# 一类并发实时系统的自动验证*

赵建华[1] 郑国梁[1] Dan Van Hung[2]

[1](南京大学计算机科学与技术系 南京 210093)

[2](联合国大学国际软件研究所 澳门)

E-mail: zhenggl@nju.edn.cn

**摘要** 一个被广泛用于验证实时系统的方法是根据被验证的实时性质,使用适当的双向模拟等价关系使无限的状态空间转化为有限的状态等价类空间.算法只需要在这个有限的等价类空间里搜索就可以得到正确答案.但是,这个等价类空间的规模一般随着系统规模的增大而产生爆炸性的增长,以至于在很多情况下,穷尽搜索这个空间是不现实的.该文引入了一个等价关系来验证一个由多个实时自动机通过共享变量组成的并发系统是否满足一个线性时段特性.同时,还引入了格局之间的兼容关系来避免对状态等价类空间的穷尽搜索.基于这两个关系,文章提出了一个算法来验证是否一个实时自动机网满足一个线性时段特性.实例研究显示,此算法在某些情况下比其他一些工具有更好的时间和空间效率.

**关键词** 模型验证,实时系统.

**中图法分类号** TP311

在过去的几年里,人们研究开发了一些用于实时系统的自动验证工具[1~3].这些工具中使用的核心算法大多数是由 Alur 和 Dill[4]在他们的开创性工作中提出的时间自动机的可达性分析算法.人们还提出了一系列的方法,试图解决状态空间爆炸问题,其中主要包括用来表达等价类空间的高效数据结构[5]和避免穷尽搜索状态空间的算法.

在本文中,我们使用线性时段特性(一种典型的时段演算(duration calculus)公式)作为系统规范,并使用实时自动机网作为系统的模型.因为线性时段特性是通过对系统在各个状态上的时间积分来定义的,所以相对于可达性分析来说,线性时段特性更加难以验证,并且在进行可达性分析时使用的等人价关系不能直接用于验证线性时段特性.周巢尘等人使用线性规划的方法解决了验证是否为一个实时自动机满足一个线性时段特性的问题[6].他们所提的技术被李宣东等人扩展到可以验证时间自动机的一个子集[7].

本文将解决验证一个由实时自动机通过共享变量组成的并发系统(实时自动机网)相对于一个线性时段特性的正确性问题.我们首先指出,一个实时自动机网满足一个线性时段特性当且仅当它的所有整数行为满足这个特性.随后,我们定义了在实时自动机网的格局集合上的等价关系和兼容关系,并基于这两个关系提出了一个验证算法.对于 Fischer 互斥协议的实例研究表明,这个方法在一些情况下可以很大程度地减低空间和时间上的开销.

本文第 1 节正式定义了带共享变量的实时自动机网.第 2 节表述线性时段特性和它被实时自动机网满足的条件.第 3 节给出我们的基本思想和算法.第 4 节描述了使用我们的算法对 Fischer 互斥协议进行验证时得到的一些统计结果以及和其他工具的比较.最后一节是对本文的总结和讨论.

* 本文研究得到国家自然科学基金(No. 69703009)和国家 863 高科技项目基金(No. 863--306-ZT06-04-4)资助.作者**赵建华**,1971 年生,博士,主要研究领域为软件工程,形式化方法.**郑国梁**,1937 年生,教授,博士生导师,主要研究领域为软件工程.**Dan Van Hung**,1950 年生,研究员,主要研究领域为实时系统设计,形式化方法.

本文通讯联系人:郑国梁,南京 210093,南京大学计算机科学与技术系

本文 1998-11-10 收到原稿,1999-03-01 收到修改稿

## 1 带共享变量的实时自动机网

在这一节中,我们将给出带共享变量的实时自动机网的正式定义.在本文中,我们使用 $Nat$ 来表示非负整数集合,而用 $Intv$ 表示所有在 $Nat$ 上的区间的集合.首先,我们来描述实时自动机[6]的定义.

**定义 1.** 一个实时自动机是一个二元组$\langle A,\Gamma\rangle$,其中:(1) $A=\langle S,s_0,E\rangle$是一个有限自动机;$S$ 是状态集合,$s_0$ 是初始状态,$E$ 是转换的集合.(2) $\Gamma$ 是从 $E$ 到 $Intv$ 的一个映射;$\Gamma$ 给 $E$ 中的每个转换 $e$ 赋予一个区间$[a_e,b_e]$或$[a_e,\infty)$,这里,$a_e,b_e$ 是整数,并且 $0\leqslant a_e\leqslant b_e$. $a_e$ 表示 $e$ 的最小延时,而 $b_e$ 或$\infty$表示 $e$ 的最大延时.

对一个从 $s_1$ 到 $s_2$ 的转换 $e$,我们用 $\check{e}$ 表示源状态 $s_1$,用 $\hat{e}$ 表示目标状态 $s_2$.对于一个 $S$ 中的状态 $s$,我们使用 $U_s$ 表示 $\max\{b_e|\check{e}=s\}$.我们用$[a_e,b_e]$(当 $b_e=\infty$时,使用$[a_e,\infty)$)来表示区间 $\Gamma(e)$.

**定义 2.** 一个带共享变量的实时自动机网 $N$(简写为实时自动机网)是一个四元组$\langle P,V,G,R\rangle$,其中:(1) $P=\{A_1,A_2,\ldots,A_n\}$是一个实时自动机的有限集合;对所有的 $i$,令 $A_i=\langle A_i,\Gamma_i\rangle$并且 $A_i=\langle S_i,q_0^i,E_i\rangle$.(2) $V=\{v_1,v_2,\ldots,v_m\}$是一个共享变量的集合.(3) $G$ 是一个从$(E_1\cup E_2\cup\ldots\cup E_n)$到 Varguard(正式定义见下文)的一个映射.(4) $R$ 是一个从$(E_1\cup E_2\cup\ldots\cup E_n)$到$(V\to Nat)$的一个映射.对每一个转换 $e$,$R(e)$是一个从共享变量集合 $V$ 到 $Nat$ 的部分映射.当 $e$ 发生时,每一个在 $R(e)$定义域中的共享变量 $v$ 的值都被重置为 $R(e)(v)$,而其余的共享变量的值保持不变.

Varguard 中的公式是关于共享变量的布尔表达式,其语法如下:$\varphi=True|v=c|\varphi_1\wedge\varphi_2$,其中 $v$ 是一个共享变量,$c$ 是一个常整数.Varguard 中的公式 $\varphi$ 相对于 $V$ 的一个取值组合 $\bar{v}$ 的值为真被记作 $\varphi(\bar{v})$.在本文的其余部分,对一个向量 $x$,我们用 $x_i$ 来表示 $x$ 的第 $i$ 个元素,并用 $x[x'_i/i]$来表示通过将 $x$ 中第 $i$ 个元素替换成 $x'_i$ 而得到的向量.

**定义 3.** 设 $N=\langle P,V,G,R\rangle$是一个实时自动机网.$N$ 的一个无时间状态 $\bar{s}$ 是一个 $n$ 维向量,其中 $s_i$ 是第 $i$ 个实时自动机 $A_i$ 的一个状态.$N$ 的一个格局是一个三元组$\langle\bar{s},\bar{t},\bar{v}\rangle$,其中 $\bar{s}$ 是 $N$ 的一个无时间状态;$\bar{t}$ 是一个 $n$ 维非负实数向量,$t_i$ 表示从实时自动机 $A_i$ 的上一个转换发生的时刻到当前的时间距离;$\bar{v}$ 是共享变量的一个取值组合.

**定义 4.** 设 $N=\langle P,V,G,R\rangle$是一个实时自动机网.而$\langle\bar{s},\bar{t},\bar{v}\rangle$和$\langle\bar{s}',\bar{t}',\bar{v}'\rangle$是 $N$ 的两个格局.我们定义 $N$ 的转换系统如下:

(1) $\langle\bar{s},\bar{t},\bar{v}\rangle\xrightarrow{e}\langle\bar{s}',\bar{t}',\bar{v}'\rangle$(其中 $e$ 是 $A_i$ 的一个转换)当且仅当(a)$(s_i=\check{e})\wedge(\bar{s}'=\bar{s}[\hat{e}/i]$,并且 $t_i\in[a_e,b_e]\wedge\bar{t}'=\bar{t}[0/i]$,并且(b) $G(e)(\bar{v})$,且 $\bar{v}'$是将 $\bar{v}$ 中在 $R(e)$的定义域中的每个变量的值重置为 $R(e)(v)$后所得到的共享变量取值组合.

(2) $\langle\bar{s},\bar{t},\bar{v}\rangle\xrightarrow{e,d}\langle\bar{s}',\bar{t}',\bar{v}'\rangle(d\geqslant 0)$当且仅当 $\bar{s}=\bar{s}'\wedge\bar{v}=\bar{v}'$并且对所有的 $j$,$(1\leqslant j\leqslant n)$,$(t'_j=t_j+d)\wedge(t'_j\leqslant U_{s_j})$成立.

给定两个格局 $C_1$ 和 $C_2$,一个转换 $e$,一个实数 $d$,当且仅当存在一个格局 $C'$使得 $C_1\xrightarrow{e,d}C'$并且 $C'\xrightarrow{e}C_2'$成立,我们记作 $C'\xrightarrow{e,d}C_2$.在这个转换系统的基础上,我们可以定义实时自动机网的执行.实时自动机网 $N$ 的一个执行表示了该网从初始格局开始的演化过程.一个执行 $\alpha$ 可以表示成 $\alpha=C_0\xrightarrow{e_1,d_1}C_1\xrightarrow{e_2,d_2}\ldots\xrightarrow{e_m,d_m}C_m\xrightarrow{e,d_{m+1}}C_{m+1}$,其中 $C_0$ 是 $N$ 的初始格局.我们把带时间长度的转换序列$(e_i,d_i)\wedge(e_{i+1},d_{i+1})\wedge\ldots\wedge(e_j,d_j)(1\leqslant i<j\leqslant m)$,称为从 $C_{i-1}$开始的一个 $N$ 的行为.一个格局$\langle\bar{s},\bar{t},\bar{v}\rangle$被称为整数格局当且仅当 $\bar{t}$ 的所有元素都是整数.$\alpha$ 被称为整数执行当且仅当所有的 $d_i,1\leqslant i\leqslant m+1$ 都是整数,此时,所有的 $C_i$ 都是整数格局.

## 2 线性时段特性

从现在开始,我们使用 $N$ 来表示在上面定义 2 中定义的实时自动机网.线性时段特性是用来表示实时系统的性质的时段演算公式.一个线性时段特性是形如 $\Psi\leqslant M$ 的关于状态时段积分的线性不等式,其中 $\Psi=\sum_{i=1}^{k}c_i\int S_i$,对每个 $i(1\leqslant i\leqslant k)$,$S_i$ 是一个在 $N$ 的无时间状态集合上的谓词,$c_i$ 和 $M$ 是实常数($c_i$ 被称为对应于 $S_i$

的系数).给定 $N$ 的一个执行 $\alpha=C_0 \xrightarrow{e_1,d_1} C_1 \xrightarrow{e_2,d_2} \cdots \xrightarrow{e_m,d_m} C_m \xrightarrow{e,d_{m+1}} C_{m+1}$,$S_i$ 在 $\alpha$ 上的时段积分 $\int S_i$ 定义成 $\int S_i=\sum_{u\in\beta_i} d_{u+1}$,其中 $\beta_i=\{u\mid(0\leqslant u\leqslant m)\wedge(\bar{s}_u\Rightarrow S_i)\}$,$\bar{s}_u$ 是 $C_u$ 的无时间状态.这样,给定线性时段特性 $\Psi\leqslant M$,$\Psi=\sum_{i=1}^{k} c_i\int S_i$ 在 $\alpha$ 上的值可以计算如下,

$$\Psi(\alpha)=\sum_{i=1}^{k} c_i\int S_i(\alpha)=\sum_{i=1}^{k} c_i\Big(\sum_{u\in\beta_i} d_{u+1}\Big)=\sum_{j=0}^{m}\bar{C}_{\bar{s}_j} d_{j+1},$$

其中 $\bar{s}_j$ 是 $C_j$ 的无时间状态,$\bar{C}_{\bar{s}_j}=\sum_{i\in\{i\mid\bar{s}_j\Rightarrow S_i\}} c_i$.

**定义 5.** 一个实时自动机网 $N$ 满足 $\Psi\leqslant M$ 当且仅当 $N$ 的所有执行 $\alpha$ 都满足 $\Psi(\alpha)\leqslant M$.

例如,在如图 1 所示的实时自动机网就具有这样的性质:第 1 个自动机在状态 $A$ 上的总时间总是比它在状态 $B$ 上的总时间长.这个性质可以用线性时段特性表示成 $\int at_B-\int at_A\leqslant 0$,其中 $at_A$ 和 $at_B$ 是两个谓词,并满足 $\langle A,C\rangle\Rightarrow at_A$,$\langle A,D\rangle\Rightarrow at_A$,$\langle B,C\rangle\Rightarrow at_B$,$\langle B,D\rangle\Rightarrow at_B$.

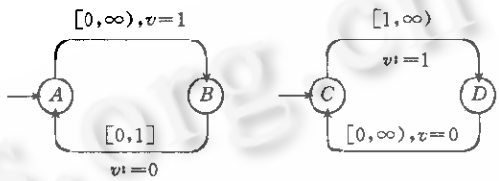


Fig. 1 A real-time network with shared variables ($v$ is a shared variable)

图1 一个带共享变量的实时自动机网络($v$是一个共享变量)

## 3 算 法

### 3.1 关于实时自动机网以及线性时段特性的一些性质

在本节中,设 $\Psi\leqslant M$ 是一个线性时段特性.下面的引理使我们能够把问题局限在整数域内考虑,从而简化了问题.

**引理 1.** 对于 $N$ 的任何执行 $\alpha$,必然存在一个 $N$ 的整数执行 $\alpha'$,使得 $\Psi(\alpha)\leqslant\Psi(\alpha')$.

由这个引理可知,实时自动机网 $N$ 如果不满足 $\Psi\leqslant M$,则必然存在一个 $N$ 的整数执行 $\alpha$ 不满足 $\Psi\leqslant M$.所以,我们验证 $N$ 是否满足 $\Psi\leqslant M$,只需要验证是否 $N$ 的所有整数执行都满足 $\Psi\leqslant M$.我们引入了如下的整数格局间的兼容关系和等价关系来将无穷的整数格局划分成有限多个等价类以及避免对等价类空间的穷尽搜索.

**定义 6.** 设 $C_1=\langle\bar{s},\vec{t},\vec{v}\rangle$ 和 $C_2=\langle\bar{s},\vec{t}',\vec{v}\rangle$ 是 $N$ 的两个整数格局.$C_1$ 兼容 $C_2$,记作 $C_1\leqslant C_2$,当且仅当对所有的 $i$,$1\leqslant i\leqslant n$,下面的公式成立.

$$(t_i=t'_i)\vee(t_i>RgBnd_{s_i}\wedge t'_i>RgBnd_{s_i})\vee(LowBnd_{s_i}<t_i<t'_i)\vee(t_i>t'_i\wedge\forall\, e\cdot(\hat{e}=s_i\Rightarrow b_e=\infty)),$$

其中 $RgBnd_{s_i}=\max(\{a_e\mid\hat{e}=s_i\}\cup\{b_e+1\mid(\hat{e}=s)\}\wedge(b_e<\infty))$,$LowBnd_{s_i}=\max(\{a_e=\hat{e}=s_i\})$.

兼容关系 $\leq$ 具有自反性和传递性.

**引理 2.** 设 $C_1$,$C_2$ 是 $N$ 的两个整数格局,并且满足 $C_1\leqslant C_2$.那么,对任何转换 $e$(可以是 $\varepsilon$),任何整数 $d$,如果存在一个格局 $C'_2$ 使得 $C_2\xrightarrow{e,d}C'_2$ 成立,那么必然存在一个格局 $C'_1$ 使得 $C_1\xrightarrow{e,d}C'_1$ 并且 $C'_1\leqslant C'_2$ 成立.

**定义 7.** 整数格局 $C_1$ 和 $C_2$ 等价,记作 $C_1\equiv C_2$,当且仅当 $(C_1\leqslant C_2)\wedge(C_2\leqslant C_1)$.

从 $\equiv$ 和 $\leq$ 的定义可以看出,$\equiv$ 是一个等价关系,并且将 $N$ 的整数格局空间划分为有限多个等价类.

**引理 3.** $R_N=\{[C]\mid(C$ 是 $N$ 的一个可达整数格局$)\}$ 是一个有限集合,其中 $[C]$ 表示包含 $C$ 的 $\equiv$ 的等价类.

### 3.2 加权有向图

一个实时自动机网 $N$ 的对应于线性时段特性 $\Psi\leqslant M$ 的加权有向图定义如下:图的节点集合是 $R_N$.从一个节点 $n$ 到另一个节点 $n'$ 有一个权为 $D_a$ 的边 $a$ 当且仅当存在一个转换 $e$ 和两个格局 $C\in n$,$C'\in n'$,使得对某个整数 $d$ 满足 $C\xrightarrow{e,d}C'$,并且 $a$ 的权 $D_a$ 等于 $\max\{\bar{C}_{\bar{s}}d\mid\exists\, C_1,C_2,e\cdot(C_1\in n\wedge C_2\in n'\wedge C_1\xrightarrow{e,d}C_2)\}$,其中 $\bar{s}$ 是 $C$ 的无时间状态.由这个定义可以看出,如果 $D_a<\infty$,则必然存在一个 $d$ 使得 $\bar{C}_{\bar{s}}d=D_a$ 并且 $C_1\xrightarrow{e,d}C_2$.加权有向图的初始节点

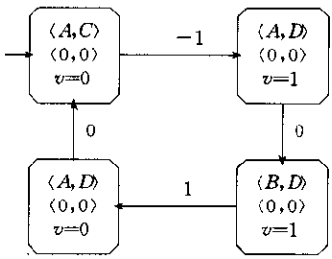


Fig. 2　A weighted directed graph
图2　加权有向图

就是包含自动机网的初始格局的节点.图2显示了图1中的实时自动机网对应于线性时段特性$\int at_B-\int at_A\leqslant 0$的加权有向图.对于一个格局$C=\langle \bar{s},\bar{t},\bar{v}\rangle$,我们用$L_C$表示$\min\{U_{s_i}-t_i\mid 1\leqslant i\leqslant n\}$.这个值表示了$N$在当前的无时间状态上所能等待的最长时间.设$p=[C_0]\xrightarrow{a_1}[C_1]\xrightarrow{a_2}\cdots\xrightarrow{a_m}[C_m]$,我们用$\hat{\Psi}(p)$来表示$p$的长度,即$\hat{\Psi}(p)=\sum_{i=1}^{m}D_{a_i}$.从加权有向图的定义可以得出下面的引理.

**引理4.** 设$p=[C_0]\xrightarrow{a_1}[C_1]\xrightarrow{a_2}\cdots\xrightarrow{a_m}[C_m]$是加权有向图的一条路径,并设$\alpha=C'_0\xrightarrow{e_1,d_1}C'_1\xrightarrow{e_2,d_2}\cdots\xrightarrow{e_m,d_m}C'_m$是满足$C'_i\in[C_i](1\leqslant i\leqslant n)$的执行.下面两个断言成立($\bar{S}$是$C_m$的无时间状态).

(1) $\Psi(\alpha)\leqslant\hat{\Psi}(p)+\max(0,\bar{C}_{\bar{S}}L_{C_m})$.

(2) 如果$\hat{\Psi}(p)+\max(0,\bar{C}_{\bar{S}}L_{C_m})=\infty$,那么对于任何实常数$M$都存在一个执行$\alpha'$满足$\Psi(\alpha')>M$.否则必然存在一个执行$\alpha''$满足$\Psi(\alpha'')=\hat{\Psi}(p)+\max(0,\bar{C}_{\bar{S}}L_{C_m})$.

### 3.3　算　法

从引理4可以得出,为了验证是否实时自动机网$N$满足一个线性时段特性$\Psi\leqslant M$,我们只需要验证是否相应的加权有向图的所有路径的长度都小于$M$.我们可以使用深度优先算法来生成有向图的路径并检验它们.由我们对兼容关系$\preceq$的讨论结果可知,在进行深度优先搜索时,如果算法发现下列两项之一成立时就可以进行回溯(设$[C]$是当前正被检验的路径的最后一个节点).

(1) $[C']$的所有后继节点都被生成并验证过.

(2) 存在一个被验证过的路径$p'$,它的最后一个节点是$[C]$,满足$C'\prec C$并且$p'$的长度大于当前路径的长度.

下面以伪代码的方式给出用来验证实时自动机网$N$是否满足$\Psi\leqslant M$的算法.$C^0$表示实时自动机的初始节点.$G$和$p$是算法中的两个辅助变量.在这个算法中,*NextNode*是一个用于枚举$p$的最后一个节点的所有后继节点的子程序.如果不能再枚举出新的后继节点,*NextNode*返回False,否则*NextNode*生成一个新的后继节点并将它加到$p$的最后,返回True.这个算法生成的路径最多只包含一个环路,而这一类路径的数目是有限的,所以这个算法会终止.

**算法.**

```
G:={⟨[C⁰],0⟩}; p:=⟨⟨[C⁰]⟩⟩
while True do
    flg:=NextNode; {给 p 扩展后继节点}
    while flg=False do
begin
      删除 p 的最后节点; {回溯}
      if (p 是空阶列) return True; {已经不用再生成新的路径来检查了}
      flg:=NextNode;
end
C:=p 的最后节点中的任意格局;
if  Ψ̂(p)+max(0,C̄_s L_C)>M), 其中 s̄ 是 C 的无时间状态
    then return False;
if  存在一个 p 的前缀 p' 使得 C 兼容 p 的最后节点中的格局, 并且 Ψ̂(p)>Ψ̂(p')
    then return False;
if ∃ ⟨[C'],l'⟩∈G·((C'⪯C)∧(l'≥Ψ̂(p)))
```

then 删除 $p$ 的最后节点;{回溯}
else
把元组$\langle[C],\Psi(p)\rangle$加入到 $G$ 中;
删除 $G$ 中所有满足 $l'\leqslant\Psi(p)\wedge C\preceq C'$的元组$\langle[C'],l'\rangle$;
end{of else}
end{of while}

## 4 Fischer 互斥协议:实例研究

我们已经实现了前面提出的算法,并将其用于验证 Fischer 互斥协议.由于人们可以通过增加系统中的进程个数来指数级地扩大 Fischer 互斥协议系统的规模,这个协议被多个模型验证算法用作检验效率的标准.这个协议可以保证一个并发系统中的多个进程的互斥.这个并发系统可以通过带有一个共享变量 $v$ 的实时自动机网来模拟.第 $i$ 个进程有它自己的标志编号 $i$,并且可以用如图3所示的实时自动机来模拟.

我们需要验证的是,在任何时刻,最多只有一个进程在它的临界区内.这个性质可以用线性时段特性描述成$\{Error\leqslant 0$,其中 $Error$ 是无时间状态集合上的谓词.对于一个无时间状态 $\bar{s}$,$\bar{s}\Rightarrow Error$ 当且仅当存在两个不同的整数 $i,j$,使得 $s_i=CS_i\wedge s_j=CS_j$.

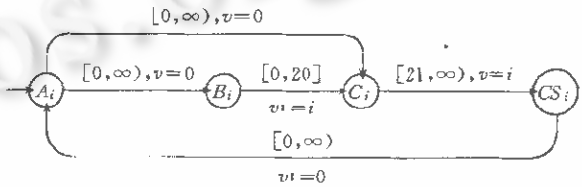


Fig. 3 A real-time automation modeling the $i$'th process
图3 模拟第$i$个进程的实时自动机

我们的程序验证了分别由2,3,...,11个进程组成 Fischer 互斥协议系统的互斥特性.我们的算法验证这些不同大小的系统时的空间需求可以参见表1.从表1中可以看出,当进程个数小于11时,每增加一个进程大约会增加3到4倍的空间需求.当被验证的系统包含11个进程时,算法所需要的时间大约是15分钟.作为对比,一个著名的模型验证工具 UPPAAL(一个基于可达性分析的工具)只能通过状态空间搜索来验证由8个进程组成的系统.表2中的统计数字[5]显示,当使用 UPPAAL 时,被验证系统的进程每增加一个,其空间需求会增加12倍以上.另一个工具 HyTech 的结果也与 UPPAAL 相类似.从这个结果可以看出,虽然线性时段特性要比可达性分析更加复杂,在某些情况下,我们的算法还是具有更好的空间和时间效率.

**Table 1** The space requirement of our algorithm for checking Fischer's protocol
**表1** 我们的算法验证 Fischer 互斥协议时的空间需求

| Number of the processes① | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|
| Number of the Nodes② | 18 | 65 | 20 | 727 | 2 378 | 7 737 | 25 080 | 81 035 | 260 998 | 837 949 |

①进程个数,②节点个数.

**Table 2** The space requirement of UPPAAL for checking Fischer's protocol
**表2** UPPAAL 在验证 Fischer 协议时的空间需求

| Number of the processes① | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| Demands of memory② | 16 | 237 | 3 528 | 59 715 |

①进程个数,②内存需求.

## 5 结 论

在本文中,我们提出了一个算法来验证一个实时自动机网是否满足一个线性时段特性,这种特性是关于状态时段积分的线性不等式.据我们所知,这是文献中第1个通过状态空间搜索来验证一个实时系统是否满足一个线性时段特性的算法.为了解决由时间约束引起的状态空间爆炸问题,我们引入了格局之间的兼容关系 $\preceq$.我们实现了这个算法并将它应用于一些实例,我们发现,兼容关系的引入在一些情况下可以有效降低算法的空间和

时间复杂性.虽然我们的算法要处理的问题比可达性分析更加复杂(线性时段特性可以用来表达状态的可达性,反之则不能),在一些例子中,我们的算法甚至比一些基于可达性分析的工具具有更好的效率.在实时系统的验证算法中引入兼容关系来降低算法的复杂性是一种新的尝试.我们将把这个方法运用到其他更加复杂的模型中去.同时,我们可将这个技术和现存的技术结合起来以获得一个更好的算法.

## 参考文献


1 Larsen K G, Paul Pettersson, Wang Yi. UPPAAL: status and developments. In: Orna Grumberg ed. Proceedings of the 9th International Conference on Computer-Aided Verification. LNCS 1254, Berlin: Springer-Verlag, 1997. 456～459

2 Henzinger T A, Ho P-H, H Wong-Toi. A users guide to HyTech. Technical Report, Departments of Computer Science, Cornell University, 1995

3 Daws C, Olivero A, Tripakis S, Yovine S. The tool Kronos. In: Rajeev Alur, Henzinger T A, Sontag E D eds. Hybrid Systems III, Verification and Control. LNCS 1066, Berlin: Springer-Verlag, 1996

4 Alur R, Dill D. Automata for modelling real-time systems. In: Paterson M S ed. Proceedings of the Automata Languages and Programming. LNCS 443, Berlin: Springer-Verlag, 1990

5 Larsen K G, Fredrik Larsson, Paul Pettersson *et al*. Efficient verification of real-time systems: compact data structure and state-space reduction. In: Padya P ed. Proceedings of the 18th IEEE Real-Time Systems Symposium. IEEE Computer Society, 1997

6 Zhou Chao-chen, Zhang Jing-zhong, Yang Lu *et al*. Linear duration invariants. Technical Report 11, Macau: UNU/IIST, 1993.

7 Li Xuan-dong, Hung Dan Van. Checking linear duration invariants by linear programming. Technical Report 70, Macau: UNU/IIST, 1996. Published in Joxan Jaffar, Roland H C yap eds. Advances in Computing Science, LNCS 1179, Berlin: Springer-Verlag, 1996. 321～332


# Automatic Verification of a Class of Concurrent Real-Time Systems


ZHAO Jian-hua[1] ZHENG Guo-liang[1] Dan Van Hung[2]

[1](*Department of Computer Science and Technology Nanjing University Nanjing* 210093)

[2](*International Institute for Software Technology The United Nations University Macau*)



**Abstract** A widely used method for checking real-time systems is, according to the real-time property to be checked, to use a proper bi-simulation equivalence relation to convert the infinite-timed state space to a finite equivalence class space. The algorithm needs only to explore the finite space to get a correct answer. In most cases, exhaustive exploration is very difficult because the equivalence class space increases explosively when the scale of the system increases. In this paper, an equivalence relation is introduced to check whether a concurrent system, which is composed of a finite set of real-time automata, satisfies a linear duration property. To avoid exhaustive exploration, this paper also introduces a compatibility relation between timed states (configurations). Based on these two relations, an algorithm is proposed to check whether a real-time automaton network satisfies a linear duration property. The cases study shows that under some conditions this algorithm has better efficiency than the tools in the literature.



**Key words** Model-checking, real-time system.