理论模型-时间自动机的语法和语义。最后，给出了描述本文的验证属性主要用到的逻辑公式 PBLTL 的语法和语义。下一章我们将介绍如何建模整个异构系统的架构以及如何基于时间自动机理论来验证异构系统中各个组件协同行为的正确性。

第三章　异构系统协同行为的正确性验证

在本章节，我们首先用 SysML 建模语言对整个异构系统的架构进行建模，其中用 SysML 的 BDD 图来建模系统的组件，用 SysML 的 IBD 图来建模系统中各个组件之间的关联关系。由于 SysML 只是用来建模系统的架构，该建模语言得到的模型不可执行。因此，我们基于 FMI 标准，将 SysML 的 BDD 描述的组件用 FMU 进行实现，将 SysML 的 IBD 图转化为 FMU 之间的接口配置文件。因此，我们只需要设计协同仿真主算法就可以完成异构系统基于 FMI 的协同仿真。然而，在我们将基于 FMI 的可仿真模型输入到协同仿真引擎之中进行仿真之前，我们需要确保该模型的协同行为的正确性，即要保证该模型可以正确的进行协同仿真。为了解决这个问题，我们需要进行两个方面的验证:（1）我们要验证协同仿真主算法的正确性，即算法没有出现死锁以及其他一些特性；（2）我们要验证系统之间各个组件之间关联顺序及数据交互的正确性，其中最主要的要保证多个 FMU 没有出现环路依赖。由于 FMU 的执行时基于时间的，而时间自动机在建模时间有较大的优势，且时间自动机的验证有着良好的工具支持，因此，在接下来的内容中，我们重点描述如何基于时间自动机理论来验证整个可仿真模型的协同行为的正确性。在第一小节，我们给出整个验证过程的技术框架；第二小节介绍如何对上述的第一个方面进行验证，即协同仿真的主算法的验证；在第三小节中，我们使用一个具体的案例来描述如何对上述的第二个方面进行验证，即异构系统协同行为的验证。

3.1　技术框架

图3.1为异构系统协同行为正确性验证的技术框架图. 首先，我们在建模（Modeling）阶段使用 SysML 的 BDD 和 IBD 图来建模整个异构系统的架构。SysML 的

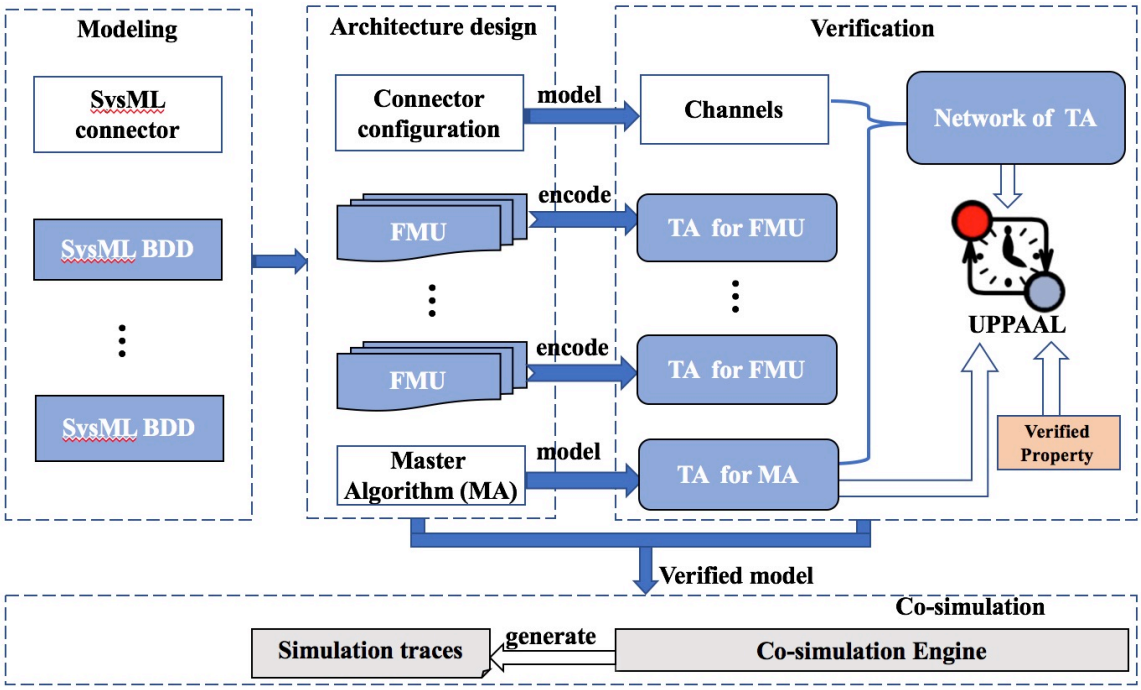


图 3.1: 协同行为正确性验证技术框架图

BDD 中的块建模了异构系统的一个个组件；SysML 的 IBD 图描述了各个块之间的连接关系。为了借助协同仿真技术对整个系统进行协同仿真，我们在架构设计（Architecture design）阶段将一个个块用 FMU 进行实现，同时将 IBD 图描述的关联关系转化为一个 FMU 之间的配置文件（Connector configuration）。接下来，我们自己设计了协同仿真的主算法（Master Algorithm，MA）来实现各个 FMU 之间的交互，同时来驱动协同仿真过程的执行。接下来是协同行为的验证（verification）阶段，也是我们本文的主要贡献点之一。为了验证协同行为的正确性，（1）我们先验证了协同仿真主算法的正确性：首先，我们用时间自动机对协同仿真的主算法进行形式化建模，然后将该时间自动机模型输入到 UPPAAL 模型检测工具中进行验证（2）我们验证了整个系统协同行为的正确性：我们提出了一种从 FMU 到时间自动机的映射规则，我们根据此规则将 FMU 映射为时间自动机，接下来将 FMU 之间的配置文件转化为时间自动机的通道（Channels）。这样，我们就得到了一个时间自动机网络（Network of TA）：包括 FMU 映射成的时间自动机、主算法的时间自动机及各个时间自动机之间的通道）。最后我们将该时间自动机网络和要验证的属

性（死锁、可达性、活性等）输入到 UPPAAL 中来验证该模型是否满足某些属性。一旦验证了协同行为的正确性，我们就可以将通过验证的模型输入到协同仿真的引擎之中进行协同仿真并得到协同仿真的迹。接下来，我们将详细介绍整个协同行为的验证过程。

3.2　协同仿真主算法的验证

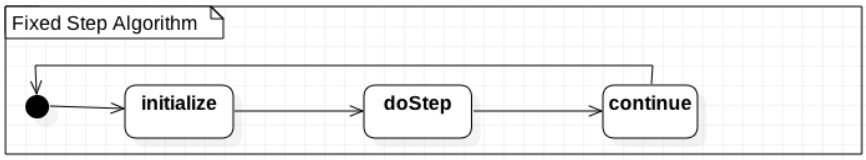
在本小节我们首先介绍了三种协同仿真的主算法：固定步长算法、可回滚算法及可预测步长算法，之后我们用时间自动机形式化建模了这几种算法，最后将这几种算法的形式化模型输入到 UPPAAL 工具中，分别验证了这几种算法的有无死锁、可达性和活性的属性。

3.2.1　协同仿真主算法介绍

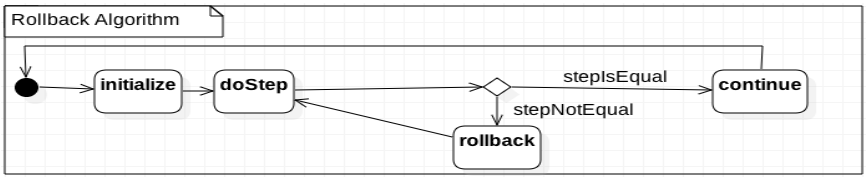
协同仿真的主算法用了调度和协同异构系统的多个 FMU 组件的执行。每一个 FMU 都可以被看做一个可独自仿真的黑盒，但是多个 FMU 之间在某些特定的时刻需要进行数据交互和同步。图3.2描述了当前三种主要的协同仿真主算法的活动图。在接下来的内容中，我们对这三种算法进行简单的介绍。

固定步长算法

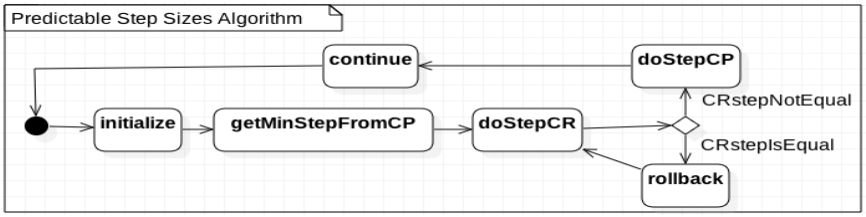
对于固定步长算法 [**?**]，所有的 FMU 都有一个相同的步长。当协同仿真主算法在 t 时刻调用 $doStep$ 函数执行步长 h 时，所有的 FMU 将从 t 时刻执行 h 步长并到达 $t+h$ 时刻。在执行下一个 $doStep$ 函数之前，要确保所有的 FMU 都执行完了上一个步长并且完成了数据交换。固定步长算法的活动图如图3.2(a)所示，该活动图主要包含三个活动：$initialize$, $doStep$ 和 $continue$，首先对所有的 FMU 进行初始化，之后调用 $deStep$ 函数进行仿真，最后在 $continue$ 活动中完成 FMU 的数据交换。在固定步长算法中，只要保证每个 FMU 的仿真是可靠的，则可以保证整个仿真过程的可靠性。但是，如果在某个 FMU 的仿真过程中出现错误，则会导致整个协同仿真过程出错。为了克服固定步长算法的缺陷，出现了可回滚的算法。



(a) 固定步长算法



(b) 可回滚算法



(c) 可预测步长算法

图 3.2: 三种协同仿真主算法的活动图

可回滚算法

FMI2.0 相对于 FMI1.0 添加了几个重要的特性，它支持了对于 FMU 状态的保存和回滚。例如，主算法调用 FMU_1 和 FMU_2 的 $doStep$ 函数执行 h 步长，但是 FMU_1 接受了该步长且 FMU_2 拒绝了该步长，则 FMU_1 和 FMU_2 都会执行 h 步长然后回滚到上一个时间点。可回滚算法的活动图如图3.2(b)所示，相比固定步长算法，可回滚算法要求所有的 FMU 支持 $rollback$ 方法, 且当某个 FMU 拒绝该步长时，所有的 FMU 将会回滚到上一个步长执行完之后的时间点。

可预测步长算法

如果可以预测 FMU 下一步能执行的最大步长（FMU 在不错过事件时，能执行的仿真的最长时间），则可以大大提高协同仿真的效率。因此，论文 [**?**] 提出了 $GetMaxStepSize$ 函数来预测 FMU 下一步能执行的最大步长，有了该函数的支持，就出现了可预测步长算法。图3.2(c)为可预测步长算法的活动图，首先，所有的 FMU

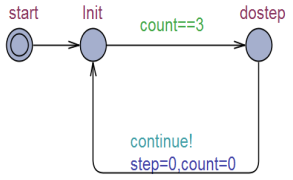
进行初始化，然后支持 $GetMaxStepSize$ 函数的 FMU 在 $getMinStepFromCP$ 活动中预测他们下一步能执行的最大步长，并且在所有的最大步长中选择最小的一个 h 作为所有 FMU 下一步执行的步长，之后所有的 FMU 执行 h 步长，如果所有的 FMU 都接受了该步长，则所有的 FMU 仿真该步长然后进行下一步。如果有 FMU 拒绝了该步长，则将所有的 FMU 回滚到上一个时间点，然后选择一个更小的步长进行仿真。

3.2.2　协同仿真主算法的建模和验证

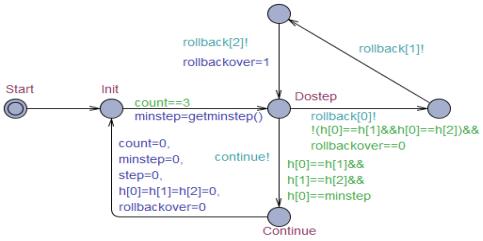
我们用时间自动机将三种不同类型的主算法进行形式化建模，图3.3是三种主算法的时间自动机模型。固定步长算法的时间自动机模型包含 $Init$ 和 $doStep$ 两个状态，且与 FMU 通过 $continue$ 信道进行通信。可回滚算法包括 $Init$、$DoStep$ 和 $Continue$ 三个状态。如果所有的 FMU 都接受了下一步要进行仿真的步长，则可回滚算法对应的时间自动机将发送 $continue$ 信号，且迁移到 $Continue$ 状态；否则，将发送 $rollback$ 信号，且返回到上一个状态。可预测步长算法包括 $Init$, $find\_CP\_MIN$, $DoStep$, $writeCP$ 四个状态。首先他们获得支持预测步长算法的多个 FMU 的最大步长，然后取所有最大步长的最小值 $step1$ 。然后执行该步长，如果所有的 FMU 都接受则发送 $continue$ 信号并迁移到下一个状态，否则发送 $rollback$ 信号并回滚到 $DoStep$ 状态。

UPPAAL 是基于时间自动机理论对实时系统进行验证的工具，其中使用到的时间自动机模型增加了整型变量、多种数据类型及同步信号等扩展。我们使用 UPPAAL 工具验证了这三个模型的可达性、活性及死锁。具体的验证属性如下所示：

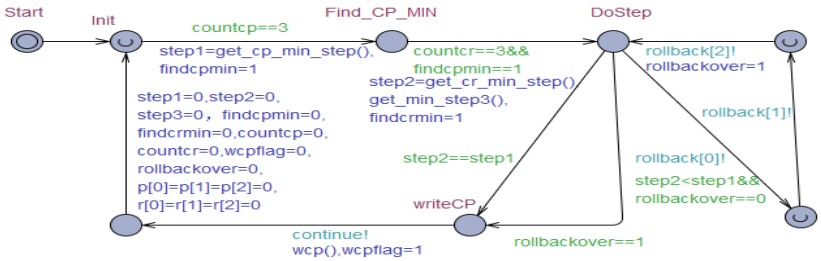
- $E\langle\rangle master.dostep$, $E\langle\rangle master.Continue$ and $E\langle\rangle master.writeCP$ 是可达性验证，用来验证这些系统状态是否可达；
- $master.Init->master.dostep$, $master.Init->master.Continue$ and $master.Init->master.Continue$ 是活性验证。如果主算法可以到达前一个状态，那么它最终也会到达后一个状态。



(a) 固定步长算法的时间自动机模型



(b) 可回滚算法的时间自动机模型



(c) 可预测步长算法的时间自动机模型

图 3.3: 三种不同主算法的时间自动机模型。

- $A[]notdeadlock$ 是死锁的验证, 用来验证主算法是否存在死锁。

实验的结果如表3.1所示，从表中我们可以看出不存在死锁，且可达性和活性都满足，说明我们的主算法是正确的。例如：属性 $A[]\ not\ deadlock$ 满足，说明主算法不存在死锁；属性 $E\langle\rangle\ master.doStep$ 满足，说明系统最终会到达 $doStep$ 状态。总结来说，我们在这一小节验证了我们所用到的协同仿真的主算法的正确性。

3.3　异构系统协同行为的验证

在本小节，我们使用一个水箱的案例对整个协同行为的验证过程进行详细的描述。由于，在验证过程中，需要用时间自动机对 FMU 进行形式化描述，我们首先给出了从 FMU 到时间自动机的映射规则，然后我们使用 SysML 对整个系统进行架构设计，之后基于 FMI 标准对系统的各个组件进行实现，并给出多个 FMU 之间的连接关系配置，最后我们用时间自动机建模了 FMU 的形式化模型，并输入到 UPPAAL 工具中针对验证属性进行验证分析。

表 3.1: 主算法的验证实验结果

主算法	验证属性	结果
固定步长	$A[]\ not\ deadlock$	True
	$master.Init- > master.dostep$	True
	$E\langle\rangle\ master.dostep$	True
可回滚	$A[]\ not\ deadlock$	True
	$master.Init- > master.Continue$	True
	$E\langle\rangle\ master.Continue$	True
可预测	$A[]\ not\ deadlock$	True
	$master.Init- > master.writeCP$	True
	$E\langle\rangle\ master.writeCP$	True

3.3.1 FMU 到时间自动机的映射

在本文的第二章的预备知识中，我们给出了 FMU 和时间自动机的语法及语义，下面我们根据第二章的语法语义给出 FMU 和时间自动机的映射关系。我们发现 FMU 和时间自动机的语义之间存在一定的关联。FMU 的语义关注于 FMU 的执行序列，也就是状态随着时间的不断迁移；因此，时间自动机的执行迹跟 FMU 的执行序列十分相似，也是状态随着时间的迁移序列。因此，我们使用时间自动机来对 FMU 进行形式化描述，从而来分析多个 FMU 之间的协同行为。给定一个 FMU$F = (S, U, Y, D, s_0, set, get, doStep)$, 我们可以根据他们执行语义之间的关联，将 FMU 用时间自动机 $A = (L, l_0, E, I)$ 进行形式化描述：

- L 是时间自动机的有限位置集合。因此，时间自动机语义模型 L_A 中的状态可以看做为 FMU 中的状态，即 $(l, v) \Rightarrow s$。
- 时间自动机语义模型 L_A 中的初始状态可以看做为 FMU 中的初始状态, 即 $(l_0, v_0) \Rightarrow s_0$。
- FMU 中的输入变量 $u \in U$ 可以看做为时间自动机的 $Act_i \cup \{absent\}$。
- FMU 的输出变量 $y \in Y$ 可以看做为时间自动机的 $Act_o \cup \{absent\}$。

- 时间自动机的输入动作 $e \in Act_i$ 可以看做 FMU 中的 set 函数。
- 时间自动机的输出动作 $e \in Act_o$ 可以看做为 FMU 中的 get 函数。
- 时间自动机之间依靠 channel 的同步可以看做为 FMU 之间的依赖关系。$(u, y) \in D$ 表示输出变量 y 依赖于输入变量 u，在时间自动机中输出动作同样依赖于输入动作。
- 对于时间自动机，一个动作 $e \in Act$ 会触发一个迁移 $s \xrightarrow{e} s'$, 这个过程就相当于 FMU 里面的 $doStep$ 执行，会使其到达下一个状态。例如：时间自动机的迁移 $l \xrightarrow{e} l'$ 可以描述 FMU 的 $doStep(s, h)$ 被调用，并且此 FMU 接受了步长 h 并到达了下一个状态 s'；然而，此 FMU 也可能会拒绝步长 h, 并且发生了回滚，这个过程在时间自动机里面可以用一条边 $l' \xrightarrow{e} l$ 来进行描述，来表示回滚到了上一个状态。

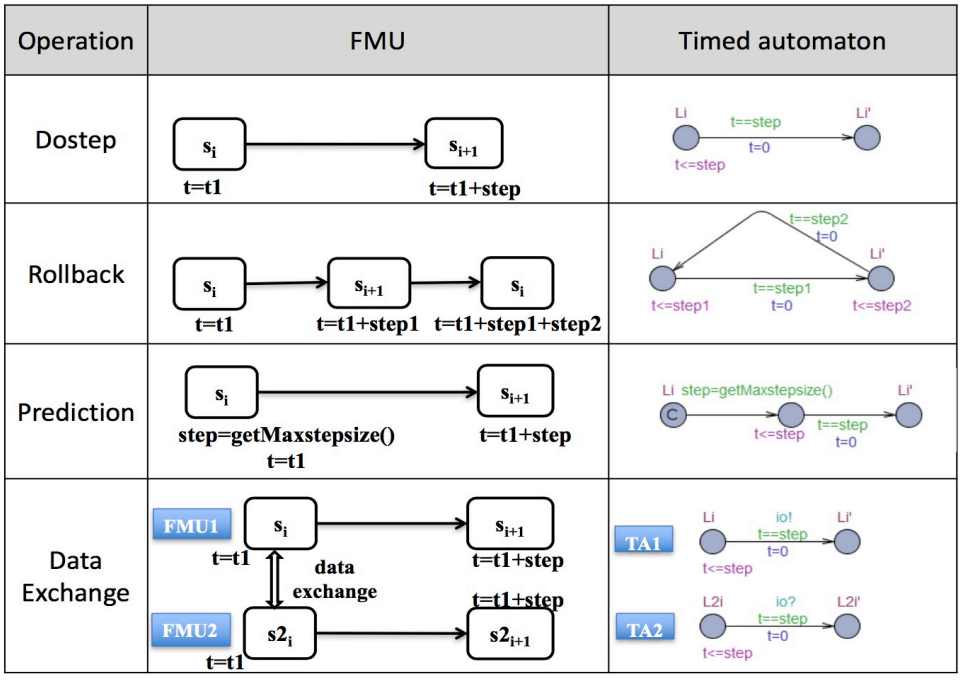


图 3.4: Encoding rules from FMU to TA.

将 FMU 直接转化为时间自动机是比较难得，S. Tripakis 层在论文 [?] 中将时间自动机编码为 FMU。我们受到此论文的启发，根据时间自动机和 FMU 之间语义

的关联，提出了几条从 FMU 到时间自动机在操作语义上的映射规则如图3.4所示。

给定 FMUt_1 时刻的当前状态 s_i，操作函数 $Dostep$ 可以使得 FMU 在 $t_1 + step$ 时刻到达 s_{i+1} 状态。这个操作可以用时间自动机的迁移来进行表示：在位置 L_i 延迟 $step$ 的时间并迁移到一个新的位置 L_i'。

对于操作函数 $Rollback$，给定 FMUt_1 时刻的当前状态 s_i，FMU 首先执行步长 $step1$ 并在 $t_1 + step1$ 时刻到达 s_{i+1} 状态，然后操作函数 $rollback$ 又使得 FMU 回滚到上一个状态 s_i。对于时间自动机来说，这个可以表示为在 L_i 位置延迟了 $step1$ 时间单位并迁移到新的位置 aL_i'，之后又迁移到了上一个位置 L_i。

对于操作函数 $Prediction$,给定一个状态 s_i,FMU 可以由一个函数 $getMaxstepsieze$ 得到下一步能执行的最大步长，然后执行此步长并在 $t_1 + step$ 时刻到达 s_{i+1} 状态。对于时间自动机，可以表示为在 L_i 位置执行了一个函数并得到要延迟的时间 $step$, 然后延迟该时间单位并迁移到一个新的位置 L_i' .

对于操作函数 $DataExchange$,两个 FMU 在 t_1 时刻的 s_i 状态执行 $DataExchange$ 进行了数据交换, 然后他们执行相同的步长并迁移到下一个位置 s_{i+1} 和 $s2_{i+1}$。对于时间自动机，可以表示为两个时间自动机在 t_1 时刻通过信道 io 进行了同步，然后延迟了相同的时间并 L_i 位置迁移到 L_{i+1} 位置。

为了将 FMU 用时间自动机进行形式化描述，我们提出了以上操作语义的映射规则。为了证明我们以上规则的正确性，我们分析了 FMU 和时间自动机的执行片段如下所示：

- 对于操作函数 $Dostep$ 的映射，在 FMU 和时间自动机中的执行片段为 (s_i, t_1), $(s_{i+1}, t_1 + step)$ 和 (l_i, t), $(l_i', t + step)$. 这说明时间自动机和 FMU 都执行 $step$ 的时间单位并到达了一个新的状态或是位置。

- 对于操作函数 $Rollback$ 的映射, 在 FMU 和时间自动机的执行片段为 (s_i, t_1), $(s_{i+1}, t_1+step1)$, $(s_i, t_1+step1+step2)$ 和 (l_i, t), $(l_i', t+step1)$, $(l_i, t+step1+step2)$。这说明时间自动机和 FMU 都首先执行了 $step1$ 时间单位，并且到达了一个新的状态或位置，然后执行了 $step2$ 时间单位, 又返回到了之前的状态或位置。

- 对于操作函数 $Prediction$ 的映射，FMU 和时间自动机的执行片段为 (s_i, t_1), $(s_{i+1}, t_1 + step)$ 和 (l_i, t), $(l_i', t + step)$. 这说明时间自动机和 FMU 都成功预测到了步长 $step$ 并执行了此步长。

- 对于操作函数 $DataExchange$ 的映射，FMU1 和时间自动机 TA1 的执行片段为 (s_i, t_1), $(s_{i+1}, t_1 + step)$ 和 (l_i, t), $(l_i', t + step)$。FMU2 和时间自动机 TA2 的执行片段为 $(s2_i, t_1)$, $(s2_{i+1}, t_1 + step)$ 和 $(l2_i, t)$, $(l2_i', t + step)$。这说明时间自动机和 FMU 在经过了数据交换后又执行了 $step$ 时间单位。

为了更加准确的验证映射规则的正确性，我们分析了时间自动机和 FMU 的整个执行序列。我们通过分析得到 FMU 和时间自动机的执行序列是等价的，验证了映射规则的正确性。在之后几个小节的内容中，我们将此映射规则应用到了水箱的案例中，根据我们之后对案例仿真片段的分析，我们也发现映射规则是正确的。

3.3.2　基于 SysML 的架构建模

为了更好的阐述我们的方法，我们采用了水箱的案例 [**?**] 作为驱动以更加形象的展示我们的方法。下面我们先简单的介绍一下水箱案例，然后用 SysML 建模语言对整个案例的架构建模。水箱案例：有一个水源可以向水箱里面注水，且水箱里面的水通过一个管道流入到一个水池当中，这个水源的流出由一个阀门控制，而阀门的开关由一个控制器进行控制。在本小节，由于水箱、阀门、控制器连接方式的不同，我们在此描述了三种不同类型的水箱系统。

SysML 是为模型驱动式软件开发（Model-Based Systems Engineering，MBSE）[**?**] 而提出的一种通用的领域建模语言（Domain-Specific Language，DSL）[**?**]，它起源于国际系统工程委员会（INCOSE）的倡议 cite Pepper2015International 并于 2001 年 1 月发布。SysML 的 BDD 图用块来描述系统中组件的结构；IBD 图用来描述各个块之间的关联关系。各个块的接口由连接器进行关联，因此，系统各个组件的依赖关系可以用各个块的接口之间的连接进行表示。

图3.5是水箱案例的 SysML BDD 图，其中包含三个块：$Valve$, $Tank$ 和 $Controller$。

$Valve$ 和 $Tank$ 是物理组件；$Controller$ 是信息组件。每一个组件都有自己的输入和输出接口，例如：$Valve$ 的输入接口是 vin，它用来输入阀门的开关 $OpenClosed$ 信号。

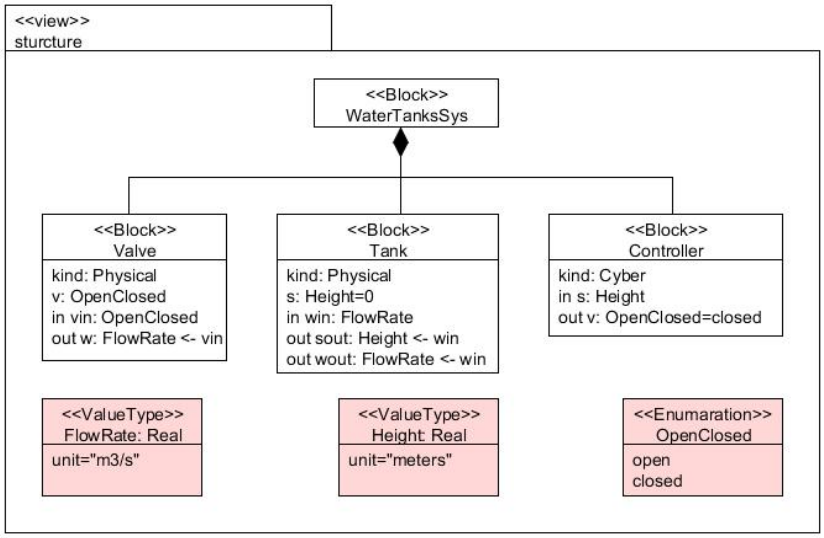
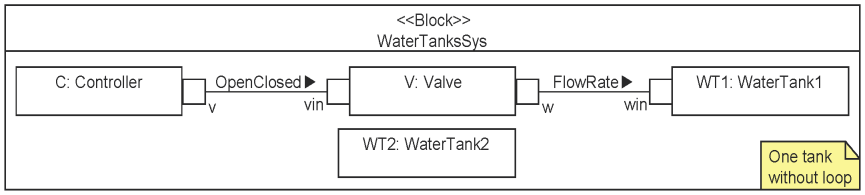


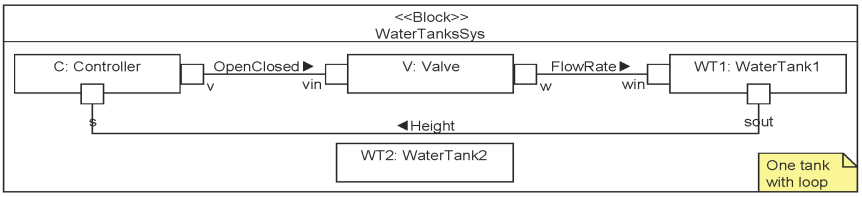
图 3.5: 水箱系统的 SysML BDD

图3.6 是 SysML 的 IBD 图，在这里我们给出了三种连接情形。在第一种情形中，系统包含一个阀门、一个控制器和一个水箱，控制器随机的给阀门发送开关信息，导致水箱中的水位不断的变化；在第二种情形中，控制器信号的发送受到水箱水位的影响，控制器根据水箱的水位发送开关信号；在第三种情形中，我们添加了一个水箱 $waterTank2$，水箱 $waterTank1$ 中的水通过管道先流入 $waterTank2$ 中，最后流入水池。

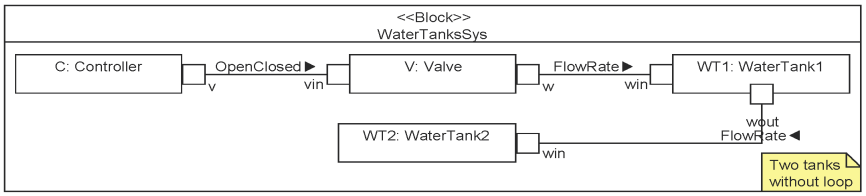
在本小节我们用 SysML BDD 图描述了系统组件的结构，并用 SysML 的 IBD 图描述了各个组件之间的连接。在下一个小节，我们基于 FMI 标准用 FMU 来实现每一个系统组件，并将 SysML IBD 中描述的关联关系用一个 FMU 之间的接口配置文件进行表示。



(a) 情形 1



(b) 情形 2

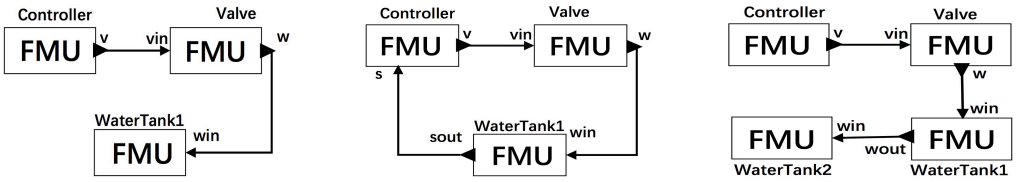


(c) 情形 3

图 3.6: 水箱系统的 SysML IBD

3.3.3　基于 FMI 的连接关系配置

图3.7描述的是水箱系统中的 FMU 组件及各个 FMU 之间的连接。根据图3.6给定的 SysML IBD，我们也得到了三种 FMU 的情形。第一种情形如图3.7(a)所示，系统中有三个 FMU 组件：$Controller$, $Valve$ 和 $WaterTank1$ 和两个接口 v_vin 及 w_win。$Controller$ 和 $Valve$ 由 v_vin 接口连接；$Valve$ 和 $WaterTank1$ 由 w_win 接口连接。第二种情形如图 3.7(b)所示，其中在第一种情形上添加了 $WaterTank1$ 和 $Controller$ 的接口 $sout\_s$，表示控制器信号的发送受到水箱 $WaterTank1$ 的水位影响。图3.7(c)是第三种情形，在第一种情形上添加了另一个水箱 $WaterTank2$, 水箱 $WaterTank1$ 和 $WaterTank2$ 由接口 w_out 连接。



(a) 情形 1　(b) 情形 2　(c) 情形 3

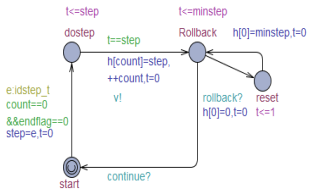
图 3.7: 水箱系统中 FMU 的三种连接情形

在本小节我们设计了 FMU 及 FMU 之间的连接，我们只需要添加一个协同仿真的主算法就可以在协同仿真引擎当中对整个异构系统进行协同仿真并得到仿真迹。但是，在进行仿真之前，我们要保证各个 FMU 之间协同行为的正确性，因此，我们在下一个小节基于时间自动机理论对系统的协同行为进行验证，这也是我们本文的主要贡献点之一。

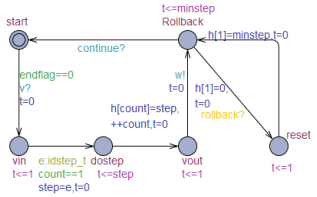
3.3.4　协同行为的验证分析

在本小节我们基于时间自动机理论对水箱系统中组件之间的协同行为的正确性进行验证。首先我们根据小节3.3.1中提出的映射规则，将水箱系统中的 FMU 用时间自动机进行形式化描述，并取小节3.2.2中建模的一个主算法（在此，采用可回滚算法作为主算法，其他算法的验证分析与该算法类似，在此不做描述），FMU 之

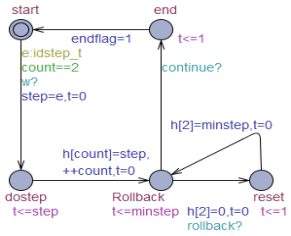
间的接口配置文件我们用时间自动机之间的信道进行描述。由此，我们得到了一个由主算法、FMU 的时间自动机模型及由配置文件转化得到的信道组成的时间自动机网络。图3.8为上述小节中情形 1 的时间自动机网络模型。



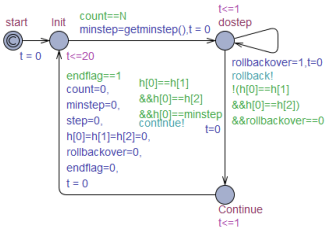
(a) FMU_controller 的时间自动机模型



(b) FMU_valve 的时间自动机模型



(c) FMU_WaterTank1 的时间自动机模型

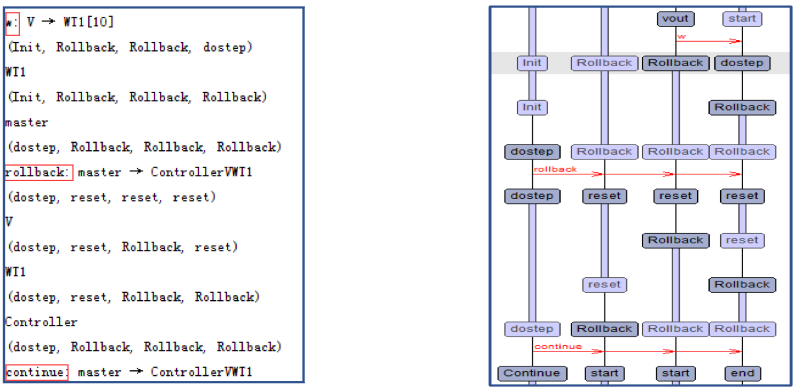


(d) 可回滚主算法的时间自动机模型

图 3.8: 情形 1 的时间自动机网络: $controller \parallel valve \parallel WaterTank1 \parallel MA$.

图3.8(a), 3.8(b), 3.8(c)分别是 $controller$, $valve$ 和 $WaterTank1$ 的时间自动机模型。这些自动机都有四个主要的位置：$start$, $dostep$, $Rollback$ 和 $reset$。图3.8(a)是 $controller$ 的时间自动机模型，它首先通过信道 v 与 $valve$ 的时间自动机模型进行交互并到达 $Rollback$ 状态，然后等待主算法的信号，直到它收到了来自主算法的 $continue$ 信号再与其他的 FMU 进行数据交互并到达 $start$ 状态；否则，它收到 $rollback$ 信号，并回到 $Rollback$ 状态。$valve$ 和 $WaterTank1$ 的时间自动机中位置和迁移与 $controller$ 类似。图3.8(d) 是主算法的时间自动机模型，首先主算法先进性参数初始化，然后根据条件来判断发出 $continue$ 信号或是 $rollback$ 信号。

图3.9是协同过程在 UPPAAL 中的执行片段，我们可以看到 $valve$ 首先发送了信号 w 来与 $WaterTank1$ 进行数据交换，然后，$WaterTank1$ 到达了 $dostep$ 状态，之后主算法广播 $rollback$ 信号导致 FMU 到达 $reset$ 状态，最后主算法发送 $continue$

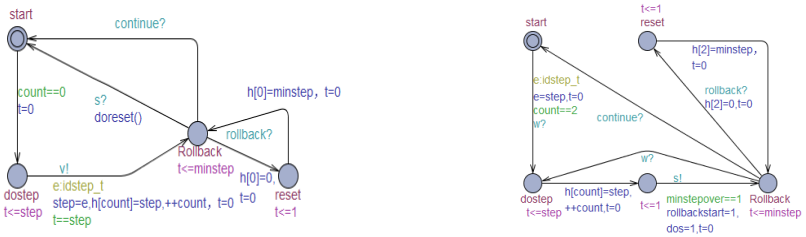


(a) 部分迹　　(b) 执行序列图

图 3.9: 协同过程在 UPPAAL 中的执行序列。

信号使得所有的 FMU 到达 *start* 状态并开始下一步仿真。通过对执行序列的分析，我们发现模型可以正确仿真。

为了比较小节3.3.3中描述的三种情形的协同行为，我们同时用时间自动机形式化描述了其他两种情形。对于第二种情形，我们在第一种情形基础上添加了 $controller$ 和 $WaterTank1$ 之间的信道 s 如图3.10所示；对于第三种情形我们建模了 $WaterTank2$ 的模型并添加了信道 $w2$ 如图 Fig.3.11所示。其他的模型与第一种情形中的模型类似，我们在此只给出了新增加的或有改动的模型。接下来我们将对这三种情形进行验证。



(a) TA for FMU_controller　　(b) TA for FMU_WaterTank1

图 3.10: TA for connection case 2.

我们对每一种情形都验证了以下属性：

- $E\langle\rangle\, WT1.Rollback$ 和 $E\langle\rangle\, master.Continue$ 为可达性验证，它表示 $WaterTank1$

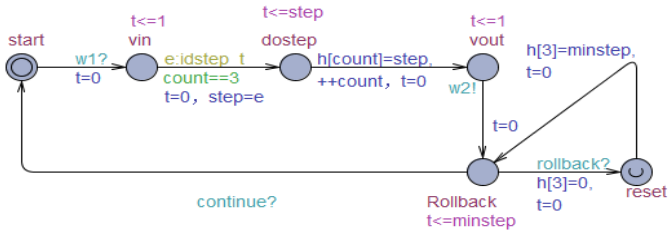


图 3.11: TA for FMU_WaterTank2 of connection case 3.

将到达 *Rollback* 状态且主算法会到达 *Continue* 状态。

- $master.start -> master.Continue$ 为活性验证，它表示一旦主算法开始，它最早会到达 *Continue* 状态.
- $A[]\ not\ deadlock$ 为死锁的验证，它用来验证系统有无死锁。

验证结果如表3.2所示，我们发现情形 1 和情形 3 的验证属性都满足，它表示该情形的协同是正确的。然而情形 2 的可达性和活性不满足，是由于在该模型中出现了环路依赖，我们需要消除该环路依赖再进行下一步的仿真，在本文中我们只关注协同行为的验证，对于如何消除环路依赖在接下来的工作中我们会做进一步研究。

3.4　本章小结

在本小节我们提出了一种新的方法来验证异构系统协同行为的正确性，我们首先用 SysML 建模语言建模了整个系统的架构，然后基于 FMI 标准对整个架构进行实现。之后提出了一种映射规则将 FMU 用时间自动机进行了形式化描述，最后基于时间自动机理论对整个系统的协同行为的正确性进行了验证。经过本小节的验证，我们可以得到通过验证的基于 FMI 标准的模型，该模型可以在协同仿真引擎中直接仿真并得到仿真迹，在下一个小节，我们将提出一种高效的统计模型检测方法，针对某些验证属性对协同仿真的迹进行定量的评估。

表 3.2: 三种情形的验证结果

情形	验证属性	结果
情形 1	$E\langle\rangle\ WT1.Rollback$	True
	$E\langle\rangle\ master.Continue$	True
	$master.start- > master.Continue$	True
	$A[]\ not\ deadlock$	True
情形 2	$E\langle\rangle\ WT1.Rollback$	True
	$E\langle\rangle\ master.Continue$	False
	$master.start- > master.Continue$	False
	$A[]\ not\ deadlock$	True
情形 3	$E\langle\rangle\ WT1.Rollback$	True
	$E\langle\rangle\ master.Continue$	True
	$master.start- > master.Continue$	True
	$A[]\ not\ deadlock$	True

第四章　基于分布式统计模型检测的异构系统验证分析

通过对上一个章节得到的模型进行协同仿真，我们可以得到仿真迹。将仿真迹和需要验证的属性输入到统计模型检测器中，即可以对整个异构系统的行为进行定量的评估分析。

4.1　基于抽象和学习的分布式统计模型检测方法

4.2　异构系统验证分析

4.3　本章小结

第五章　工具实现

5.1　MODANA 平台介绍

5.2　基于 MODANA 平台的 Co-SMC 工具介绍

5.3　基于统计模型检测的异构系统验证程序实现

5.4　本章小结

第六章　案例分析与实验评估

6.1　案例一建模与分析：XXX 系统

6.1.1　XXX 系统建模及验证属性

6.1.2　XXX 系统验证分析

6.1.3　模型检测算法效率评估

6.2　案例二建模与分析：XXX 系统

6.2.1　XXX 系统建模及验证属性

6.2.2　XXX 系统验证分析

6.2.3　模型检测算法效率评估

6.3　本章小结

第七章　总结与展望

致　　谢

在此论文完成之际，我首先要感谢我的导师杜德慧副教授。她严肃的科学态度，严谨的治学精神，精益求精的工作作风，深深地感染和激励着我。从课题的选择到项目的最终完成，杜老师都始终给予我细心的指导和不懈的支持。三年多来，杜老师不仅在学业上给我以精心指导，同时还在思想、生活上给我以无微不至的关怀，在此谨向杜老师致以诚挚的谢意和崇高的敬意。

感谢在研究生学习期间给我诸多教诲和帮助的软件学院的各位老师和同学、以及和我一起生活两年半的室友，你们的执着、勤奋、以及对生活的态度，值得我学习。“君子和而不同”，我们正是如此！愿我们以后的人生都可以充实、快乐！

最后感谢我的家人，谢谢你们在我成长道路上支持、鼓励我，让我独立地选择自己的人生道路；同时谢谢我的女朋友，在学习、生活中对我的帮助和鼓励！

姜 凯强

二零一八年五月

攻读硕士学位期间发表论文、参与科研和获得荣誉情况

■ 已完成学术论文

[1] Cheng B, Wang X, Liu J, et al. Modana: An Integrated Framework for Modeling and Analysis of Energy-Aware CPSs[C]//Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual. IEEE, 2015, 2: 127-136. （一作）

[2] 杜德慧, 程贝, 刘静. 面向安全攸关系统中小概率事件的统计模型检测 [J]. 软件学报, 2015(2):305-320. （二作，导师一作）

[3] Cheng B, Du D. Towards a Stochastic Occurrence-Based Modeling Approach for Stochastic CPSs[C]//2014 Theoretical Aspects of Software Engineering Conference (TASE). IEEE, 2014: 162-169.（一作）

■ 参与的科研课题

[1] 信息物理融合系统的随机行为建模与验证方法研究（国家自然科学基金面上项目, 61472140）

[2] 基于统计模型检测的信息物理融合系统的验证方法研究（上海市自然科学基金项目, 14ZR1412500）

■ 获得荣誉情况

[1] 2015 年获得国家奖学金

[2] 2015 年获得华东师范大学优秀学生称号