

面向复杂随机系统的启发式统计模型检测方法^{*}

何佳^a, 张敏^b, 郭延楠^a, 吕悦^a

(华东师范大学 a. 上海市高可信计算重点实验室; b. 计算机科学与软件工程学院, 上海 200062)

摘要: 统计模型检测是一种高效的验证技术, 常用于复杂的随机系统验证, 如分布式算法等。而在超长路径上对性质进行验证时, 其验证效率会急剧降低。为解决这个问题, 提出一种启发式的统计模型检测算法。在对路径进行验证时, 会查找帮助剪枝的最短前缀; 并在后续抽样时, 利用前缀信息直接判定路径是否满足给定性质, 避免进入费时的路径验证阶段。在与 PRISM 的比较中, 它的路径验证次数相对更少, 且平均抽样路径长度更短。因此统计模型检测技术可应用于超长路径上的性质验证。

关键词: 统计模型检测; 复杂随机系统; 超长路径; 最短前缀; 启发式算法; PRISM

中图分类号: TP31 **文献标志码:** A **文章编号:** 1001-3695(2016)10-3036-05

doi:10.3969/j.issn.1001-3695.2016.10.037

Heuristic statistical model checking approach for complex stochastic systems

He Jia^a, Zhang Min^b, Guo Yannan^a, Lyu Yue^a

(a. Shanghai Key Laboratory of Trustworthy Computing, b. School of Computer Science & Software Engineering, East China Normal University, Shanghai 200062, China)

Abstract: Statistical model checking is an efficient verifying technique. It is suitable for verifying complex stochastic systems, such as distributed algorithm. But its performance drops down when verification is over extremely long path. To address this problem, this paper presented a heuristic statistical model checking approach. In verification stage, it searched for the shortest prefix of path to help pruning. In latter sampling stages, it used these prefixes to determine whether current path satisfied the property. This helped to avoid path verification that was time-consuming. In comparison with PRISM, the results show that it verifies less and sampled shorter paths in average. As a result, it can use statistical model checking to verify properties over extremely long paths.

Key words: statistical model checking; complex stochastic system; extremely long path; shortest prefix; heuristic algorithm; PRISM

为形式化地对随机系统进行验证, 模型检测技术发展出了概率模型检测方法 (probabilistic model checking)^[1,2]。然而随着系统规模的快速增长, 对传统的概率模型检测方法^[3]提出了挑战。使用穷举式的算法来验证诸如调度问题^[1]、分布式算法^[4,5]和生物系统^[6,7], 变得非常困难。统计模型检测被提出来解决这一问题^[6,8-11]。不同于穷尽式的分析, 它们会先对模型进行大量的随机路径抽样, 并基于这些样本, 给出近似的解。目前有两种统计模型检测方法, 即参数估计式^[11]和假设检验式^[6,8-10]。两者主要在求解问题类型上有一些区别: 在收集到足够的样本后, 前者会基于这些样本, 给出模型满足给定性质的近似概率, 属于定量的结果; 而后者会基于样本给出模型是否满足给定性质, 属于定性的结果。

除此之外, 两者也存在共同点: 对于每次抽样, 它们都仅随机选取一条从初始状态开始的有限路径, 并对之验证。这使得验证的复杂度与模型的复杂度无关, 从而在理论上可用于验证非常复杂的随机系统。可以发现, 抽样与验证的数量是影响这两种验证算法效率的关键。因为如果抽样 n 次, 则同样需要对路径验证 n 次。对于路径验证的时间复杂度往往要比路径抽样的时间复杂度高得多 (CTL 即 computation tree logic 的验证

时间复杂度至少是 PTIME, LTL 即 Linear temporal logic 的验证时间复杂度至少是 PSPACE-complete^[12], 而路径抽样至多只需要与路径长度呈线性的时间复杂度)。因此, 假设验证一条长度为 k 的路径的时间复杂度至少为 $O(k \times |\psi|)$, 其中 $|\psi|$ 是所需验证性质的复杂度。忽略可信度计算时间, 则抽样 n 次并验证 n 次所需的时间复杂度就是 $O(n \times k \times |\psi|)$ 。当所需验证的路径长度 k 增大 1 时, 所需验证时间就会增大 $n \times |\psi|$ 。由于需要更多的抽样来获得更精确的近似解, 在超长路径上验证性质时, 验证时间就会急剧增加。为解决上述问题, 这里提出一种启发式的模型检测算法。通过在验证时利用算法定位最早能够决定该路径是否满足性质的关键状态。并在后续的抽样中, 利用所收集的关键状态信息, 来帮助直接判断路径是否满足性质, 从而避免对超长路径进行验证。

1 相关工作

国外的学者提出了几种统计模型检测方法, Herault 等人^[11]引入了一种基于参数估计的验证方法 APMC (approximate probabilistic model checking)。它主要应用于离散随机模型的

收稿日期: 2015-11-09; 修回日期: 2015-12-30 基金项目: 国家自然科学基金青年科学基金资助项目 (61202105); 国家自然科学基金资助项目 (61361136002)

作者简介: 何佳 (1986-), 男, 硕士研究生, 主要研究方向为形式化方法 (jhe@ecnu.cn); 张敏 (1977-), 女, 副教授, 主要研究方向为形式化方法; 郭延楠 (1993-), 男, 硕士研究生, 主要研究方向为形式化方法; 吕悦 (1994-), 女, 硕士研究生, 主要研究方向为形式化方法。

验证(如离散时间的马尔可夫链),并获得一个定量的结果,如“系统在 n 个时间单位内终止的概率”。该方法会使用 Chernoff 界来计算 N ,即需要抽样的数量。而在不断抽样与验证的过程中,该方法会记录下 A ,即符合给定性质的路径条数。在完成 N 次抽样和验证后,它会使用 A/N 来估计模型满足给定性质的概率。解的正确性由 Chernoff 界保证。

Younes 等人提出了一种基于 Wald 序贯概率比率测试(Wald's sequential probability ratio test)^[13]的方法。它实际是一种假设检验式的方法。不同于文献[11],它只能给出模型是否满足给定性质,而不能给出模型多大程度满足给定性质。它可用于验证诸如“系统在 n 个时间单位内终止的概率是否大于 $C\%$ ”的性质。Jha 等人^[6]提出了一种基于贝叶斯因子的方法。它也是基于假设检验。因此,其主要过程与文献[9, 10]类似。Zuliani 等人^[8]发表了另外一种基于贝叶斯统计学的方法。它主要基于贝叶斯区间估计,因此是一种参数估计式的方法。

由于上述方法都没有在抽样时考虑记录信息来帮助后续的验证,所以它们的抽样次数与验证次数总是相等的。在抽样到相同前缀的路径甚至相同的路径时,依然会进行重复的验证。

而国内的学者主要研究统计模型检测的应用领域。文献[14]中的验证方法主要混合了上面^[6,8,9]几种方法,根据不同的应用场景采取不同的统计验证方法,提高了预测小概率事件的成功率。而文献[15]中主要将统计模型检测技术应用于海洋锋面的结构研究,通过结合预处理和遗传算法,提高了结果的有效性。

2 研究背景

2.1 离散时间马尔可夫链与概率迁移系统

验证的对象主要是针对随机离散事件系统(离散时间)。这类系统抽象模型最先由 Younes 等人^[9]提出。可利用已广泛使用的离散时间马尔可夫链来对这类系统进行建模。

为形式化地对这类系统进行验证,先给出离散时间马尔可夫链模型 DTMC(discrete-time Markov chain)的定义:

定义 1 离散时间马尔可夫链是一个二元组 (S, T) 。其中: S 是有限状态集; $T: S \times S \rightarrow [0, 1]$ 是概率迁移函数,并对所有的状态 s 与其所有后继 s' 满足 $\sum T(s, s') = 1$ 。

其次,需要为马尔可夫链添加标签的功能。假设 AP 是一个原子命题集合,则有:

定义 2 概率迁移系统 PTS(probabilistic transition system)是一个四元组 (S, T, s_0, L) 。其中: (S, T) 是一个离散时间马尔可夫链; $s_0 \in S$ 是唯一的初始状态;而 $L: S \rightarrow 2^{AP}$ 是一个从状态到原子命题集的映射函数。

2.2 事件的测度

假设 $p = s_0 s_1 s_2 \cdots s_k$ 是一条有限路径上的状态序列。其中对于所有 $i \geq 0$, 有 $P(s_i, s_{i+1}) > 0$ 。并令 $p[i]$ 表示 p 的第 $(i+1)$ 个状态。如果随机获得一条模型中的有限路径,则抽样到该条路径的概率是可测度的,并且在文献[12]中已给出了完整的定义。它在测度函数中使用了圆柱体集合(cylinder set)作为它的基本事件。这里使用 $Pr(event)$ 表示事件 $event$ 发生的概率。

2.3 概率计算树逻辑

这里主要通过概率计算树逻辑 PCTL(probabilistic computation tree logic)来指定性质。它是计算树逻辑 CTL 的扩展,并加入了概率操作符。其一般形式为 $P_{\theta \triangleright t}(\psi)$ 。其中: $\triangleright \in \{<, >, =, \leq, \geq\}$; $t \in [0, 1]$; 这里称 θ 为概率阈值。其语法已经在文献[12]中完整定义,故不再赘述。这里主要定义其满足关系。假设 $a \in AP$ 是一个原子命题, $PTS = (S, T, s_0, L)$ 是一个概率迁移系统,且状态 $s \in S$ 。 φ 和 ψ 是 PCTL 状态公式,且 ψ_i 是 PCTL 路径公式。并令 $Path(s)$ 表示所有以状态 s 为初始状态的路径。这样对于状态 s 来说,状态相关的满足关系可以定义如下:

定义 3 对于状态 s ,其满足关系定义如下:

- $s \models a$ 当且仅当 $a \in L(s)$;
- $s \models \neg \phi$ 当且仅当 s 不满足 ϕ ;
- $s \models \phi \wedge \psi$ 当且仅当 $s \models \phi \wedge \psi$ 且 $s \models \phi \wedge \psi$;
- $s \models P_{\theta \triangleright t}(\psi)$ 当且仅当 $Pr(s \models \psi) \sim \theta$ 。

定义 4 对于路径 p , $|p|$ 为路径的长度,则其满足关系定义如下:

- $p \models O\psi$ 当且仅当 $p[1] \models \psi$ 且 $|p| \geq 1$;
- $p \models \psi_1 U^{\leq n} \psi_2$ 当且仅当存在 i 且 $0 \leq i \leq n \leq |p|$, 满足 $p[i] \models \psi_1$ 并且对于所有 $0 \leq j < i$, 有 $p[j] \models \psi_2$;

其中, $Pr(s \models \psi)$ 表示 $Pr(\{p \mid p \models \psi \wedge p \in Path(s)\})$ 。

最后,可以定义概率迁移系统的满足关系如下:

定义 5 对于概率迁移系统 PTS,其满足关系定义如下:

$PTS \models \psi$ 当且仅当 $Path(s_0) \models \psi$ 。

它表示如果从初始状态 s_0 开始的所有路径符合,则概率迁移模型 PTS 符合 ψ 。

3 启发式统计模型检测抽样与验证算法

前文已提到,统计模型检测一般分为参数估计式和假设检验式两类。而它们在抽样随机有限路径的过程中,都包含固定的两个步骤,即随机有限路径的选取和有限路径的验证。然而由于路径验证的时间复杂度较高,在需要大量样本来求解的情况下,会需要大量时间来进行路径的验证。并且当需要验证的路径长度为 k ,且给定性质的复杂度为 $|\psi|$ 时,则每条路径验证的时间复杂度都至少是 $O(k \times |\psi|)$;另一方面,随机取样至多需要 $O(k)$ 的时间复杂度。因此,根据已有样本信息来避免进入路径验证阶段和减少抽样数量,能大大提高验证的效率。

3.1 一个基本观察

观察图 1,这是经典的用公平硬币来模仿六面骰子的过程,它最初由 Knuth 等人提出^[16]。它的初始状态为 s_0 ,表示还未开始掷硬币。每条出边表示当硬币为正面或反面时,所迁移往的状态。并且硬币为正面与为反面的概率相等,都是 50%。而最底下的圆形状态表示根据硬币投掷情况所映射到的骰子数字。并且都指向自己,属于吸收状态^[12]。值得注意的是,在 $s'_{1,2,3}$ 和 $s'_{4,5,6}$ 分别有指向 $s_{1,2,3}$ 和 $s_{4,5,6}$ 的回边。它表示根据目前所掷硬币的情况,映射到了一个‘空’的位置,因此需要重新抛掷硬币来决定。

假设现在要验证性质“在三次投掷次数内,最终所映射的数字是 2 或 3 的概率大于 40%”。该性质使用 PCTL 可定义为

$$P_{\theta > 0.4}(\text{true} U^{\leq 3} (s_2 \vee s_3)) \quad (1)$$

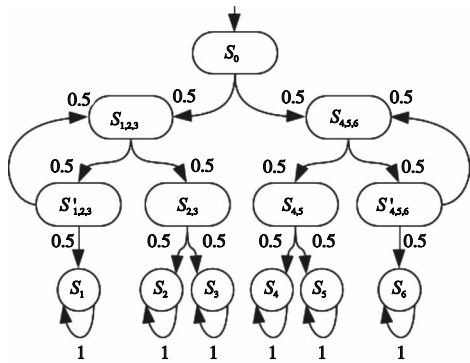


图1 使用硬币模仿骰子的马尔可夫链

模型是明显不满足性质式(1)的,最终所映射的数字是2或3的概率肯定小于40%。但是依然按照一般统计模型检测的方法来尝试验证。假设抽取的有限路径长度为3,第一次抽样到的路径为 $p_1 = \langle s_0, s_{1,2,3}, s_{2,3}, s_2 \rangle$,则经过验证,路径 p_1 是符合性质的。所以目前抽样总数 $n=1$ 且满足的路径数 $x=1$ 。假设抽样第二条路径 p_2 时,第二条路径 p_2 的前三个状态依然是 $\langle s_0, s_{1,2,3}, s_{2,3} \rangle$ 。通过观察可以发现,如果继续随机选择状态,则必定会迁移到状态 s_2 或 s_3 。因此,并不需要继续抽样路径的过程,就已经可以断定 p_2 满足性质(1)。通过这个观察可以发现,通过在抽样时保存一些关于验证结果的信息,对加快后续随机有限路径的验证很有帮助。因此,考虑在验证的同时,保存一些有利于后续路径验证的信息,使得尽早地停止抽样并避免路径验证的过程,从而加快整个验证过程。

3.2 最短可判前缀

通过上面的观察,本文考虑通过在验证的过程中保存路径的前缀,使得在下次再次随机抽样到该路径前缀时能够马上作出判断。然而主要问题是:如何找出最短的路径前缀?举个例子,在对路径 $p_3 = \langle s_0, s_{1,2,3}, s_{2,3}, s_2 \rangle$ 进行验证时,前文已提到,在状态 $s_{2,3}$ 时,已经可以判定该条路径满足性质式(1);而其实在状态 s_2 或 s_3 时,也可以判定该路径满足性质式(1)。因此,一条路径上可能存在多个可以进行判定的前缀。但在状态 $s_{2,3}$ 时,可以相比其他两个状态,更早地确定该路径满足性质式(1),且省去了一部分的抽样过程。

进一步地,需要验证的性质往往包括子性质。如果要找出满足上层性质的最短前缀,则需要预先知道可判定子性质的最短前缀。因此,整个过程是递归的,这与经典的CTL验证算法类似^[12]。所以本文考虑在对路径验证的同时来收集关于最短前缀的信息。另外假定关键状态表示最短前缀中,最后一个状态。并且,由于关键状态的位置,决定了最短前缀的长度,所以,将问题转换为寻找关键状态。

为形式化地解决这个问题,先给出路径 p 中状态 s 对于PCTL性质 Φ ,其最短前缀和关键状态的定义。为方便描述,先定义一种偏序关系:假定对于路径 $p = \dots s_i \dots s_j \dots s_k$,称 s_j 是 s_i 和 s_i 两者中的较大值,而称 s_i 是两者中的较小值。

定义6 最短可判前缀。对于路径 p 中状态 s 验证PCTL性质 Φ 时, p 的最短可判前缀是始于 s 的、最短的、能够确定剩余路径是否满足 Φ 的路径前缀。

定义7 关键状态。关键状态是最短可判前缀中最后一个状态。

下面给出并证明一系列定理,通过它们可以根据状态 s 和不同性质公式来确定关键状态的位置,这样就可以在算法中基

于这些定理来找出对应的关键状态。

定理1 对于路径 p 中状态 s 、原子公式集AP、PCTL路径公式 ψ_i ,有:

a) 对状态 s 验证性质 $\Phi = a$ ($a \in AP$),则状态 s 对 Φ 的关键状态是 s 自己。

b) 对状态 s 验证性质 $\Phi = !\psi$,则状态 s 对 Φ 的关键状态是 s 对 ψ 的关键状态。

c) 对状态 s 验证性质 $\Phi = \psi_1 \wedge \psi_2$,如果:

(a) s 满足 ψ_1 且同时满足 ψ_2 ,则 s 对于 $\psi_1 \wedge \psi_2$ 的关键状态是 s 对于 ψ_1 的关键状态与 s 对于 ψ_2 的关键状态的较大值。

(b) s 不满足 ψ_1 且同时不满足 ψ_2 ,则 s 对于 $\psi_1 \wedge \psi_2$ 的关键状态是 s 对于 ψ_1 的关键状态与 s 对于 ψ_2 的关键状态的较小值。

(c) s 仅满足 ψ_1 或 ψ_2 ,则 s 对于 $\psi_1 \wedge \psi_2$ 的关键状态是 s 所不满足那个性质的关键状态。

d) 对状态 s 验证性质 $\Phi = \psi_1 \vee \psi_2$,如果:

(a) s 满足 ψ_1 且同时满足 ψ_2 ,则 s 对于 $\psi_1 \vee \psi_2$ 的关键状态是 s 对于 ψ_1 的关键状态与 s 对于 ψ_2 的关键状态的较小值。

(b) s 不满足 ψ_1 且同时不满足 ψ_2 ,则 s 对于 $\psi_1 \vee \psi_2$ 的关键状态是 s 对于 ψ_1 的关键状态与 s 对于 ψ_2 的关键状态的较大值。

(c) s 仅满足 ψ_1 或 ψ_2 ,则 s 对于 $\psi_1 \vee \psi_2$ 的关键状态是 s 所满足那个性质的关键状态。

e) 对状态 s 验证性质 $\Phi = O\psi$ (next 操作符),则状态 s 对于 $O\psi$ 的关键状态是路径 p 中 s 下一个状态对于 ψ 的关键状态。

f) 对状态 s 验证性质 $\Phi = \psi_1 U^{\leq n} \psi_2$ (until 操作符),如果:

(a) s 不满足 $\psi_1 U^{\leq n} \psi_2$,则状态 s 对于 $\psi_1 U^{\leq n} \psi_2$ 的关键状态是从 s 开始的,第一个同时不满足 ψ_1 和 ψ_2 的状态对于 ψ_1 和 ψ_2 的关键状态的较大值。

(b) s 满足 $\psi_1 U^{\leq n} \psi_2$,则状态 s 对于 $\psi_1 U^{\leq n} \psi_2$ 的关键状态是从 s 开始的,第一个满足 ψ_2 的状态对于 ψ_2 的关键状态。

证明 由于篇幅有限,且其他定理的证明过程类似,所以主要对b)进行证明。先对可确定性进行证明:因为通过 s 对 ψ 的最短可判前缀,可以确定 s 是否满足 ψ ,所以也能够确定 s 是否满足 $!\psi$ 。再对它是最短的进行证明:假设 s 对 $!\psi$ 的最短可判前缀 Pre_1 要短于 s 对 ψ 的最短可判前缀 Pre_2 。由于通过 Pre_1 可以确定 s 是否满足 $!\psi$,所以,也能够确定 s 是否满足 ψ 且长度短于 Pre_2 ,这样便产生矛盾。因此, Pre_1 的长度要大于等于 Pre_2 , Pre_2 是最短长度。但是仅仅这些还不够,需要找出整条有限路径的关键状态。假定 $p[i]$ 表示路径中第 i 个状态,则路径 p 对于PCTL性质 Φ 的关键状态可定义如下:

定义8 路径 p 对于PCTL性质 Φ 的关键状态,就是 $p[1]$ 对于性质 Φ 的关键状态。

这样一来,只需找出初始状态对于给定性质的关键状态,就能够确定该条路径的关键状态。

3.3 启发式抽样算法描述

基于之前的讨论,现在的目标是找到一种算法,它可以对有限路径进行验证,并同时收集最短可判前缀。因此,首先给出抽样算法。它会在每个选择状态的阶段中,根据已保存的最短可判前缀信息来尝试判定该路径前缀是否满足给定性质,以避免进入路径验证状态。其次再给出PCTL的验证算法。它会在验证的过程中保存最短可判前缀,并给出验证结果。由于对算法进行启发式的改进是独立于该方法是参数估计式或假设检验式,所以仅给出基于APMC^[11]方法改进的参数估计式算法。随机路径抽样算法见算法1。它的输入参数分别是概

率迁移模型 PTS, 给定路径长度 k 和已知的最短可判前缀映射 prefixMap。假定 prefixMap 中的键是路径前缀, 而所映射的值是该前缀是否满足给定性质。它会根据 k 不断地随机选取下一个状态来组成路径。如果当前的路径前缀是已知的最短路径前缀, 则立即返回该路径前缀和其是否满足的结果。这样一来就能够在抽样时, 不用每次都随机选取 k 个状态且对路径进行验证就能直接得到结果。

算法 1 启发式随机路径抽样算法 getRandomPath()

输入: PTS, k , prefixMap。
输出: 抽样路径和路径是否满足给定性质。
 $p = \text{path}()$ // 创建空路径
 $\text{isSat} = \text{"unknown"}$ // 路径是否满足
for $i = 1$ to k do
 $s = \text{randomState}()$ // 根据当前状态随机获取下个状态
 $p = p \text{ concat}(s)$ // 将状态接上路径
 if p in prefixMap then // p 是最短可判定前缀之一
 $\text{isSat} = \text{prefixMap.find}(p)$ // 获得 p 是否满足性质
 return p , isSat
 else
 continue
return p , isSat

验证算法见算法 2。它主要基于 APMC 方法^[11], 但是增加了用于保存最短可判前缀的功能。它的输入参数分别是模型 PTS、给定路径长度 k 、APMC 的可信参数、APMC 的精度参数、最短可判前缀映射 prefixMap 和需要验证的性质 ψ 。首先, 它会使用 APMC 的方法 sampleSize 来获得需要抽样的路径数量, 并根据该数量来对模型进行随机路径抽样。不过在调用 getRandomPath 方法后, 它会先判断该路径是否已经被判定为满足或不满足。如果已判定, 就跳过后续的路径验证阶段, 直接开始抽样下一条随机路径。如果 getRandomPath 返回的结果是“unknown”的(见算法 1), 则 getShortestPrefix 方法会根据验证结果和定理 1 来获得其最短可判前缀, 并将其添加到最短路径前缀的映射中。这样就能够在下一次调用 getShortestPrefix 时, 匹配更多的最短路径前缀。

算法 2 验证算法

输入: PTS, k , c , w , prefixMap, ψ 。
输出: PTS 多大概率满足 ψ 。
 $\text{size} = \text{sampleSize}(c, w)$
 $x = 0$
 $n = 0$
for $i = 0$ to size do
 p , $\text{isSat} = \text{getRandomPath}(\text{PTS}, k, \text{prefixMap})$
 $n = n + 1$
 if $\text{isSat} = \text{True}$ then // 判定为满足
 $x = x + 1$
 continue
 else if $\text{isSat} = \text{False}$ then // 判定为不满足
 continue
 else // 没有匹配前缀
 $\text{verify}(p, \psi)$
 $\text{prefix} = \text{getShortestPrefix}(p)$
 if $p \neq \psi$ then
 $x = x + 1$
 $\text{prefixMap.insert}(\{\text{prefix}, \text{True}\})$
 else
 $\text{prefixMap.insert}(\{\text{prefix}, \text{False}\})$
 end if
end for
return x/n

4 案例研究

为了衡量该算法验证复杂随机系统的效率, 这里使用它来验证一个进程抢占调度的模型。这主要有两个原因: a) 由于

并发进程运行过程的复杂性, 其模型状态会根据并发进程的数量指数倍地增加, 所以传统的模型检测方法不能很好地处理这类问题, 而它却适合于使用应用统计模型检测来进行验证; b) 由于调度理论上可以无限地进行下去, 而不会马上到达一个吸收状态, 这样会使得最终保存的路径前缀都是已到达吸收状态的路径, 这比较适合衡量在验证超长路径验证时该算法在性能上的改善。

为了比较原本统计模型检测方法与启发式统计模型检测方法之间的区别, 笔者开发了基于启发式方法的统计模型检测工具 SPAC (statistical probabilistic approximate model checker)。它主要实现了对离散时间马尔可夫链模型 (DTMC) 上的概率计算树逻辑 (PCTL) 的验证, 并实现了基于 APMC^[11] 的启发式算法。为了得到原始 APMC 验证算法的数据, 本文主要使用概率模型检测工具 PRISM 来进行相同的验证。因为 PRISM 不仅提供了对 APMC 验证算法的良好实现, 并且可通过简洁的语言来对离散时间马尔可夫链模型进行建模。下面先介绍问题背景。假设有 N 个进程, 一个打印机。其中进程 Proc_i 占用打印机的时间为 i 。并假设每个进程除了在使用打印机的时间段之外, 总是在申请使用打印机。而在 N 个进程都在申请打印机的情况下, 调度器分配打印机给进程 Proc_i 的概率为 $i/(N(N+1)/2)$ 。当 $N=4$ 时, 该模型在 PRISM 中的定义见算法 3。

算法 3 进程调度模型在 PRISM 中的定义

输入: 无。
输出: 无。
dtmc
module proc1
 t: int init 0;
 x1: [0..1] init 0;
 x2: [0..1] init 0;
 x3: [0..1] init 0;
 x4: [0..1] init 0;
 x2dur: [0..2] init 0;
 x3dur: [0..3] init 0;
 x4dur: [0..4] init 0;
 x1got: int init 0;
 x2got: int init 0;
 x3got: int init 0;
 x4got: int init 0;
 [] x1 = 0 & x2 = 0 & x3 = 0 & x4 = 0 -> 1/10: (x1' = 1) & (x2' = 0) & (x3' = 0) & (x4' = 0) & (x1got' = x1got + 1) + 2/10: (x1' = 0) & (x2' = 1) & (x3' = 0) & (x4' = 0) & (x2dur' = 2) & (x2got' = x2got + 1) + 3/10: (x1' = 0) & (x2' = 0) & (x3' = 1) & (x4' = 0) & (x3dur' = 3) & (x3got' = x3got + 1) + 4/10: (x1' = 0) & (x2' = 0) & (x3' = 0) & (x4' = 1) & (x4dur' = 4) & (x4got' = x4got + 1);
 [] x1 = 1 & x2 = 0 & x3 = 0 & x4 = 0 -> 1: (x1' = 0) & (x2' = 0) & (x3' = 0) & (x4' = 0);
 [] x1 = 0 & x2 = 1 & x3 = 0 & x4 = 0 & x2dur > 1 -> 1: (x1' = 0) & (x2' = 1) & (x3' = 0) & (x4' = 0) & (x2dur' = x2dur - 1);
 [] x1 = 0 & x2 = 1 & x3 = 0 & x4 = 0 & x2dur = 1 -> 1: (x1' = 0) & (x2' = 0) & (x3' = 0) & (x4' = 0) & (x2dur' = x2dur - 1);
 [] x1 = 0 & x2 = 0 & x3 = 1 & x4 = 0 & x3dur > 1 -> 1: (x1' = 0) & (x2' = 0) & (x3' = 1) & (x4' = 0) & (x3dur' = x3dur - 1);
 [] x1 = 0 & x2 = 0 & x3 = 1 & x4 = 0 & x3dur = 1 -> 1: (x1' = 0) & (x2' = 0) & (x3' = 0) & (x4' = 0) & (x3dur' = x3dur - 1);
 [] x1 = 0 & x2 = 0 & x3 = 0 & x4 = 1 & x4dur > 1 -> 1: (x1' = 0) & (x2' = 0) & (x3' = 0) & (x4' = 1) & (x4dur' = x4dur - 1);
 [] x1 = 0 & x2 = 0 & x3 = 0 & x4 = 1 & x4dur = 1 -> 1: (x1' = 0) & (x2' = 0) & (x3' = 0) & (x4' = 0) & (x4dur' = x4dur - 1);
Endmodule

对于该模型, 先尝试验证以下性质:

$$P_{\theta > 0.5}(\text{true} U^{\leq \text{len}} x2\text{got} > t) \quad (2)$$

它的含义为: 在 len 个时间单位内, 进程 Proc_2 抢占到打印机的次数超过 t 次的概率小于 50%。该性质在 SPAC 与

PRISM 中所需的验证次数统计如表 1 所示,其中对于 APMC 的可信度参数与宽度参数都设置为 0.01。

表 1 SPAC(启发式 APMC)与 PRISM(APMC)的验证次数

len 路径长度	t 分配次数	SPAC(APMC) 验证次数	PRISM(APMC) 验证次数
10	1	6 982	26 492
50	1	5 436	26 492
100	1	5 429	26 492
200	2	15 560	26 492
500	2	15 509	26 492

通过表 1 可以发现,由于 APMC 的抽样与验证次数是根据其可信参数与精度参数而定,所以它的验证次数总是不变的,而 SPAC 的验证次数要明显少于没有使用启发式算法的 APMC 方法。当 t 不变而 len 逐渐变大时,SPAC(APMC)所需验证次数逐渐变少。因为分配给 Proc_2 打印机的概率总是一定的。相比 10 个时间单位,在 50 个时间单位内, Proc_2 得到打印机的概率更大。这对于启发式算法来说,就有更大的概率在路径结束之前根据最短路径直接返回验证结果。因此,对于更长的路径,启发式验证算法的优势还会更加明显。相反,即使路径长度变长,如果该性质在当前路径长度下能够判定的概率较小,则 SPAC 相对 PRISM 的优势会变小。所以性质在 len = 200/500, $t = 2$ 时,虽然路径长度变长了,相对 $t = 1$ 时,仍需要验证更多的样本,不过依然要比无启发的 APMC 要少。因此,启发式算法更适合在对于超长的路径上验证性质。

它的含义为:在 len 个时间单位内,进程 Proc_2 抢占到打印机的次数超过 2 次的概率小于 50%。其中验证的长度 len 会从 10、20 一直变化到 100。分别在 PRISM 与 SPAC 中对它进行验证,可以得到图 2。为了评估启发式方法在不同路径长度上对相同性质进行验证时其抽样路径的平均长度变化,尝试验证以下性质:

$$P_{\theta > 0.5}(\text{true } U^{\leq \text{len}} x2\text{got} > = 2) \quad (3)$$

观察图 2,图中的直线表示对于性质式(3)验证完成后,各种方法所有抽样路径的平均长度。圆点的直线是表示朴素的抽样验证算法。它每次抽样完整路径并验证。因此,它的平均抽样路径长度与路径长度参数 k 相同。另外,虽然抽样路径的长度不断在增长,但是由于 SPAC 启发式算法在对于每条路径抽样时会更早对其进行判定,所以代表 SPAC 方法的三角形曲线在 k 增大时趋于平缓。对比无启发式方法的抽样和 PRISM 方法,其增长速度更加缓慢,所以也避免了大部分非必须的抽样与路径验证,节省了时间,进一步提高了在超长路径上验证性质的效率。

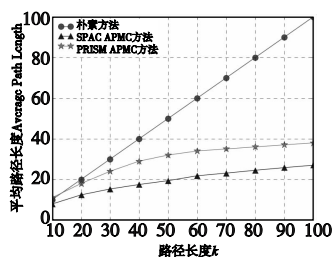


图2 抽样路径的平均长度比较

5 结束语

本文提出了一种启发式的统计模型检测算法。它在验证超长路径的情况下,能够在对路径进行验证的同时保存最短可

判前缀的信息,并在后续的随机有限路径抽样过程中,直接根据已知的最短可判前缀判定该路径是否满足给定性质,避开路径验证的过程,从而加快验证速度。通过实验发现,在越长的路径上验证性质,相对于没有采用启发式算法的方法,该算法能够起到的效果就越明显。在下一步研究工作中,将会考虑将该算法应用到一些需要在超长路径上验证的系统,并继续完善统计模型检测工具 SPAC 对于各类概率模型的支持,如连续时间马尔可夫链(continuous-time markov chain,CTMC)、马尔可夫决策过程(markov decision process,MDP)等。

参考文献:

- [1] Kwiatkowska M Z, Norman G, Parker D. PRISM 2.0: a tool for probabilistic model checking[C]//Proc of the 1st International Conference on Quantitative Evaluation of Systems. Netherlands: IEEE Press,2004:322-323.
- [2] Baier C, Kwiatkowska M Z. On the verification of qualitative properties of probabilistic processes under fairness constraints[J]. Information Processing Letters,1998,66(2):71-79.
- [3] Clarke E M, Grumberg O, Peled D. Model checking[M]. London: The MIT Press, 1997:35-50.
- [4] Kwiatkowska M Z, Norman G, Parker D. Probabilistic verification of Herman's self-stabilization algorithm[J]. Formal Aspects of Computing,2012,24(4-6):661-670.
- [5] Kwon Y, Agha G. Performance modeling of mobile sensor networks[C]//Proc of Ad hoc, Mobile, and Wireless Networks. Mexico: Springer, 2007:262-272.
- [6] Jha S K, Cslarke E M, Langmead C J, et al. A Bayesian approach to model checking biological systems[C]//Proc of Computational Methods in Systems Biology. Italy:Springer,2009:218-234.
- [7] Heath J, Kwiatkowska M Z, Norman G, et al. Probabilistic model checking of complex biological pathways[J]. Computational Methods in Systems Biology,2008,391(3):239-257.
- [8] Zuliani P, Platzer A, Clarke E M. Bayesian statistical model checking with application to stateflow/simulink[J]. Formal Methods in System Design,2013,43(2):338-367.
- [9] Younes H, Simmons R G. Statistical probabilistic model checking with a focus on time-bounded properties[J]. Information and Computation,2006,204(9):1368-1409.
- [10] Younes H, Kwiatkowska M Z, Norman G. Numerical vs. statistical probabilistic model checking[J]. International Journal on Software Tools for Technology Transfer,2006,8(3):216-228.
- [11] Herault T, Lasseigne R, Magniette F. Approximate probabilistic model checking[C]//Proc of Verification, Model Checking and Abstract Interpretation. Italy: Springer, 2004:73-84.
- [12] Baier C, Katoen J P. Principles of model checking[M]. London: The MIT Press, 2008:745-908.
- [13] Wald A. Sequential tests of statistical hypotheses[J]. The Annals of Mathematical Statistics,1945,16(2):117-186.
- [14] 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,2015,26(2):305-320.
- [15] 吴曲然,胡建宇,孙振宇,等. 海洋锋面统计模型检测法的改进与验证[J]. 厦门大学学报:自然科学版,2015,54(2):199-206.
- [16] Knuth D E, Yao A C. The complexity of nonuniform random number generation[M]//Algorithms and Complexity: New Directions and Recent Results. New York:Academic Press,1976:357-428.
- [17] Grubbs F. On designing single sampling inspection plans[J]. The Annals of Mathematical Statistics,1949,20(1):242-256.