SPS 系统的一种精确语义描述*)

戎 玫
(暨南大学深圳旅游学院 深圳 518053)

摘 要 规约模式系统 SPS 是根据性质的语义抽象而成的描述程序性质的表达模式,既能方便程序员使用,又有对应的时序逻辑表达式。但是,它现有的语义描述不够精确。首先介绍了规约模式系统及其现有的语义,并用它描述了实例的性质,接着采用模型检测工具 SPIN 验证了该系统表达的性质,通过对比验证结果及现有语义,给出了系统精确的语义描述。

关键词 规约模式系统,程序性质,形式化方法,SPIN

Exact Semantic of Specification Patterns System

RONG Mei
(Shenzhen Tourism College, Jinan University, Shenzhen 518053, China)

Abstract Specification patterns system is a pattern system, which is abstracted from program properties according to property's semantic. It is used to describe program properties. Programmers can easily use it. And it has corresponding LTL formula. At present, its semantic is inexplicit. In order to make it explicit, the article first introduces specification patterns system and its existing semantic, and describes an example's properties in it, then verifies properties expressed in the system by SPIN. Contrasting verifying results and existing semantic, the article shows exact semantic.

Keywords SPS, Program properties, Formal method, SPIN

目前对于软件系统的开发,如何保证软件的正确性受到了极大的关注。为了保证其正确性,工业界常用的方法是测试。然而测试只能证明软件有错,不能保证软件是正确的。对于一些对安全性要求很高的系统,用测试方法来保证软件的正确性显然不能满足要求,于是形式化的验证方法受到了广泛的关注。

形式化的验证方法是通过验证系统是否满足规定的性质,从而判断该系统是否正确。为了准确地描述系统的性质,一般采用时序逻辑公式,如 LTL,CTL。但是程序员很难正确地书写 LTL 或 CTL 表达式,于是美国内布拉斯加州大学的 Matthew Dwyer 教授于 1998 年提出了规约模式系统 SPS (Specification Patterns System)[1]。该系统是根据性质的语义抽象而成的一种表达模式,独立于各种时序逻辑语言,也能方便地转化成时序逻辑表达式。它主要用来描述程序性质,由性质模式和范围组成。通过此规约模式系统,程序员可以方便地描述各种程序性质,描述的性质也能方便地转化成时序逻辑表达式。在此系统的基础上,Matthew Dwyer 等人已经开发出验证 JAVA 程序的形式化验证工具 Bandera[2,3]。

在对程序形式化验证的过程中,前提条件是表达的性质是正确的。然而 Matthew Dwyer 给出的性质模式及范围的语义都较含糊,为了使其精确化,本文采用 SPIN 模型检测工具,对规约模式系统对应的 LTL 表达式进行分析验证,通过对比验证及现有的语义,最后给出了精确的语义描述。

本文将首先介绍模型检测工具 SPIN 和规约模式系统及现有的语义描述,接着用 SPIN 工具对该系统进行分析验证,最后给出了系统精确的语义描述。

1 模型检测工具 SPIN 简介

模型检测是一种基于有限状态模型并检验该模型是否具有期望性质的一种技术[4]。它是验证系统性质的算法和方法,通常采用搜索状态空间的方法,检测一个给定的模型是否满足某个用时序逻辑公式或自动机表示的性质。将所要验证的系统描述成有限的模型,然后建立系统性质的时序逻辑表达式,通过遍历模型上的所有可达状态,找出不满足逻辑表达式的路径,并提供违反公式的路径。模型检测是对状态空间的一种穷举搜索,而模型的有限性保证了该搜索可以在一定范围内中止。

模型检测主要应用于复杂软件系统的开发过程,作用是在开发早期发现和纠正错误。工作方式是:对软件系统或系统的某一部分建模,用某种形式化的方式说明系统应拥有的性质和满足的属性,通过对模型进行检测,判断软件系统是否拥有了这些期望的性质。模型检测涉及到大量的形式化工作,主要支持自动化的验证。

SPIN 是一种模型检测工具,用于验证 Promela 模型是否满足线性时序逻辑(LTL)描述的性质。下面简要介绍 SPIN 的验证过程以及功能。

1.1 SPIN 的验证过程

SPIN 的验证过程如下:首先描述系统的模型,经分析没有语法错误后,对系统的交互过程进行模拟,直到确认模型拥有预期的行为;然后通过命令产生一个用 C 语言编写的验证程序,经编译器编译后生成可执行的验证文件,执行该文件可得到验证结果。如果发现有违背性质的,则反馈给交互模拟

*)基金项目:重庆市自然科学基金(2006BB2259),江苏省高校自然科学基金(08KJB520010)。戎 玫 博士,副教授,CCF 高级会员,主要研究方向为形式化方法、软件工程、电子商务。

机，通过重现细节，可以发现引起错误的原因，图 1 描述了该验证过程。

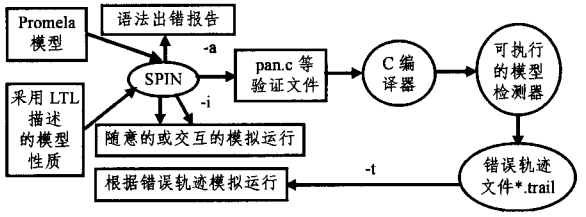


图 1　基于 SPIN 的验证过程

1.2　SPIN 的功能

SPIN 是一个广泛使用的用于分析并发系统逻辑一致性的软件工具，以 Promela(a Process Meta language)为建模语言，采用 LTL 描述模型性质。通过检测 Promela 模型是否满足 LTL 表达式，从而判断模型是否正确。对于错误的结果，将返回错误路径。它能够检测系统设计的逻辑错误。Promela 模型致力于描述系统中的交互过程，尽量抽象过程内部的细节，以此来减少系统的规模。系统中的过程之间主要通过通道传递消息，或者通过共享全局变量。

SPIN 区别于其他的模型检测器主要有 2 个方面：(1)用一种直观的、类 C 的建模语言 Promela 来描述系统，可以不用给出系统的执行细节，通过特殊符号可以引入 C 程序；(2)可以验证所有采用 LTL 公式表达的性质。

2　规约模式系统简介

规约模式系统包含性质模式和范围。本小节将首先介绍性质模式及其语义，接着给出范围的表示及相应的语义，最后用实例展示规约模式系统的表达能力。

2.1　性质模式的表示及其语义描述

规约模式系统包含的性质模式有 8 种，分别为：Absence，Universality，Existence，BoundedExistence，Response，Precedence，ChainResponse 及 ChainPrecedence。其中，Absence，Universality，Existence，Response 及 Precedence 是 5 种基本模式，其他性质模式通过这些模式来表达，如 BoundedExistence 表示为“Existence(n)”，ChainResponse 表示为“S，T Response P”，“P Response S，T”等，ChainPrecedence 表示为“S，T Precedence P”，“P Precedence S，T”等。

性质模式之间的关系如图 2 所示。

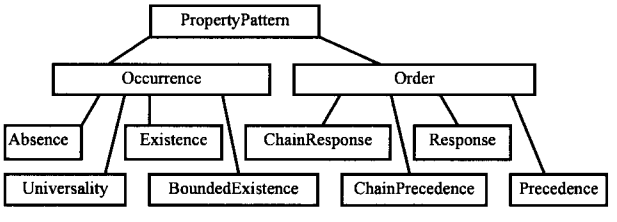


图 2　性质模型

5 种基本性质模式对应的语义如表 1 所示。

表 1　基本的性质模式及其语义描述

性质模式	语义描述
P Absence	在给定的范围内，状态 P 不会成立。
P Universality	在给定的范围内，状态 P 一直成立。
P Existence	在给定的范围内，状态 P 会在某些时刻成立。
S Response P	在给定的范围内，状态 P 的成立会导致 S 成立。
S Precedence P	在给定的范围内，状态 S 总是在 P 第一次成立之前成立。

2.2　范围的表示及其语义描述

范围分为 5 种：Globally，Before，After，Between-and 和 After-until。“Globally”表示“整个程序的执行过程”；“Before R”表示“状态 R 成立之前的时刻”；“After Q”表示“状态 Q 成立之后的时刻”；“Between R and Q”表示“状态 R 之后而 Q 之前的时刻”；“After R until Q”表示“状态 R 之后直到 Q 成立，如果 Q 不成立，则范围一直持续下去直到结束”。5 种范围的语义用图形表示，如图 3 所示。

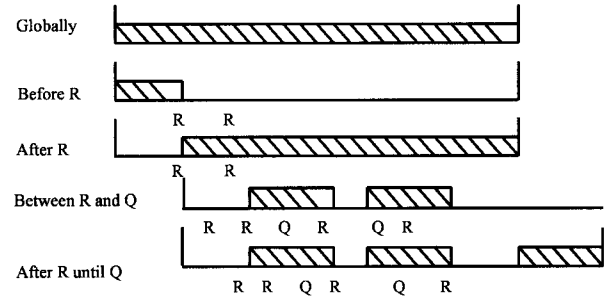


图 3　五种范围及其语义描述

2.3　实例描述

本小节以客户与厂商网上交易的规则为例，首先给出该规则对应的自然语言描述，接着给出基于 SPS 的表达形式。通过对比可以发现，SPS 能够描述这些性质，并且结构简单，容易理解。

2.3.1　非形式化的性质描述

假定一家电脑生产商可以提供网上销售服务，客户想通过网上购买产品，那么整个交易过程需满足如下性质：

(1)客户可以在线定购产品，任何订单在处理前需要做检查。对于有效的订单，客户将收到一个确认通知；对于不合理的订单，客户将收到取消通知。

(2)客户与生产商之间的事务处理过程应该满足如下规则。首先，客户只有在收到定购的产品后才付款。另外，生产商只有在确认客户会为定购的产品付款后，才会发出产品。于是，需要第三方监督机构银行的介入。为了让生产商开始发出产品，客户首先需将预付款存到银行的帐户上，接着银行将会通知生产商，说明预付款已经收到。客户任何时候取消订单，银行都将返回资金，如果客户收到产品，则银行将资金划到生产商的帐户上。

(3)为了能够满足订单，生产商首先需要检查货品清单。如果还有足够的产品，则定购的产品会被寄出，否则将会通知客户寄出时间。

2.3.2　基于 SPS 的性质描述

针对以上规则，下面给出了每个条款对应的 SPS 表达形式。其中 Customer，Manufacturer 及 Bank 分别指的是客户、生产商及银行，小数点后面是这些实体相应的动作，根据英文语义可以知道其表达的意义。

(1)((Customer. getConfirmNotification) Existence Globally

Xor (Customer. getCancelNotification) Existence Globally)

And ((Manufacturer. checkOrder) Precedence (Customer. getConfirmNotification) Globally

And (Manufacturer. checkOrder) Precedence (Customer. getCancelNotification) Globally)

(2)(Bank. deposit) Response (Customer. getConfirmNotification) Globally

(Manufacturer. startOrderProcessing) Response (Bank. deposit) Globally

(Bank. transfer) Response (Customer. getOrderFullfilled) Globally

(3)(Manufacturer. checkInventory) Precedence (Manufacturer. scheduleProducing) Globally

3 规约模式系统的验证

在规约模式系统中,每种性质都有对应的 LTL 表达式。那么,规约模式系统的语义与 LTL 表达的意义是否一致呢?由于从规约模式系统逐步推导出 LTL 表达式的过程很复杂,所以本文利用 SPIN 工具直接对 LTL 表达式进行验证。首先选取一个简单的模型,使用规约模式系统描述该模型的性质。本文仅以"(S,T)Precedence (P) Globally"性质为例,然后用 SPIN 工具验证该性质对应的 LTL 表达式,通过不断地调整 S、T 及 P 三种状态之间的相对顺序,对比验证结果及系统的语义,从而判断两者是否一致。

选取的模型用 Promela(SPIN 的模型描述语言)表达如下:

```
#define limit 20
byte I=0;
active proctype main(){
    do
    ::(I<limit)->I=I+1;
    ::(I==limit)->break;
    od;}
```

对于"(S,T)Precedence (P) Globally"性质,可以按下述方法选取测试用例,用来验证 Precedence 模式和 Globally 的语义。首先假定 S 与 T 同时发生,验证(S,T)性质对与 P 的相对顺序,可以选取 3 组测试用例,分为 S,T 发生在 P 之前、S,T 发生在 P 后,以及 S,T 及 P 同时发生;然后假定(S,T)性质对在 P 之前发生,验证 S 与 T 之间的相对顺序,选取的测试用例分为:S 在 T 之前发生、S 与 T 同时发生、S 在 T 之后发生;最后选取测试用例,使(S,T)性质对在 P 之后发生。验证结果如表 3。

表 2 Precedence 模式及其对应的 LTL 表达式

(S,T) Precedence (P)	Globally	◇P→(! PU (S & ! P& o(! P U T)))
	Before R	◇R→(! PU (R | (S & ! P& o(! P U T))))
	After Q	([]! Q) | (! Q U (Q &◇ P→ (! P U (S & ! P& o(! P U T)))))
	Between Q and R	[]((Q &◇ R)→ (! P U (R | (S & ! P& o(! PU T)))))
	After Q until R	[](Q→(◇ P→ (! P U (R | (S & ! P& o(! PU T))))))

表 3 验证结果

Test1:…(STP)… Pattern:S,T Precedence PScope:Globally S:I==7,T:I==7,P:I==7 期望结果:Violation 实际结果:Violation	Test2:…(ST),P… Pattern:S,T Precedence PScope:Globally S:I==6,T:I==6,P:I==7 期望结果:No Violation 实际结果:No Violation
Test3:…P,(ST)… Pattern:S,T Precedence PScope:Globally S:I==8,T:I==8,P:I==7 期望结果:Violation 实际结果:Violation	Test4:…S,(TP)… Pattern:S,T Precedence PScope:Globally S:I==5,T:I==7,P:I==7 期望结果:Violation 实际结果:No Violation
Test5:…T,S,P… Pattern:S,T Precedence PScope:Globally S:I==6,T:I==5,P:I==7 期望结果:Violation 实际结果:Violation	Test6:…P,T,S… Pattern:S,T Precedence PScope:Globally S:I==9,T:I==8,P:I==7 期望结果:Violation 实际结果:Violation
Test7:…P,S,T… Pattern:S,T Precedence PScope:Globally S:I==5,T:I==7,P:I==7 期望结果:No Violation 实际结果:No Violation	Test8:…T..S…… Pattern:S,T Precedence P..Scope:Globally S:I==5,T:I==3,P:I==22 期望结果:Violation 实际结果:No Violation
Test9:…(ST)… Pattern: S, T Precedence P.. Scope: Globally S:I==5,T:I==5,P:I==22 期望结果:Violation 实际结果:No Violation	Test10:…S,T…… Pattern: S, T Precedence PScope: Globally S:I==5,T:I==6,P:I==22 期望结果:Violation 实际结果:No Violation

4 精确的语义描述

根据 SPIN 的验证结果及现有的语义描述,本文重新定义了规约模式系统的语义描述。性质模式的语义如表 4 所示。

表 4 性质模式及其精确的语义描述

性质模式	语义描述
P Absence	在给定的范围内,状态 P 不会成立。
P Existence	在给定的范围内,状态 P 在某些时刻会成立。
P Existence(2)	在给定的范围内,状态 P 在会成立二次。
P Universality	在给定的范围内,状态 P 一直成立。
S Precedence P	1)在给定的范围内,如果状态 P 成立时,那么在 P 之前的时刻 S 已经成立,或者 S 和 P 同时成立。 2)在给定的范围内,如果状态 P 不成立时,不论 S 情况如何,都符合此性质模式的要求。
S Response P	1)在给定的范围内,如果状态 P 成立时,必然会导致 S 成立,即 S 在 P 之后成立,或者与 P 同时成立。 2)在给定的范围内,如果状态 P 不成立时,不论 S 情况如何,都符合此性质模式的要求。
S,T Precedence P	1)在给定的范围内,如果状态 P 不成立时,不论 S 与 T 情况如何,都符合此性质模式的要求。 2)在给定的范围内,如果状态 P 成立时,S,T 及 P 应该满足如下几种关系: S 先成立,之后 T 成立,最后 P 才成立;S 与 T 同时成立,之后 P 才成立;S 先成立,之后 T 与 P 同时成立。
P Precedence S,T	1)在给定的范围内,如果 S 与 T 都成立,并且满足 S 在 T 之前或与 T 同时成立,P 必定在 S 之前或与 S 同时成立。 2)在给定的范围内,如果 S 或 T 不成立,S 在 T 之后成立,不论 P 情况如何,都符合此性质模式的要求。
P Response S,T	1) 在给定的范围内,如果 S 与 T 都成立,并且满足 S 在 T 之前或与 T 同时成立时,此 S-T 序列对的成立必然导致 P 成立(即 S 与 T 满足上述条件后,必然导致 P 在 T 之后或与 T 同时成立)。 2) 在给定的范围内,如果 S 或 T 不成立、S 在 T 之后成立,不论 P 情况如何,都符合此性质模式的要求。
S,T Response P	1) 在给定的范围内,如果状态 P 不成立,不论 S 与 T 情况如何,都符合此性质模式的要求。 2)在给定的范围内,如果状态 P 成立,必然导致 S-T 序列对的成立(即 S 必须在 P 之后成立,或与 P 同时成立)。P 与 S-T 序列对应该满足如下关系: P 先成立,接着 S 先成立,最后 T 成立;P 先成立,接着 S 与 T 同时成立;P 与 S 同时成立,之后 T 成立。

范围的语义描述如下所示。

Globally:表示了整个程序的执行过程。

Before R：当 R 成立时，表达的范围是 R 第一次成立之前的所有时刻，且不包括第一次成立的时刻；当 R 不成立时，表达的范围不存在。

After R：当 R 成立时，表达的范围是 R 第一次成立之后的所有时刻，且包括第一次成立的时刻；当 R 不成立时，表达的范围不存在。

Between R and Q：当 Q 在 R 之后成立时，每个 R 成立的时刻都与紧随其后的、第一次 Q 成立的时刻，构成 R 成立之后到 Q 成立之前的 R-Q 区间，包括左边界但不包括右边界；当 R 或 Q 不成立、Q 在 R 之前或与 R 同时成立时，表达的范围不存在。

After R until Q：当 Q 在 R 之后成立时，每个 R 成立的时刻都与紧随其后的、第一次 Q 成立的时刻，构成 R 成立之后到 Q 成立之前的 R-Q 区间，包括左边界但不包括右边界；当 R 成立，而 Q 不成立时，表达的范围是从 R 成立开始直到程序结束的区间，包括 R 成立的时刻；当 Q 成立而 R 不成立、Q 在 R 之前或与 R 同时成立时，表达的范围不存在。

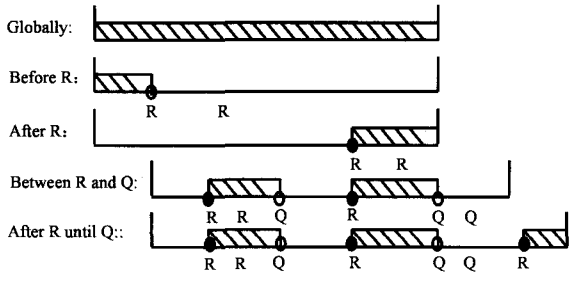


图 4　范围及其精确的语义描述

“After R until Q”与“Between R and Q”的区别仅在于：当 R 成立而 Q 不成立时，“Between R and Q”不能表达合理的范围，而“After R until Q”表达的范围存在。用图形描述 5 种范围的语义如图 4 所示。

结束语　本文采用 SPIN 模型检测工具，对规约模式系统对应的 LTL 表达式进行了分析验证。通过对比验证及现有的语义，给出了精确的语义描述。这种思想，文献[5]已经做了尝试，不过该文献只提出了错误，没有给出解决方法。本文在它的基础上，完成了对剩下的模式的验证，并给出了规约模式系统精确的语义描述。

参 考 文 献

[1] Dwyer M B, Avrunin G S, Corbett J S. A System of Specification Patterns[EB]. http://www.cis.ksu.edu/santos/spec_patterns. 1998

[2] Corbett J C, Dwyer M B, Hatcliff J, et al. A language framework for expressing checkable properties of dynamic software [C]// Havelund K, Panix J, Visser W, eds. LNCS 1885. Berlin: Springer-Verlag, 2000: 205-223

[3] Hatcliff J , Dwyer M . Using the Bandera Tool Set to Model - Check Properties of Concurrent Java Software//Larsen K. G, Nielsen M, eds. LNCS 2154. Berlin: Springer-Verlag, 2001. 39-58

[4] 戎玫，张广泉. 模型检测新技术研究. 计算机科学，2003，30(5)：101-104

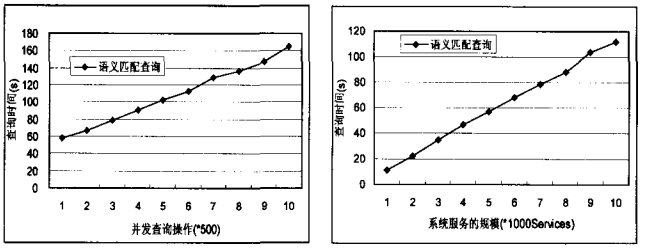
[5] Salamah S, Gates A, Roach S, et al. Verifying Pattern-Generated LTL Formulas: A Case Study // Godefroid P, ed. LNCS 3639. Berlin: Springer-Verlag, 2005: 200-220

[6] Yu Jian, Phan manh Tan, Han Jun, et al. Pattern based property specification and verification for service composition[C]//Aberer K, ed. LNCS 4255. Berlin: Springer-Verlag, 2006: 156-168

(上接第 268 页)

从 500 变化到 5000；(2)服务规模变化(1000～10000 个服务)，并发查询数量固定(1000 个)。实验所采用的查询语句类似于图 4，其中的约束条件是随机生成的。实验所报告的语义匹配消耗的时间如图 6 所示。

从图 6 可以看到，支持语义的服务匹配策略其查询所需时间随并发查询操作数量或系统服务的规模增大均近似为线性增长。在系统规模为 10000 个服务时，平均每增加 500 个并发查询操作，所需时间增加 11.221s。当固定并发查询数量为 1000 时，查询时间以 11.218s/1000Services 为起点，以 11.198s/1000Services 的速率增长。



(a)系统服务的规模：10000Service　(b)固定并发查询操作所数量：1000

图 6　语义匹配策略性能

语义匹配策略扩展了服务描述与查询条件的灵活性，从而在很大程度上提高了服务发现的表达能力和智能性。但与系统的精确/部分匹配操作的性能相比，语义匹配策略在提高了匹配能力的同时也带来了性能下降、匹配度规则难以制定的问题。

结束语　本文提出了一种支持服务多样性、异构性及上下文动态变化的服务描述模型 SDMDHD，同时设计实现了高效的存储结构和支持精确匹配/部分匹配、语义匹配的服务匹配策略。下一步，我们将进一步研究服务模型的语义支持，提高语义匹配策略的性能和智能性，并将工作拓展到 MANET 环境下，同时将研究多种服务发现协议的互操作问题。

参 考 文 献

[1] Guttman E. Service Location Protocol: Automatic Discovery of IP Network Services. IEEE Internet Computing, 1999, 3(4)

[2] Bluetooth SIG. Specification of the Bluetooth System, 2004

[3] Adjie-Winoto W, Schwartz E, Balakrishnan H, et al. The Design and Implementation of an Intentional Naming System // 17th ACM Symposium on Operating System Principles. 1999

[4] Czerwinski S E, Zhao B Y, Hodes T D, et al. An Architecture for a Secure Service Discovery Service // 5th Annual International Conference on Mobile Computing and Networks. 1999

[5] Chakraborty D, Joshi A, Yesha Y, et al. GSD: A novel group-based service discovery protocol for MANETS // IEEE International Conference on Mobile and Wireless Communications Networks. 2002

[6] Klein M, Konig-Ries B, Obreiter P. Service rings - a semantic overlay for service discovery in ad hoc networks // DEXA Workshops. 2003: 180-185

[7] Robinson R, Indulska J. Superstring: A Scalable Service Discovery Protocol for the Wide Area Pervasive Environment // Proc. of the 11th IEEE International Conference on Networks. Sydney, September 2003