基于SPIN的Andrew Secure RPC协议并行攻击模型检测

肖美华 朱 科 马成林
(华东交通大学软件学院 南昌 330013)

摘 要 Andrew Secure RPC协议具有身份认证和秘钥交换功能,其因简洁明了而被广泛应用于对称密钥加密体系中。模型检测技术具有高度自动化的优点,在协议安全性验证领域得到广泛应用,但模型检测方法只能检测到一轮协议会话中存在的攻击,难以检测到多轮并行会话中存在的并行攻击。针对Andrew Secure RPC协议运行环境中存在的并行性与可能出现的安全隐患,提出了组合身份建模方法。该方法运用著名的SPIN模型检测工具,对Andrew Secure RPC协议进行模型检测,从而得到攻击序列图,成功发现并行反射攻击和类缺陷攻击。上述组合身份建模方法为复杂环境下协议的模型检测提供了新的方向。

关键词 Andrew Secure RPC协议,模型检测,SPIN,组合身份建模,并行攻击
中图法分类号 TP393.08 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.7.022

Model Checking of Parallel Attack in Andrew Secure RPC Protocol Based on SPIN

XIAO Mei-hua ZHU Ke MA Cheng-lin
(School of Software, East China Jiaotong University, Nanchang 330013, China)

Abstract Andrew Secure RPC protocol is a kind of protocol with functions of identity authentication and key exchange, which is widely used in symmetric cryptography because of its conciseness. Model checking technology is widely used in verification of protocols due to its high automation, however, there is also a disadvantage in model checking technology that it can only find out attacks in single round of protocol session, which is hard used in multi rounds of protocol session. We proposed a modeling method, called combinatorial identity modeling method, which uses SPIN to verify Andrew Secure RPC protocol in consideration of the parallel environment and potential danger in Andrew Secure RPC protocol. According to the research conclusion, we found out two kinds of attacks which are reflection attack and type flaw attack in Andrew Secure RPC protocol. With this conclusion, we offered a new direction in model checking research in verifying protocol under complicated environment.

Keywords Andrew Secure RPC protocol, Model checking, SPIN, Combinatorial identity modeling method, Parallel attacks

安全协议是以密码学为基础的消息交换协议,其目的是在网络环境中提供各种安全服务。1989年,Satyanarayanan提出了Andrew Secure RPC协议,作为一个基于对称密钥体系的、具有身份认证和秘钥交换双重功能的协议,它开始被广泛使用。1989年,Burrows、Abadi、Needham[1]首先用BAN逻辑对Andrew Secure RPC协议进行了验证,并发现了协议中存在的缺陷;1996年,Lowe[2]使用CSP模型和基于CSP的模型检验工具FDR再次对Andrew Secure RPC协议进行了分析,发现该协议存在并行会话攻击。基于前人的研究,周清雷等[3]又运用串空间方法对Andrew Secure RPC协议进行了分析,并且发现了其中的漏洞。

模型检测技术由于其自动化程度高,并能在系统不满足性质时提供反例路径,因此在工业界比演绎证明更受推崇。SPIN[4]作为一种著名的协议模型检测验证工具,因其高度自动化和简洁明了的语言而备受关注,G. J. Holzmann因开发SPIN工具在2002年荣获ACM的"Software System Award"[4]。因为模型检测无法达到完备测试的目的,在协议的模型检测过程中往往只能检测出一轮协议会话中所存在的漏洞,所以在不同主体多轮并行开展协议会话的情况下,模型检测显得很乏力。而本文基于SPIN模型检测工具,针对Andrew Secure RPC协议存在的并行会话特性,对Maggi[6]所提出的建模基本方法做出改进[7],在对Andrew Secure RPC协议进行建模后,验证Andrew Secure RPC协议是否满足身份认证和密钥交换属性,并且成功得出并行反射攻击和类缺陷攻击序列图。

本文第1节对SPIN模型检测工具进行简单介绍;第2

到稿日期:2014-06-25 返修日期:2014-09-14 本文受国家自然科学基金(61163005),计算机软件新技术国家重点实验室开放课题(KFKT2012B18),江西省高校科技落地计划项目(KJLD13038),江西省自然科学基金(2010GZS0150,20132BAB201033)资助。
肖美华(1967-),男,博士,教授,博士生导师,主要研究方向为信息安全、软件形式化方法,E-mail:xiaomh@ecjtu.edu.cn;朱 科(1989-),男,硕士生,主要研究方向为信息安全、模型检测技术,E-mail:jovezk@126.com;马成林(1989-),女,硕士生,主要研究方向为软件形式化方法,E-mail:mamcl2014@gmail.com。

节对 Andrew Secure RPC 协议进行介绍；第 3 节为 Andrew Secure RPC 协议组合身份建模过程；第 4 节为身份认证属性和密钥交换属性的 LTL 描述；第 5 节对实验结果进行分析；最后总结全文。

1 模型检测及 SPIN 工具

模型检测(model checking)是一种很重要的自动验证技术。它通过显式状态搜索或隐式不动点计算来验证有穷状态并发系统的模态/命题性质。由于模型检测可以自动执行，并能在系统不满足性质时提供反例路径，因此在工业界比演绎证明更受推崇。

SPIN (Simple Promela Interpreter)是用来分析并发系统逻辑一致性的工具，尤其在数据通信协议的模型检测中显得更加强大。在 SPIN 模型检测工具中集成了 promela 语言编译环境，并发系统模型 M 将会被描述成 promela 语言，而系统需要满足的性质将被描述成为 LTL(线性时态逻辑)公式 φ，M 和 φ 会放入 SPIN 工具中，SPIN 工具会自动验证模型 M 是否满足公式 φ，即 $M \vdash \varphi$。

具体验证流程如图 1 所示[8]。

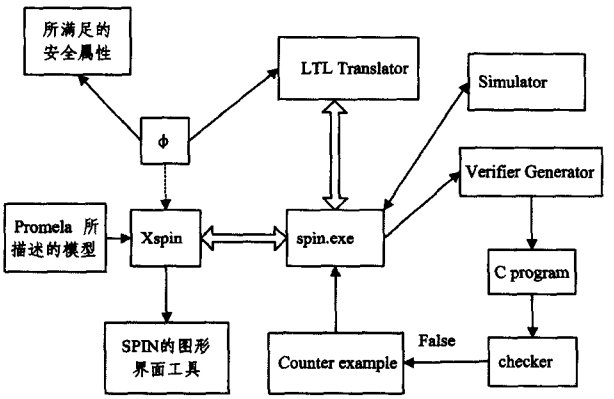

图 1 SPIN 的模型检测流程

2 Andrew Secure RPC 介绍

Andrew Secure RPC 协议主体只有 A 和 B 两方，所用的加密方法为 A 和 B 的共享密钥加密，协议的功能主要有两个：(1)双方的身份认证；(2)A 和 B 交换新的共享密钥。

Andrew Secure RPC 协议运行流程如下：

Msg1 A->B:{Na}Kab

Msg2 B->A:{Na+1,Nb}Kab

Msg3 A->B:{Nb+1}Kab

Msg4 B->A:{Kab′,Nb′}Kab

协议基于对称密钥算法，整个身份认证和密钥交换过程没有可信第三方参与。Msg1 到 Msg3 中，A 和 B 使用已有的共享密钥 Kab 加密消息，并且确认了相互的身份。然后主体 B 产生一个新的会话密钥 Kab′以及新的随机数 Nb′(用于后续通信的一个初始序列号)，用 Kab 加密后传送给 A，从而在 A 和 B 之间建立一个新的会话密钥 Kab′。

3 Andrew Secure RPC 协议组合身份建模过程

Andrew Secure RPC 协议的建模过程分为诚实主体建模和攻击者建模。攻击者的建模遵循 Dolev-Yao 攻击模型[9]。

3.1 诚实主体建模

本文把协议主体分为发起者和响应者两种类型，所以我们为模型创造两个进程，分别为发起者进程 proctype PIni() 和响应者进程 proctype PRes()。本文还为模型定义了两个消息通道，这里因为协议的特殊性和 Dolev-Yao 模型中攻击者能力的强大性，本文分别定义了发起者 PIni 和攻击者 EVE 的消息通道 cIE、响应者 PRes 和攻击者 EVE 的消息通道 cRE。

根据协议中传输的消息数，两个通道定义如下：

chan cIE=[0] of {mtype,mtype,mtype,mtype};

chan cRE=[0] of {mtype,mtype,mtype,mtype};

接着本文对协议中所有用到的消息类型做枚举说明：

mtype={A,B,Na1,Na,Na1_,Na_,Na__,Nb1,Nb,Nb1_,Nb_,Nb__,Kab,Kab_A,Kab_B,R,none,x};

其中，A,B 代表诚实主体 A 和 B；Na,Na_,Na__都是 A 所用的随机数，这里符号 Na 为 A 作为发起者时所用的随机数，Na_为 A 作为响应者时所用的随机数，Na__表示 A 作为响应者最后交换所用的全新随机数，Kab_A 为 A 作为响应者交换所用的全新共享密钥；Nb 为 B 作为响应者时所用的随机数，Nb_为 B 作为发起者时所用的随机数，Nb__是 B 作为响应者交换所用的全新随机数，Kab_B 为 B 作为响应者最后交换所用的全新共享密钥。这里随机数枚举的顺序也是有根据的，因为在消息通道发送中，如果 mtype 枚举类型数据完成加 1 操作(如：Na+1)，则这个 mtype 类型数据将会变为原 mtype 类型枚举顺序的前一个数据(即：Na+1 在通道中显示的数据为 Na1，这里表示为 Na+1)，因为协议中随机数会执行加 1 操作后传输，所以枚举中随机数顺序的定义为：随机数的前一个枚举对象表示为该随机数加 1。R 为不可识别主体，Kab 为 A 和 B 的原始共享密钥，Kab_为 A 和 B 交换后的新共享密钥，none 为空消息，x 可代表任意诚实主体。

根据 Andrew Secure RPC 协议的消息交换特点，本文对发起者进程和响应者进程做以下定义：

```
proctype PIni (mtype self;mtype party;mtype noncel)
{
mtype g1;mtype g2;mtype g3;
    atomic {
        IniRunning(self,party);
        cIE!self,noncel,none,Kab;
    }
    atomic {
        cIE?g2,eval(noncel+1),g1,eval(Kab);
        IniCommit(self,party);
        cIE!self,g1+1,none,Kab;
    }
    atomic {
        cIE?g3,Kab_,N,eval(Kab);
        Inifinished(self,party);
    }
}
```

发起者进程中发起者一共完成了两次发送操作和两次接

收操作，其中消息结构中发送消息的第一个元素 self 表示以 A 或者 B 的身份发送这条消息，中间的为传输的实际消息元素，none 表示这个位置无任何消息，最后的元素 Kab 表示用 A 和 B 的共享密钥加密。

把接收操作和下一个发送操作(如果有的话)放在一个原子步中，用 atomic 括起来，表示这两个步骤是一个原子步同时完成，这样可以有效减少状态空间。这里因为所用的是发起者和攻击者一个通道，响应者和攻击者一个通道，合法主体所发送的消息必定是发给攻击者的。发送者进程定义中有 3 个参数，分别为 self、party、none1，分别代表发起者的身份 self、发起者所要发起对话的对象 party、发起者所用到的随机数 none1。IniCommit(self, party)表示发起者 self 提交了与对象 party 的对话，Inifinished(self, party)表示发起者 self 收到了对象 party 发送过来的新共享密钥，以下是对这些函数的宏定义，这些函数的具体作用在第 4 节的 LTL 属性描述过程中再做详细介绍。

```
#define Inifinished(x,y) if\
::((x==A)&&(y==B))->InifinishedAB=1\
::((x==B)&&(y==A))->InifinishedBA=1\
::else skip \
```

其余函数宏定义如：Resfinished(x,y)，IniRunning(x,y)，IniCommit(x,y)，ResRunning(x,y)，ResCommit(x,y)与之相似，不再赘述。

下面是对属性参数的初始化操作：

```
bit InifinishedAB=0;bit ResfinishedAB=0;
bit IniRunningAB=0;bit IniCommitAB=0;
bit ResRunningAB=0;bit ResCommitAB=0;
bit InifinishedBA=0;bit ResfinishedBA=0;
bit IniRunningBA=0;bit IniCommitBA=0;
bit ResRunningBA=0;bit ResCommitBA=0;
```

响应者的进程描述和发起者相类似，不再具体介绍。

3.2 攻击者建模

在 Dolev-Yao 模型中攻击者无所不能，它可以截获所有消息通道内的消息，并且重组或者转发给任何主体。尽管攻击者如此强大，但是也不是万能的，比如攻击者不能在不知道密钥的情况下解密加密过的消息，也不能在不知道密钥的情况下，构造一份具有明确语义并且和已知密文相同类型的密文，而且攻击者也不可预测主体所用的随机数等等。因为 Andrew Secure RPC 协议是基于对称密钥的协议，Kab 为 A 和 B 的共享密钥，所以所有用 Kab 加密过的密文攻击者都不可以解密得到明文，攻击者只能转发这些消息。这里沿用 Maggi 所用的静态分析方法，列举出攻击者所能截获到的所有消息 M 和合法主体所能接收到的所有消息 N，然后两个集合取交集 $M \cap N$ 便得到了攻击者所能存储的有用的知识元素。

求出 M 和 N 的具体过程是分析发起者和响应者中的接收和发送语句，这里不做具体介绍。

现在把攻击者可能截获到的 16 条消息和诚实主体能够接收到的 14 条消息做交集，得出对攻击者而言有用的消息，一共 10 条，分别是：

```
{Nb_}Kab,{Na}Kab,{Na_+1}Kab
{Nb+1}Kab,{Kab_A,Na_}Kab
{Kab_B,Nb_}Kab,{Na+1,Na_}Kab
{Na+1,Nb}Kab,{Nb_+1,Na_}Kab
{Nb_+1,Nb}Kab
```

前 4 条消息是响应者能够接收的且攻击者可能得到的消息，后 6 条消息为发起者能够接收到的且攻击者可能得到的消息。

然后设置攻击者的知识系统，分别对攻击者的知识函数进行宏定义：

```
#define k(x1) if\
::(x1==Nb_)->kNb_=1\
::(x1==Na) -> kNa=1 \
::(x1==Na_+1)->kNa1_=1 \
::(x1==Nb+1)->kNb1=1\
::else skip\
fi
#define k1(x1,x2) if\
::(x1==Nb_+1&& x2 ==Na_) -> k_Nb1_Na_=1\
::(x1==Na+1&& x2==Nb)->k_Na1_Nb=1 \
::(x1==Na+1&& x2==Na_)->k_Na1_Na_=1\
::(x1==Nb_+1&&x2==Nb)->k_Nb1_Nb=1\
::(x1==Kab_A)->kab_A=1 \
::(x1==Kab_B)->kab_B=1 \
::else skip \
fi
```

在这些知识函数的宏定义中，定义了攻击者在截取到有用消息后，把相应的知识变量定为 1，表示攻击者已经获取该条消息的信息，可以进行转发。比如：攻击者接收到类型如{Nb_}Kab 的消息后，知识函数 k(x1)判定 x1==Nb_成立，则知识变量 kNb_赋值为 1。

攻击者截获消息和转发消息的进程定义如下：

```
proctype PI() {//变量初始化过程省略
do
    ::cIE!x,Na+1,Na_,(k_Na1_Na_->Kab;R)
    ::cIE!x,Na+1,Nb,(k_Na1_Nb->Kab;R)
    ::cIE!x,Nb_+1,Na_,(k_Nb1_Na_->Kab;R)
    ::cIE!x,Nb_+1,Nb,(k_Nb1_Nb->Kab;R)
    ::cIE!x,Kab_A,Na_,(kab_A->Kab;R)
    ::cIE!x,Kab_B,Nb_,(kab_B->Kab;R)
    ::cRE!x,Na_+1,none,(kNa1_->Kab;R)
    ::cRE!x,Na,none,(kNa->Kab;R)
    ::cRE!x,Nb_,none,(kNb_->Kab;R)
    ::cRE!x,Nb+1,none,(kNb1->Kab;R)//转发消息
  ::d_step {
      cRE?_,x1,x2,x3->k1(x1,x2);
      x1=0;x2=0;x3=0;
      }
  ::d_step {
    cIE ?_,x1,x2,x3->k(x1);
    x1=0;x2=0;
      }//截获消息
  od
}
```

现在合法主体和攻击者建模已经完成。下面定义 init 初始化进程：

```
init
{
atomic{
    if
    ::run PIni(A,B,Na)
    ::run PIni(B,A,Nb)
    fi;
    if
  ::run PRes(A,B,Na_)
  ::run PRes(B,A,Nb_)
    fi;
  run PI();
}
}
```

因为该协议中两个主体 A 和 B 既能充当发起者，又能充当响应者，且当一个主体充当发起者开始一轮协议运行的同时，这个主体还能充当响应者进行另一轮协议的运行，这便是并行会话协议的复杂之处。本文定义 A 既能以发起者身份开始一轮协议，同时也能以响应者的身份开始另一轮协议，B 与之相同，这样任何参与协议诚实主体的身份情况便全部考虑在内，这就是组合身份建模的主体思想。攻击者作为单独进程和其他进程并行运行。

4 协议身份认证属性和密钥交换属性的 LTL 公式描述

Andrew Secure RPC 协议所要达到的功能为通信双方的身份认证和密钥交换功能。本文把协议运行分为两种情况：(1)A 以发起者身份和 B 以响应者身份完成一轮会话，达到身份认证和密钥交换功能。(2)B 以发起者身份和 A 以响应者身份完成一轮会话，达到身份认证和密钥交换功能。

在 3.1 节诚实主体建模中已经给出了所有属性描述函数和属性描述变量的定义以及初始化操作，下面为了方便描述属性，把各个属性变量定义为字母的形式：

```
#define p IniCommitAB  #define q ResRunningAB
#define r ResCommitAB  #define s IniRunningAB
#define m InifinishedAB  #define n ResfinishedAB
#define p_ IniCommitBA  #define q_ ResRunningBA
#define r_ ResCommitBA  #define s_ IniRunningBA
#define m_ InifinishedBA  #define n_ ResfinishedBA
```

以第一种情况：A 以发起者身份和 B 以响应者身份完成一轮会话，达到身份认证和密钥交换功能为例，其属性描述如下：

([] ((<> !p) || (!p U q))) && ([] ((<> !r) || (!r U s))) && ([] ((<> !m) || (!m U n)))

其中，p 表示 IniCommitAB，意义为发起者 A 提交了与响应者 B 的一次对话，q 表示 ResRunningAB，意义为响应者 B 发起了与发起者 A 的一次对话。那么只有在 B 发起与 A 的对话后，A 才能提交与 B 的对话，这样 B 的身份便得到了 A 的认证。

上述情况为 A 作为发起者，B 作为响应者的情况。第二种情况：A 作为响应者，B 作为发起者的情况与之相类似，具体过程不做描述，LTL 公式的描述为：

([] ((<> !p_) || (!p_ U q_))) && ([] ((<> !r_) || (!r_ U s_))) && ([] ((<> !m_) || (!m_ U n_)))

所以把这两个 LTL 公式用"与"的逻辑符号"&&"联接后便是完整的 LTL 属性描述：

([] ((<> !p) || (!p U q))) && ([] ((<> !r) || (!r U s))) && ([] ((<> !m) || (!m U n))) && ([] ((<> !p_) || (!p_ U q_))) && ([] ((<> !r_) || (!r_ U s_))) && ([] ((<> !m_) || (!m_ U n_)))

把 LTL 公式输入 SPIN 的 LTL property manager 中，生成相应的 never claim，SPIN 工具会自动进行验证工作。

5 实验结果分析

在把模型和 LTL 公式输入 SPIN 验证工具后，系统自动进行验证，发生了属性违反，自动生成了相应的攻击路径，攻击序列如图 2 所示。

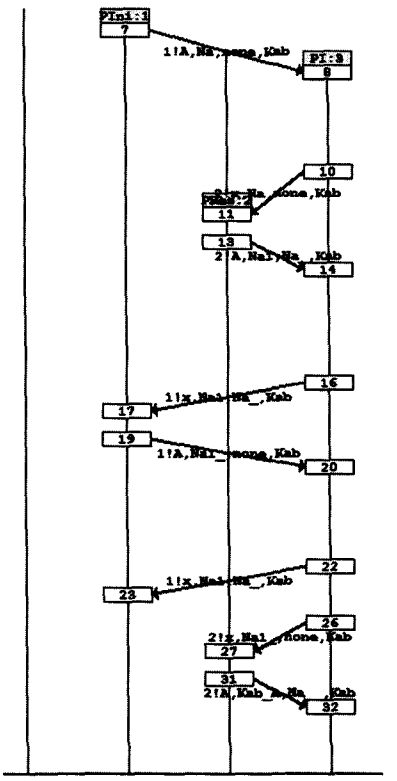

图 2 攻击序列

由上图所得的攻击序列为：

A->I(B):{Na}Kab

I(B)->A:{Na}Kab

A->I(B):{Na+1,Na_}Kab

I(B)->A:{Na+1,Na_}Kab

A->I(B):{Na_+1}Kab

I(B)->A:{Na+1,Na_}Kab

I(B)->A:{Na_+1}Kab

A->I(B):{Kab_A,Na_}Kab

这个攻击序列揭示了 Andrew Secure RPC 协议的两种攻击类型，第一种是并行反射攻击，攻击者通过不断反射 A 所发送过来的信息，完成了与 A 的两次协议运行，A 认为与 B 已经完成了两轮协议会话，但是 B 却没有参与任意一次协议

会话。第二种是类缺陷攻击，在A作为发起者完成身份验证后，攻击者转发了一条以前截获的信息{Na+1，Na_}Kab，这时A接收到这条信息，以为是新的共享密钥和新的随机数，没有做类型检测就确认了交换，把Na+1当做新的共享密钥，把Na_当做了新的随机数。

结束语 模型检测技术在网络协议的安全验证中起了重要的作用，由于模型检测的高度自动化和自动生成攻击路径等特点，使得模型检测方法被人们所熟知。但是由于网络技术的发展，协议运行的网络环境也十分复杂，协议参与者们可能会遇到多协议并行运行的情况；协议开发者为了使协议能够有更多的功能和更加强大的安全性，将协议设计得也更加复杂；这便给分析协议的学者们带来了更多的挑战，在模型检测的过程中，可能会忽略多轮协议并行运行的情况，使得并行会话中存在的攻击路径不能被发现；更加复杂的协议内容，使得建立的模型十分复杂，且容易发生状态爆炸问题[10,11]。本文所用的组合身份建模方法解决了两轮协议会话并行运行过程中的模型检测问题，并且这种建模方法简单明了，建模步骤清晰，代码可读性强，为以后复杂环境下协议的模型检测研究提供了参考。未来所要研究的是完善建模方法，使其应用在更加复杂的协议运行环境(三轮并行会话及以上)之中，并且拓展模型检测技术的应用领域[12]，增加模型检测技术的实用性。

参考文献

[1] Burrows M, Abadi M, Needham R M. A logic of authentication [J]. Series A, Mathematical and Physical Sciences, 1989, 426 (1871): 233-271

[2] Lowe G. Some new attacks upon security protocols[C]// CSFW, 1996. 1996: 162-169

[3] 周清雷，赵琳，赵东明. 基于串空间模型的 Andrew RPC 协议的分析与验证[J]. 计算机工程与应用，2007，43(13)：153-155
Zhou Qing-lei, Zhao Lin, Zhao Dong-ming. Analysis and verification of Andrew RPC protocol based on strand spaces[J]. Computer Engineering and Applications, 2007, 43(13): 153-155

[4] Holzmann G J. The model checker SPIN[J]. IEEE Transactions on software engineering, 1997, 23(5): 279-295

[5] 吴昌，肖美华，罗敏，等. 安全协议验证模型的高效自动生成[J]. 计算机工程与应用，2010，46(2)：79-82
Wu Chang, Xiao Mei-hua, Luo Min, et al. Effective automatic generation of verification model on security protocol[J]. Computer Engineering and Applications, 2010, 46(2): 79-82

[6] Maggi P, Sisto R. Using SPIN to verify security properties of cryptographic protocols [M] // Model Checking Software. Springer Berlin Heidelberg, 2002: 187-204

[7] Krawczyk U, Sapiecha P. Effective reduction of cryptographic protocols specification for model-checking with Spin[J]. Annales UMCS, Informatica, 2011, 11(3): 27-40

[8] Ruys T C, Holzmann G J. Advanced spin tutorial[M]// Model Checking Software. Springer Berlin Heidelberg, 2004: 304-305

[9] Dolev D, Yao A C. On the security of public key protocols[J]. IEEE Transactions on Information Theory, 1983, 29(2): 198-208

[10] 侯刚，周宽久，勇嘉伟，等. 模型检测中状态爆炸问题研究综述[J]. 计算机科学，2013，40(z6)：77-86，111
Hou Gang, Zhou Kuan-jiu, Yong Jia-wei, et al. Survey of State Explosion Problem in Model Checking[J]. Computer Science, 2013, 40(z6): 77-86, 111

[11] 李兴锋，张新常，杨美红，等. 基于 SPIN 的模块化模型检测方法研究[J]. 电子与信息学报，2011，33(4)：902-907
Li Xing-feng, Zhang Xin-chang, Yang Mei-hong, et al. Study on Modularized Model Checking Method Based on SPIN[J]. Journal of Electronics & Information Technology, 2011, 33(4): 902-907

[12] Yamada Y, Wasaki K. Automatic generation of SPIN model checking code from UML activity diagram and its application to Web application design[C]// 2011 7th International Conference on Digital Content, Multimedia Technology and its Applications (IDCTA). IEEE, 2011: 139-144

(上接第 73 页)

[9] Kalczynski P J, Kamburowski J. An improved NEH heuristic to minimize makespan in permutation flow shops [J]. Computers & Operations Research, 2008, 35(9): 3001-3008

[10] Farahmand R S, Ruiz R, Boroojerdian N. New high performing heuristics for minimizing makespan in permutation flowshops [J]. OMEGA-the International Jouranal of Management Science, 2009, 37: 331-345

[11] Sioud A, Gravel M, Gagne C. A genetic algorithm for solving a hybrid flexible flowshop with sequence dependent setup times [C] // 2013 IEEE Congress on Evolutionary Computation (CEC). 2013: 2512-2516

[12] Wang X, Tang L. A tabu search heuristic for the hybrid flowshop scheduling with finite intermediate buffers [J]. Computers & Operations Research, 2009, 36(3): 907-918

[13] Lin S W, Ying K C. Applying a hybrid simulated annealing and tabu search approach to non-permutation flowshop scheduling problems [J]. International Journal of Production Research, 2009, 47(5): 1411-1424

[14] Yagmahan B, Yenisey M M. A multi-objective ant colony system algorithm for flow shop scheduling problem[J]. Expert Systems with Applications, 2010, 37(2): 1361-1368

[15] Liao C J, Tjandradjaja E, Chung T P. An approach using particle swarm optimization and bottleneck heuristic to solve hybrid flow shop scheduling problem [J]. Applied Soft Computing, 2012, 12 (6): 1755-1764

[16] Taillard E. Some effective heuristic methods for the flowshop sequencing problem [J]. European Journal of Operational Research, 1990, 47: 67-74