3.2 构造线性整数算术公式

令 $k\geqslant 0$,s 是一个 n 维向量,φ是一个 EG 逻辑公式.构造线性整数算术公式的算法 $Trans(\varphi,s,k)$,如图 2 所示.

算法 1 接收一个 EG 逻辑公式φ、一个 n 维向量 s 以及一个非负整数 k 作为输入,而输出是一个线性整数算术公式.该递归算法根据 EG 逻辑公式φ的结构进行递归.若φ为原子公式,则生成原子约束.若φ的最外层算子是命题逻辑算子,则根据否定和合取的语义进行构造.若φ形如 $E\langle a\rangle\varphi_1$ 或 $EG\varphi_1$,则通过迁移约束和路径约束结合限界语义生成相应的线性整数算术公式.注意,算法返回的线性整数算术公式是一个闭合公式(closed formula).

算法 1. $Trans(\varphi,s,k)$.
输入:EG 逻辑公式φ,n 维向量 $s=(s_1,\ldots,s_n)^T$,非负整数 k.
输出:线性整数算术公式ψ.

1 **begin**
2 **case** $\varphi=A\cdot M\geqslant B$ **do**
3 $\psi:=\wedge_{i=1}^{m}\left(\sum_{j=1}^{n}a_{ij}\cdot s_j\geqslant b_i\right)$
4 **case** $\varphi=\neg\varphi_1$ **do**
5 $\psi:=\neg Trans(\varphi_1,s,k)$
6 **case** $\varphi=\varphi_1\wedge\varphi_2$ **do**
7 $\psi:=Trans(\varphi_1,s,k)\wedge Trans(\varphi_2,s,k)$
8 **case** $\varphi=E\langle a\rangle\varphi_1$ **do**
9 $\psi:=k\geqslant 1\wedge\exists t_1\ldots\exists t_n.(T(s,(t_1,\ldots,t_n),a)\wedge Trans(\varphi_1,(t_1,\ldots,t_n),k))$
10 **case** $\varphi=EG\varphi_1$ **do**
11 $\theta:=Path((u(0)_1,\ldots,u(0)_n),\ldots,(u(k)_1,\ldots,u(k)_n))\wedge\wedge_{i=1}^{n}u(0)_i=s_i\wedge\wedge_{j=0}^{k}Trans(\varphi_1,(u(j)_1,\ldots,u(j)_n),k)$
12 $\psi:=\exists u(0)_1\ldots\exists u(0)_n\ldots\exists u(k)_1\ldots\exists u(k)_n.\theta$
13 **retrun** ψ
14 **end**

Fig.2 Algorithm for generating corresponding LIA formulas
图 2 生成对应的线性整数算术公式的算法

下面我们证明:如果输入的向量 s 中所有分量都满足大于 0 的条件,且同时算法生成的线性整数算术公式ψ成立,则ψ中任意变量 x 都满足非负条件,即 $x\geqslant 0$.为了证明这个结论,我们首先证明以下引理.

引理 1. 给定 $r\in\Delta$,n 维向量 $s=(s_1,\ldots,s_n)^T$ 和 $t=(t_1,\ldots,t_n)^T$,如果 $\wedge_{i=1}^{n}s_i\geqslant 0$,$s(^\bullet r)\geqslant 1$ 和 $T^-(s,t,r)$同时成立,则 $\wedge_{i=1}^{n}t_i\geqslant 0$ 成立.

证明:因为 $T^-(s,t,r)$成立,故对于任意 $1\leqslant i\leqslant n$,由映射 P^-的定义,可得 $t_i=s_i+P^-(^\bullet r,r^\bullet)_i=s_i+c^-(^\bullet r,r^\bullet,X_i)$.注意:对于任意 $r\in\Delta$,$r^\bullet(X_i)\geqslant 0$.我们分以下两种情况讨论.

- 如果 $X_i\neq{}^\bullet r$,则 $t_i=s_i+r^\bullet(X_i)\geqslant s_i\geqslant 0$;
- 如果 $X_i={}^\bullet r$,则 $t_i=s_i+r^\bullet(X_i)-1=r^\bullet(X_i)+s(X_i)-1=r^\bullet(X_i)+s(^\bullet r)-1\geqslant r^\bullet(X_i)\geqslant 0$.

综上,$\wedge_{i=1}^{n}t_i\geqslant 0$ 成立. □

借助引理 1,我们可以证明以下定理.

定理 2. 给定 EG 逻辑公式φ、n 维向量 $s=(s_1,\ldots,s_n)^T$ 和非负整数 k.令$\pi=E\langle a\rangle\varphi$或 $EG\varphi$.若 $\wedge_{i=1}^{n}s_i\geqslant 0\wedge Trans(\pi,s,k)$成立,则对于任意在构造线性整数公式 $Trans(\pi,s,k)$的过程中新增的变量 x 都满足非负条件,即 $x\geqslant 0$.

证明:我们分别对$\pi=E\langle a\rangle\varphi$和$\pi=EG\varphi$两种情况讨论.

- 假设$\pi=E\langle a\rangle\varphi$.在 $Trans(E\langle a\rangle\varphi,s,k)$中,新增的变量为 $t_1,\ldots,t_n$,因为 $T(s,(t_1,\ldots,t_n),a)$成立,由引理 1 可知:对于任意 $1\leqslant i\leqslant n$,$t_i\geqslant 0$ 为真;
- 假设$\pi=EG\varphi_1$.在 $Trans(EG\varphi,s,k)$中,新增的变量为 $u(0)_1,\ldots,u(0)_n,\ldots,u(k)_1,\ldots,u(k)_n$.我们对步长 k 进行归纳证明.当 $k=0$ 时,由于 $\wedge_{i=1}^{n}u(0)_i=s_i$ 和 $\wedge_{i=1}^{n}s_i\geqslant 0$ 成立,因此变量 $u(0)_1,\ldots,u(0)_n$ 非负.对于 $1\leqslant j\leqslant k$,由于存在 $r\in\Delta$使得 $u(j-1)(^\bullet r)\geqslant 1\wedge T^-(u(j-1),u(j),r)$为真,且由归纳假设,变量 $u(j-1)_1,\ldots,u(j-1)_n$ 非负,因此由引理 1 可知,变量 $u(j)_1,\ldots,u(j)_n$ 非负.综上,变量 $u(0)_1,\ldots,u(0)_n,\ldots,u(k)_1,\ldots,u(k)_n$ 都为非负.

根据算法 1,在构造线性整数算术公式 $Trans(\varphi,s,k)$时,只有遇到 $E\langle a\rangle$算子和 EG 算子时才会增加新的变量.因此由定理 2 我们可知:只要输入的 n 维向量 s 是一个合法的 BPP 状态,即向量中的每个分量非负,且同时算法 1 生成的线性整数算术公式 $Trans(\varphi,s,k)$成立,则基于算法 1 的编码不会出现到达不合法的 BPP 状态的情况. □

3.3 算法的正确性

下面我们对 EG 逻辑公式进行结构归纳证明算法 1 的正确性.

定理 3(算法的正确性). 给定 EG 逻辑公式φ,n 维向量 $s=(s_1,\ldots,s_n)^T$ 和非负整数 k,则算法 1 是正确的,即下列命题成立.

- 终止性:算法会终止.
- 可靠性:如果 $s\models_k\varphi$成立,则 $Trans(\varphi,s,k)$可满足.
- 完备性:如果 $Trans(\varphi,s,k)$可满足,则 $s\models_k\varphi$成立.

证明:由于算法 1 是根据 EG 逻辑公式φ的结构进行递归构造线性整数算术公式的,因此可知算法 1 是终止的.对于可靠性和完备性,我们如下对φ进行结构归纳证明.

- 假设φ是原子命题公式,即$\varphi=A\cdot M\geqslant B$,则根据原子约束 $\wedge_{i=1}^{m}\left(\sum_{j=1}^{n}a_{ij}\cdot s_j\geqslant b_i\right)$以及 $s\models_k A\cdot M\geqslant B$ 当且仅当 $A\cdot s\geqslant B$,显然可得可靠性和完备性成立.
- 假设$\varphi=\neg\varphi_1$.对于可靠性,假设 $Trans(\neg\varphi_1,s,k)$不可满足,则根据算法 1,$\neg Trans(\varphi_1,s,k)$不可满足,因此 $Trans(\varphi_1,s,k)$正确.由归纳假设,$s\models_k\varphi_1$ 成立,故 $s\models_k\neg\varphi_1$ 不成立.反之,对于完备性,假设 $Trans(\neg\varphi_1,s,k)$正确,则根据算法 1,$Trans(\varphi_1,s,k)$不可满足,由归纳假设,$s\models_k\varphi_1$ 不成立,故 $s\models_k\neg\varphi_1$ 成立.
- 假设$\varphi=\varphi_1\wedge\varphi_2$.对于可靠性,假设 $Trans(\varphi_1\wedge\varphi_2,s,k)$不可满足,则根据算法 1 的构造,$Trans(\varphi_1,s,k)\wedge Trans(\varphi_2,s,k)$不可满足,因此 $Trans(\varphi_1,s,k)$不可满足或 $Trans(\varphi_2,s,k)$不可满足.由归纳假设,$s\models_k\varphi_1$ 或 $s\models_k\varphi_2$ 不成立,因此 $s\models_k\varphi_1\wedge\varphi_2$ 不成立.反之,对于完备性,假设 $Trans(\varphi_1\wedge\varphi_2,s,k)$正确,则根据算法 1,$\neg Trans(\varphi_1,s,k)$不可满足且$\neg Trans(\varphi_2,s,k)$不可满足.由归纳假设,$s\models_k\neg\varphi_1$ 且 $s\models_k\neg\varphi_2$ 不成立,因此 $s\models_k\varphi_1\wedge\varphi_2$ 成立.
- 假设$\varphi=E\langle a\rangle\varphi_1$.对于可靠性,假设 $Trans(E\langle a\rangle\varphi_1,s,k)$可满足,则 $k=0$ 或者对于任意 $t_1,\ldots,t_n$,如果 $T(s,(t_1,\ldots,t_n),a)$,则$\neg Trans(E\langle a\rangle\varphi_1,(t_1,\ldots,t_n),k)$.若 $k=0$,则显然 $s\models_k A\langle a\rangle\neg\varphi_1$ 成立.若后者,假设 $T(s,(t_1,\ldots,t_n),a)$正确,则$\neg Trans(\varphi_1,(t_1,\ldots,t_n),k)$正确,因此 $Trans(\varphi_1,(t_1,\ldots,t_n),k)$不可满足.由归纳假设,$(t_1,\ldots,t_n)\models_k\varphi_1$ 不成立,因此 $s\models_k A\langle a\rangle\neg\varphi_1$ 成立.综上,$s\models_k E\langle a\rangle\varphi_1$ 不成立.反之,对于完备性,假设 $Trans(E\langle a\rangle\varphi_1,s,k)$正确,则 $k\geqslant 1$ 且存在 $t_1,\ldots,t_n$,使得 $T(s,(t_1,\ldots,t_n),a)\wedge Trans(\varphi_1,(t_1,\ldots,t_n),k)$,因此$\neg Trans(E\langle a\rangle\varphi_1,(t_1,\ldots,t_n),k)$不可满足.由归纳假设,$(t_1,\ldots,t_n)\models_k\neg\varphi_1$ 不成立,因此$(t_1,\ldots,t_n)\models_k\varphi_1$ 成立,所以 $s\models_k E\langle a\rangle\varphi_1$ 成立.
- 假设$\varphi=EG\varphi_1$.对于可靠性,假设 $Trans(EG\varphi_1,s,k)$不可满足,则对于任意 $u(0)_1,\ldots,u(0)_n,\ldots,u(k)_1,\ldots,u(k)_n$,如果 $\alpha=path((u(0)_1,\ldots,u(0)_n),\ldots,(u(k)_1,\ldots,u(k)_n))\wedge\wedge_{i=1}^{n}u(0)_i=s_i$,则 $\neg\wedge_{j=0}^{k}Trans(\varphi_1,(u(j)_1,\ldots,u(j)_n),k)$.假设α正确,则存在 $0\leqslant j\leqslant k$,使得$\neg Trans(\varphi_1,u(j),k)$正确,因此 $Trans(\varphi_1,u(j),k)$不可满足.由归纳假设,$u(j)\models_k\varphi_1$ 不成立,因此 $s\models_k AF\neg\varphi_1$ 成立,即 $s\models_k EG\varphi_1$ 不成立.反之,对于完备性,假设 $Trans(EG\varphi_1,s,k)$正确,则存在 $u(0)_1,\ldots,u(0)_n,\ldots,u(k)_1,\ldots,u(k)_n$,使得 $\alpha\wedge\wedge_{j=0}^{k}Trans(\varphi_1,(u(j)_1,\ldots,u(j)_n),k)$.因此对于任意 $0\leqslant j\leqslant k$,$\neg Trans(\varphi_1,u(j),k)$不可满足.由归纳假设,$u(j)\models_k\neg\varphi_1$ 不成立,即 $u(j)\models_k\varphi_1$ 成立,所以 $s\models_k EG\varphi_1$ 成立. □

3.4 线性整数算术公式构造示例

为了更加直观地认识上述算法,下面我们举例说明如何构建一个线性整数算术公式.

我们考虑 BPP(V,Δ),其中,$V=\{X_1,X_2,X_3\}$,Δ包含以下规则.

$$r_1: X_1\xrightarrow{a}X_2X_3,$$
$$r_2: X_2\xrightarrow{a}X_1X_2,$$
$$r_3: X_3\xrightarrow{b}X_1.$$

我们令初始状态 $s=X_1$,令 $k=2$,则相应的标签迁移系统如图 3 所示.

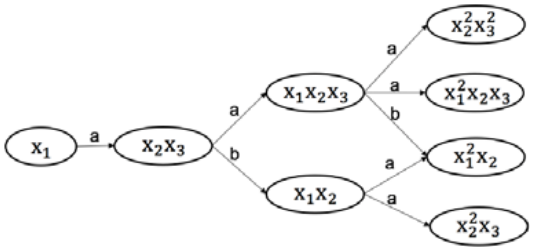

Fig.3 Corresponding labelled transition system

图 3 相应的标签迁移系统

待检测的 EG 逻辑公式 φ 为 $EG(E\langle a\rangle(X_2+X_3\geqslant 2))$,则 $Trans(\varphi,s,2)$ 为

$$\exists u(0)_1\exists u(0)_2\exists u(0)_3\exists u(1)_1\exists u(1)_2\exists u(1)_3\exists u(2)_1\exists u(2)_2\exists u(2)_3.\theta,$$

其中,

$$\theta := Path((u(0)_1,u(0)_2,u(0)_3),(u(1)_1,u(1)_2,u(1)_3),(u(2)_1,u(2)_2,u(2)_3))\wedge(u(0)_1=1\wedge u(0)_2=0\wedge u(0)_3=0)\wedge$$
$$\wedge_{j=0}^{2} Trans(E\langle a\rangle(X_2+X_3\geqslant 2),(u(j)_1,u(j)_2,u(j)_3),2).$$

我们首先展开 θ 的第 1 个合取支即路径约束.根据映射 P^- 的定义,可知:

$$P^-(^\bullet r_1,r_1^\bullet)=(-1,1,1)^T,P^-(^\bullet r_2,r_2^\bullet)=(1,0,0)^T,P^-(^\bullet r_3,r_3^\bullet)=(1,0,-1)^T.$$

因此,路径约束中的每个合取支(j=1,2)为

$$\psi_{trans}(j):=[u(j-1)_1\geqslant 1\wedge(u(j-1)_1-1=u(j)_1\wedge u(j-1)_2+1=u(j)_2\wedge u(j-1)_3+1=u(j)_3)]\vee$$
$$[u(j-1)_2\geqslant 1\wedge(u(j-1)_1+1=u(j)_1\wedge u(j-1)_2+0=u(j)_2\wedge u(j-1)_3+0=u(j)_3)]\vee$$
$$[u(j-1)_3\geqslant 1\wedge(u(j-1)_1+1=u(j)_1\wedge u(j-1)_2+0=u(j)_2\wedge u(j-1)_3-1=u(j)_3)].$$

所以路径约束为 $\psi_{trans}(1)\wedge\psi_{trans}(2)$.

接着,我们展开 θ 的第 3 个合取支.对于 j=0,1,2,考虑 $Trans(E\langle a\rangle(X_2+X_3\geqslant 2),(u(j)_1,u(j)_2,u(j)_3),2)$,该约束对应着子公式 $E\langle a\rangle(X_2+X_3\geqslant 2)$ 在 $(u(j)_1,u(j)_2,u(j)_3)$ 上 2 步内成立的情形,展开为

$$\psi_{sub}(j):=2\geqslant 1\wedge\exists t_1\exists t_2\exists t_3.(([a=a\wedge u(j)_1\geqslant 1\wedge(u(j)_1-1=t_1\wedge u(j)_2+1=t_2\wedge u(j)_3+1=t_3)]\vee$$
$$[a=a\wedge u(j)_2\geqslant 1\wedge(u(j)_1+1=t_1\wedge u(j)_2+0=t_2\wedge u(j)_3+0=t_3)]\vee$$
$$[b=a\wedge u(j)_3\geqslant 1\wedge(u(j)_1+1=t_1\wedge u(j)_2+0=t_2\wedge u(j)_3-1=t_3)])\wedge t_2+t_3\geqslant 2).$$

因此,第 3 个合取支为 $\psi_{sub}(0)\wedge\psi_{sub}(1)\wedge\psi_{sub}(2)$.注意:对于上述公式中的迁移动作,我们在实际中可以用整数进行编码.

综上,根据算法生成的公式 $Trans(\varphi,s,2)$ 为

$$\psi:=\exists u(0)_1\exists u(0)_2\exists u(0)_3\exists u(1)_1\exists u(1)_2\exists u(1)_3\exists u(2)_1\exists u(2)_2\exists u(2)_3$$
$$(\psi_{trans}(1)\wedge\psi_{trans}(2)\wedge(u(0)_1=1\wedge u(0)_2=0\wedge u(0)_3=0)\wedge\psi_{sub}(0)\wedge\psi_{sub}(1)\wedge\psi_{sub}(2)).$$

4 工具实现及实验结果

4.1 技术架构

基于线性整数算术公式刻画,我们采用基于约束的方法实现了工具,其架构如图 4 所示.

工具接受 3 个输入,分别是 BPP 模型(V,Δ)、待检测的 EG 逻辑公式 φ 以及步长 k.工具从这些输入获取 BPP 中符号集 V 和规则集 Δ 的信息,并根据 EG 逻辑公式 φ 的结构以及步长 k 的大小,生成原子约束、迁移约束和路径约束等约束条件,进而构造出相应的线性整数算术公式,保存为约束文件作为 SMT 求解器 Z3 的输入.如果 Z3 的求解结果为 sat,则说明在限界语义下 EG 逻辑公式 φ 在 k 步内满足;若求解的返回结果为 unsat,则说明在 k 步内不满足.

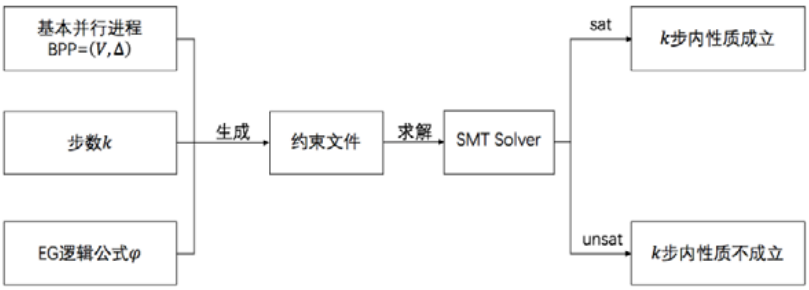

Fig.4 Architecture of the tool

图 4 工具架构

4.2 实验结果

我们在实验部分首先讨论第 3.4 节中的 BPP 的例子(如图 3 所示),该 BPP 包含 3 个进程符号以及 3 条规则.我们对以下 4 个 EG 逻辑公式在限制步长 k 分别为 1,5,10,20,50 和 100 的条件下进行限界模型检测.

$$\varphi_1 = X_1 + X_2 \geqslant 1 \wedge X_3 \geqslant 0,$$
$$\varphi_2 = EG(E\langle a\rangle(X_2 + X_3 \geqslant 2)),$$
$$\varphi_3 = EG(X_1 + X_2 \geqslant 2 \rightarrow E\langle a\rangle(X_1 \geqslant 2 \wedge X_3 \geqslant 1)),$$
$$\varphi_4 = EG(AF(X_1 + X_2 \geqslant 2)).$$

实验结果见表 2,实验结果的单位为 s.

Table 2 Experimental results

表 2 实验结果

	1	5	10	20	50	100
φ_1	0.01	0.01	0.01	0.01	0.01	0.01
φ_2	0.02	0.03	0.05	0.08	0.18	0.34
φ_3	0.03	0.05	0.07	0.12	0.27	0.54
φ_4	0.05	0.09	0.27	0.98	6.22	52.81

从表 2 可以看出,对于φ_1,步长 k 不影响求解时间.这是因为φ_1是两个原子命题公式的合取式,根据算法 1,对于原子命题公式,作为参数的 k 不参与线性整数算术公式的构造,因此求解φ_1的时间总体趋于一致.而对于φ_2,φ_3,φ_4,求解时间总体上随着步长 k 的增大而增加.公式φ_3的嵌套深度比φ_2的和φ_4的均要小,然而求解φ_3的时间只比求解φ_2的稍多,而求解φ_4的时间相对于求解φ_2,φ_3的时间较多,这在步长 k 较大,如 k=50,100 时尤为明显.这是因为限制求解速度的原因在于变量的数量.对于 EG 算子,根据算法 1,生成的线性整数算术公式需要引入$(k+1)\cdot n$个变量,其中,n 为 BPP 中进程符号的数量.而φ_4实际上等价于公式 $EG(\neg EG\neg(X_1+X_2\geqslant 2))$,因此对于$\varphi_4$,生成的线性整数算术公式中含有的变量数目为$(k+1)^2\cdot n^2$,因此在 k 较大时,求解器在较长时间后返回结果.

因为已有的程序验证基准案例并不能证明其全部可以 BPP 可解,所以这些案例并不能直接使用.同时,现有的与 Petri 网相关的验证工具都是针对传统 Petri 网,它们所使用的标准测试集也不再适用.因此,我们随机生成了测试用例,对不同规模的 BPP 在多个步长下进行实验.BPP 的规模主要体现在其进程符号的数量以及迁移规则的数目.我们实验的对象是进程符号数分别是 20,50 和 100 的 BPP.对于具有相同进程符号数的 BPP,我们将迁移规则数分别设置为 10,20 和 30 进行实验.对于 EG 逻辑公式,我们选取了上述的φ_2,φ_3,φ_4.表 3~表 5 分别对应 k 分别为 15,20,25 的实验结果,单位为 s.

Table 3 Experimental results (k=15)

表 3 实验结果(k=15)

	φ_2		φ_3		φ_4	
20	10	0.692	10	0.841	10	9.029
	20	1.346	20	1.631	20	12.689
	30	1.927	30	2.128	30	17.362
50	10	1.597	10	2.146	10	20.954
	20	3.102	20	3.588	20	29.190
	30	4.718	30	4.957	30	51.181
100	10	3.450	10	4.285	10	36.718
	20	6.092	20	6.848	20	64.620
	30	8.826	30	10.503	30	99.130

Table 4 Experimental results (k=20)

表 4 实验结果(k=20)

	φ_2		φ_3		φ_4	
20	10	0.915	10	1.137	10	11.429
	20	1.810	20	1.917	20	22.122
	30	2.664	30	2.978	30	33.313
50	10	2.135	10	2.641	10	23.680
	20	4.124	20	4.645	20	50.214
	30	6.476	30	6.803	30	91.353
100	10	4.183	10	5.572	10	59.651
	20	9.047	20	9.963	20	100.159
	30	12.443	30	13.297	30	171.353

Table 5 Experimental results (k=25)

表 5 实验结果(k=25)

	φ_2		φ_3		φ_4	
20	10	1.158	10	1.545	10	21.584
	20	2.285	20	2.645	20	30.469
	30	3.668	30	4.126	30	44.819
50	10	2.854	10	3.419	10	42.042
	20	5.429	20	5.750	20	88.374
	30	8.528	30	8.657	30	111.447
100	10	5.206	10	6.252	10	88.267
	20	12.821	20	11.074	20	181.874
	30	17.927	30	18.893	30	226.909

从实验结果可以看出,求解时间随着进程符号数量和迁移规则数目的增加以及 k 的增大而增加.进程符号数量影响了线性整数算术公式中引入的量词和变量的数目.此外,对于一个 BPP 状态 s,因为我们要保证经过一次迁移后得到的状态 t,s 和 t 中每一个进程符号的数量关系要与相应的规则一致,因此 BPP 进程符号数的增多会导致对应约束数量的增加,从而影响求解时间.在迁移约束和路径约束中,由于要在 BPP 规则集中寻找使得迁移和路径成立的规则,因此规则集的大小影响了迁移约束和路径约束中析取式的公式大小.而步长 k 决定了最外层算子为 EG 的公式对应的线性整数算术公式的路径约束中合取式的大小,即子约束的数目,同时,k 也对引入的变量数目产生影响.

5 相关工作

在基于 BPP 和 Petri 网的模型检测问题上,对于线性时间逻辑,Petri 网上的线性时间 mu-演算的模型检测问题是可判定的[20].然而,对于一些表达力比较弱的逻辑,如原子命题为 $EN(a)$且只包含 Fφ作为模态算子的线性时间逻辑,在 VBPP 上的模型检测却是不可判定的[12].对于分支时间逻辑,除了上文提及的不可判定性结果,对于 EF 逻辑,即在 Hennessy-Milner Logic 的基础上增加 EF 算子的逻辑,Esparza 还在文献[12]中证明了以下结论:

(1) 基于 Petri 网的模型检测问题是不可判定的,然而对于 BPP,即使给定的 BPP 是有穷状态的,该问题是 PSPACE 难的;(2) 在 BPP 上,对于长度固定的公式(fixed formula),模型检测问题是 $\sum_d^p$ 难的,其中,d 是模态算子的嵌套深度.在这个结论的基础上,Mayr 证明了若给定的模型为 BPP,则 EF 逻辑的模型检测问题实际上分别是 PSPACE 完备和 $\sum_d^p$ 完备的[21].EGF 是 CTL*的一个算子,其表达的是:“存在一条路径,使得某个特定的性质被满足无穷次”.在正则模型检测(regular model checking)中,EGF 代表了活性和反复可达性(recurrent reachability).在 EF 逻辑的基础上加上 EGF 算子,我们可以得到 EGF 逻辑.文献[22]证明了 BPP 上对 EGF 逻辑的模型检测问题是可判定的.而 Fu 证明了该问题的复杂度是 PSPACE 完备且对于长度固定的公式是 $\sum_d^p$ 完备的[13],也就是说,增加 EGF 算子并没有提高模型检测的复杂度.证明的关键思路是使用一种称为半片段(semisegment)的结构,而该结构也被用于处理基于 Petri 网模型检测的状态爆炸问题[23].

在安全性问题上,Petri 网的可达性问题是可判定的[24],且最早被证明是 EXPSPACE 难的[25],然而该问题的复杂度仍然是学术界的一个开问题.最近,文献[26]证明了该问题的一个新的 Ackermannian 上界,而文献[9]证明了该问题的一个新的 Tower 难下界.Petri 网的可覆盖性问题是 EXPSPACE 完备的[27,28],对于 Petri 网的可覆盖性,目前检测工具主要分为两类:一类基于完备算法,对状态空间进行搜索,判断可达的状态集是否覆盖了指定状态,这类工具包括 BFC[1],IIC[29]和 MIST;而另一类工具如 Petrinizer[30]将给定的 Petri 网的结构和状态以及可覆盖性条件转换为约束,然后交给求解器进行求解.然而,Petrinizer 在理论上是不完备的,其无法判定不安全的测试用例.以上这些工具由于 Petri 网的可覆盖性问题复杂度过高,因此都无法处理规模较大的测试用例.而 BPP 的可达性以及可覆盖性问题是 NP 完备问题[11].针对 BPP,文献[31]实现了工具 CFPCV,在 Petri 网可覆盖性检测的安全性方法的基础上,额外增加了基于 BPP 的子网可标记方法,并以此来验证状态是否可达,保证了算法在理论上是完备的.

在限界模型检测上,模型检测问题一般被规约为 SAT 问题.该方法最先被运用到检测 LTL 性质上[32],接着在 ACTL 性质检测上也具有应用[33].但是这两类研究只关注时序逻辑的存在片段(existential fragment),其限界语义不能适用于如本文提及的活性 $AF\varphi$等表示全称(universal)性质的公式.文献[34]就此问题提出了 CTL 的限界语义,并将 CTL 公式编码为 QBF 公式.以上研究讨论的模型都是有穷状态系统,对于无穷状态系统,文献[35]提出了将性质分别用 LTL_{-X} 和 $ECTL_{-X}$ 表示,并在分布式时间 Petri 网上进行限界模型检测的方法,该方法仍是将模型检测问题编码为命题逻辑公式.不同于直接处理时序逻辑公式的方法,文献[36]首先将时序逻辑公式转换为对应的ω自动机,其接受条件为安全性或活性等性质条件,然后将ω自动机编码为 Presburger 算术,而该方法适用于无穷状态系统.

6 总结及未来的工作

本文关注基本并行进程上 EG 逻辑的限界模型检测问题,并将该问题转化为线性整数算术公式的可满足性问题.首先,我们根据基于基本并行进程的限定步长为 k 的 EG 逻辑限界语义,以及基本并行进程的模型结构和表示特定性质的 EG 逻辑公式,定义相应的约束,并给出了对应的基于线性整数算术公式的刻画,然后将线性整数算术公式作为 SMT 求解器的输入进行求解.

我们未来的工作主要包括两个方面:一是对并发程序的一些性质如动态更新(dynamic updating)进行分析,使用基本并行进程对并发程序进行建模并验证这些性质;二是尝试寻找描述并发程序的其他模型,并比较和分析其在安全性和活性的验证上与基本并行进程的异同.

References:

[1] Kaiser A, Kroening D, Wahl T. Dynamic cutoff detection in parameterizedconcurrent programs. In: Proc. of the 22nd Int'l Conf. on Computer Aided Verification, Vol.6174. Berlin: Springer-Verlag, 2010. 645–659.

[2] D'Osualdo E, Kochems J, Ong CHL. Automatic verification of Erlang-styleconcurrency. In: Proc. of the 20th Int'l Symp. on Static Analysis, Vol.7935. Berlin: Springer-Verlag, 2013. 454–476.

[3] Kaiser A, Kroening D, Wahl T. Efficient coverability analysis by proof minimization. In: Proc. of the 23rd Int'l Conf. on Concurrency Theory, Vol.7454. Berlin: Springer-Verlag, 2012. 500−515.

[4] Bouajjani A, Emmi M. Bounded phase analysis of message-passing programs. In: Proc. of the 18th Int'l Conf. on Tools and Algorithms for the Construction and Analysis of Systems, Vol.7214. Berlin: Springer-Verlag, 2012. 451−465.

[5] Ganty P, Majumdar R. Algorithmic verification of asynchronous programs. ACM Trans. on Programming Languages and Systems, 2012,34(1):Article No.6.

[6] Petri CA. Communication with automata [Ph.D. Thesis]. Technische Universitat Darmstadt, 1962.

[7] Vardi MY. Verification of concurrent programs: The automata-theoretic framework. Annals of Pure and Applied Logic, 1991,51: 79−98.

[8] Ganty P, Majumdar R, Rybalchenko A. Verifying liveness for asynchronous programs. In: Proc. of the 36th Symp. on Principles of Programming Languages. New York: ACM, 2009. 102−113.

[9] Czerwinski W, Lasota S, Lazic R, Leroux J, Mazowiecki F. The reachability problem for Petri nets is not elementary. In: Proc. of the 51st Annual Symp. on Theory of Computing. New York: ACM, 2019. 24−33.

[10] Christensen S. Decidability and decomposition in process algebras [Ph.D. Thesis]. Edinburgh: The University of Edinburgh, 1993.

[11] Esparza J. Petri nets, commutative context-free grammars, and basic parallel processes. Fundamenta Informaticae, 1997,31:13−25.

[12] Esparza J. Decidability of model checking for infinite-state concurrent systems. Acta Informatica, 1997,34(2):85−107.

[13] Fu H. Model checking EGF on basic parallel processes. In: Proc. of the 9th Int'l Symp. on Automated Technology for Verification and Analysis. Berlin: Springer-Verlag, 2011. 120−134.

[14] Nieuwenhuis R, Oliveras A, Tinelli C. Solving SAT and SAT modulo theories: From an abstract Davis-Putnam-Logemann-Loveland procedure to DPLL (T). Journal of the ACM, 2006,53(6):937−977.

[15] Dutertre B. Yices 2.2. In: Proc. of the Int'l Conf. on Computer Aided Verification, Vol.8559. Berlin: Springer-Verlag, 2014. 737−744.

[16] De Moura L, Bjorner N. Z3: An efficient SMT solver. In: Proc. of the 14th Int'l Conf. on Tools and Algorithms for the Construction and Analysis of Systems. Berlin: Springer-Verlag, 2008. 337−340.

[17] Barrett C, Conway CL, Deters M, Hadarean L, Jovanović D, King T, Reynolds A, Tinelli C. CVC4. In: Proc. of the 23rd Int'l Conf. on Computer Aided Verification. Berlin: Springer-Verlag, 2011. 171−177.

[18] Cimatti A, Griggio A, Schaafsma BJ, Sebastiani R. The MathSAT5 SMT solver. In: Proc. of the 19th Int'l Conf. on Tools and Algorithms for the Construction and Analysis of Systems. Berlin: Springer-Verlag, 2013. 93−107.

[19] Minsky M. Computation: Finite and Infinite Machines. Prentice Hall, 1967.

[20] Esparza J. On the decidability of model checking for several mu-calculi and Petri nets. In: Proc. of the Trees in Algebra and Programming (CAAP'94). Berlin: Springer-Verlag, 1994. 115−129.

[21] Mayr R. Weak bisimulation and model checking for basic parallel processes. In: Proc. of the 16th Conf. on Foundations of Software Technology and Theoretical Computer Science, Vol.1180. Berlin: Springer-Verlag, 1996. 88−99.

[22] To AW, Libkin L. Recurrent reachability analysis in regular model checking. In: Proc. of the 15th Int'l Conf. on Logic for Programming, Artificial Intelligence, and Reasoning, Vol.5330. Berlin: Springer-Verlag, 2008. 198−213.

[23] Strehl K, Thiele L. Interval diagram techniques for symbolic model checking of Petri nets. In: Proc. of the 1999 Design, Automation and Test in Europe. Berlin: Springer-Verlag, 1999. 756−757.

[24] Mayr E. An algorithm for the general Petri net reachability problem. In: Proc. of the 13th Annual Symp. on Theory of Computing. New York: ACM, 1981. 238−246.

[25] Lipton R. The reachability problem requires exponential-space hard. Technical Report, 62, Department of Computer Science, Yale University, 1976.

[26] Leroux J, Schmitz S. Reachability in vector addition systems is primitive-recursive in fixed dimension. In: Proc. of the 34th Annual Symp. on Logic in Computer Science. IEEE, 2019. 1−13.

[27] Karp RM, Miller RE. Parallel program schemata. Journal of Computer and System Sciences, 1969,3(2):147−195.

[28] Rackoff C. The covering and boundedness problems for vector addition systems. Theoretical Computer Science, 1978,6(2): 223−231.

[29] Kloos J, Majumdar R, Niksic F, *et al*. Incremental, inductive coverability. In: Proc. of the Int'l Conf. on Computer Aided Verification. Berlin: Springer-Verlag, 2013. 158−173.

[30] Esparza J, Ledesma-Garza R, Majumdar R, *et al*. An SMT-based approach to coverability analysis. In: Proc. of the Int'l Conf. on Computer Aided Verification. Berlin: Springer-Verlag, 2014. 603−619.

[31] Ding RJ, Li GQ. Efficient implementation of coverability verification on communication-free Petri net. Ruan JianXue Bao/Journal of Software, 2019,30(7):1939−1952 (in Chinese with English abstract). http.//www.jos.org.cn/1000-9825/5750.htm [doi: 10.13328/j.cnki.jos.005750]

[32] Biere A, Cimatti A, Clarke EM, Zhu YS. Symbolic model checking without BDDs. In: Proc. of the Tools and Algorithms for Construction and Analysis of Systems, Vol.1579. Berlin: Springer-Verlag, 1999. 193−207.

[33] Penczek W, Wozna B, Zbrzezny A. Bounded model checking for the universal fragment of CTL. Fundamenta Informaticae, 2002, 51(1-2):135−156.

[34] Zhang WH. Bounded semantics. Theoretical Computer Science, 2015,564:1−29.

[35] Meski A, Pólrola A, Penczek W, Wozna-Szczesniak B, Zbrzezny A. Bounded model checking approaches for verification of distributed time Petri nets. In: Proc. of the Int'l Workshop on Petri Nets and Software Engineering, Vol.723. CEUR-WS.org, 2011. 72−91.

[36] Schüle T, Schneider K. Bounded model checking of infinite state systems. Formal Methods in System Design, 2007,30(1):51−81.

附中文参考文献:

[31] 丁如江,李国强.非交互式 Petri 网可覆盖性验证的高效实现.软件学报,2019,30(7):1939−1952. http.//www.jos.org.cn/1000-9825/5750.htm [doi: 10.13328/j.cnki.jos.005750]

谭锦豪(1996－),男,硕士生,CCF 学生会员,主要研究领域为形式化验证,程序语言理论.

李国强(1979－),男,博士,副教授,CCF 高级会员,主要研究领域为形式化方法,程序语言理论,知识表示与推理.