ЛОГИКО-МАТЕМАТИЧЕСКИЕ ОСНОВЫ МОЗ
(Вычислительные системы)
1985 год Выпуск 107

УДК 519.47

НЕРАЗРЕШИМОСТЬ ПРОПОЗИЦИОНАЛЬНОЙ ВРЕМЕННОЙ ЛОГИКИ ДЛЯ СЕТЕЙ ПЕТРИ

В.Е.Котов, Л.А.Черкасова

Введение средств модальной логики при описании и верификации программ существенно облегчает процесс анализа программ. Динамическая логика, введенная Праттом [1], ее пропозициональный вариант, предложенный Фишером и Ладнером [2], основаны на описании поведения программы в терминах вход-выход. Это довольно сильное ограничение, поскольку такая логика не имеет возможности рассуждать о поведении программы в процессе выполнения, т.е. о поведении программы в некоторый промежуточный момент до завершения ее работы.

Было сделано немало попыток расширения динамической логики за счет введения новых логических связок, позволяющих выражать некоторые свойства путей исполнения программы, это - логика процессов Пратта [3], язык SOAPL [4], временная логика [5]. В частности, в терминах временной логики выразимы такие свойства поведения программы, как отсутствие тупиков, живость (liveness), взаимное исключение и др. Единственным ограничением временной логики является то, что ее синтаксис не представляет средств для именования программ, т.е. модель состоит из единственной фиксированной программы.

В данной работе сделана попытка применить временную логику для описания и изучения свойств сетей Петри. В терминах временной логики легко выразимы такие свойства сетей Петри, как безопасность, достижимость разметки, живость и др. Язык пропозициональной временной логики оказывается достаточно богатым и сильным, чтобы его средствами записать свойство включения (или эквивалентности) языков произвольных помеченных сетей Петри. Неразрешимость проблемы

включения и эквивалентности языков сетей Петри приводит к неразрешимости проблемы истинности формулы в рассматриваемой пропозициональной временной логике для сетей Петри.

I. Основные сетевые понятия и определения

С е т ь П е т р и – это набор $N = (P, T, F, M_0)$, где $P = (p_1, p_2, \ldots, p_n)$ – конечное непустое упорядоченное множество м е с т, T – конечное непустое множество п е р е х о д о в, $F \subseteq P \times T \cup T \times P$ – отношение и н ц и д е н т н о с т и, $M_0 : P \to \{0,1,2,\ldots\}$ – н а ч а л ь н а я р а з м е т к а.

Для произвольного элемента сети $x \in P \cup T$ обозначим через $^\bullet x$ множество его в х о д н ы х элементов $\{y \mid y\ F\ x\}$, а через $x^\bullet$ – множество в ы х о д н ы х элементов $\{y \mid x\ F\ y\}$. Графическим представлением сети служит двудольный ориентированный граф с двумя типами вершин; вершины-места изображаются кружочками, вершины-переходы – барьерами. Из вершины x в вершину y ведет дуга тогда и только тогда, когда $x\ F\ y$. В графическом представлении сети разметка места p изображается помещением в вершину-кружок числа $M_0(p)$ или соответствующего числа точек (фишек). Пример сети приведен на рис.I.

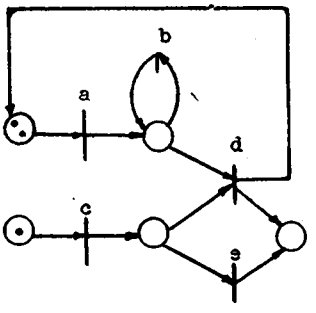

Рис. I

Функционирование сети Петри описывается формально с помощью множества последовательностей срабатываний и множества достижимых в сети разметок. Сеть начинает функционировать при начальной разметке M_0. Смена разметок происходит в результате срабатывания одного из переходов.

Введем следующую функцию инцидентности $F: P \times T \cup T \times P \to \{0,1,2,\ldots\}$, которая определяется как

$$F(x,y) = \begin{cases} 1, \text{ если } (x,y) \in F, \\ 0 - \text{в противном случае.} \end{cases}$$

Переход t может сработать при разметке M, если $\forall p \in P$: $M(p) - F(p,t) \geq 0$. В результате срабатывания перехода t при разметке M, последняя сменяется разметкой M' по следующему правилу: $\forall p \in P$: $M'(p) = M(p) - F(p,t) + F(t,p)$. В сети на рис.I на первом шаге могут сработать переход a или c. Будем говорить, что раз –

метка M' следует за разметкой M и обозначать этот факт: $M \xrightarrow{\tau} M'$. Разметка M' достижима в сети N от разметки M в результате последовательности срабатываний $\tau = t_1 t_2 \ldots t_k$, если существует последовательность следующих друг за другом разметок $M \xrightarrow{t_1} M_1 \xrightarrow{t_2} \ldots \to M'$ (обозначение: $M \xrightarrow{\tau} M'$). Разметка M достижима в сети N, если она достижима от M_0. Через $R_N(M_0)$ будем обозначать множество всех достижимых в N разметок. Обозначим через $S_N(M_0) = \{\tau \in T^* \mid \exists M' \in R_N(M_0): M_0 \xrightarrow{\tau} M'\}$ множество всех последовательностей срабатываний переходов в сети N.

Переход t достижим в сети N, если существует такая достижимая разметка M в N, при которой переход t может сработать. Переход t живой, если он достижим от любой разметки из $R_N(M_0)$. Сеть Петри живая, если все ее переходы живы.

Место p называется безопасным, если $\forall M \in R_N(M_0)$: $M(p) \le 1$; соответственно сеть безопасна, если все ее места безопасны.

Рассмотрим сеть Петри $N = (P, T, F, M_0)$ с помечающей функцией $A: T \to A$, где A – некоторый алфавит. Функция A обобщается на последовательности срабатываний из $S_N(M_0)$ естественным образом: $A(\tau t) = A(\tau)\,A(t)$. Функция $L(N) = \{A(\tau) \mid \tau \in S_N(M_0)\}$ называется префиксным языком сети N.

Разметка M в сети N называется тупиковой, если ни один из переходов сети при этой разметке не может сработать. Последовательность срабатываний переходов называется тупиковой, если срабатывание соответствующих переходов приводит к тупиковой разметке в сети. Обозначим через $T_N(M_0)$ множество всех тупиковых последовательностей срабатываний из $S_N(M_0)$, а через $L_0(N) = \{A(\tau) \mid \tau \in T_N(M_0)\}$ – тупиковый язык сети N. Обозначим через $T_N^{\omega}(M_0)$ множество всех бесконечных последовательностей срабатываний переходов, а через $L_{\omega}(N) = \{A(\tau) \mid \tau \in T_N^{\omega}(M_0)\}$ язык бесконечных слов. Определим $L_0^{\omega}(N) = L_0(N) \cup L_{\omega}(N)$. Обозначим через $\mathcal{L}$ класс всех L-языков, а через $\mathcal{L}_0^{\omega}$ – класс всех L_0^{ω}-языков, порожденных помеченными сетями Петри.

2. Вычислительная модель

На основе следующей абстрактной вычислительной модели, предложенной в [6], будет рассмотрена модель конкретной вычислительной системы, заданной при помощи сети Петри.

Абстрактная модель состоит из следующих элементов:

S – множество с о с т о я н и й, возможно бесконечное. Каждый элемент $s \in S$ представляет полную конфигурацию вычислительной системы;

θ – н а ч а л ь н ы й п р е д и к а т. Будем рассматривать вычисления, начинающиеся в состоянии s_0, для которого $\theta(s_0)$ выполнено;

T – конечное множество п е р е х о д о в. С каждым переходом $t \in T$ ассоциируем частичную функцию $f_t : S \to 2^S$, где $f_t(s)$ сопоставляет все возможные выходы перехода t из состояния s. Переход t на состоянии s в о з м о ж е н, если $f_t(s) \neq \emptyset$, в противном случае переход t н е в о з м о ж е н. Состояние $s \in S$, при котором любой переход t из T невозможен, называется з а к л ю ч и т е л ь н ы м.

Н а ч а л ь н ы м в ы ч и с л е н и е м такой системы является последовательность состояний

$$\sigma: s_0 \overset{t_1}{\to} s_1 \overset{t_2}{\to} s_2 \overset{t_3}{\to} \dots , \quad t_i \in T,$$

удовлетворяющая следующим требованиям:

а) последовательность σ максимальна, т.е. или бесконечна, или последнее состояние s_k является заключительным;

б) первое состояние последовательности σ удовлетворяет начальному предикату, т.е. $\theta(s_0)$ = истина;

в) для каждого шага $s_i \overset{t_i}{\to} s_{i+1}$ последовательности σ выполнено: $s_{i+1} \in f_{t_i}(s_i)$.

Суффикс начального вычисления называется д о п у с т и м ы м в ы ч и с л е н и е м.

Когда рассматривается конкретная вычислительная система, то вышеописанные элементы отождествляются с более конкретными объектами. Если вычислительная система задается сетью Петри $N=(P_N, T_N, F_N, M_{0N})$ с помечающей функцией $A: T_N \to A$, то множеством состояний S будет множество допустимых разметок $R_N(M_0)$, начальный предикат предписывает рассматривать работу сети из начальной разметки M_{0N}, под множеством переходов T модели подразумевается множество помеченных переходов $A(T_N)$ сети. Частичная функция f_t $(t \in A(T_N))$ по заданной разметке M_i сопоставляет подмножество новых разметок, каждая из которых получается в результате сраба-

тывания одного из переходов, помеченных в сети символом t. Если ни один из переходов, помеченных символом t, не может сработать при разметке M_i, то $f_t(M_i) = \beta$. Начальным вычислением является последовательность достижимых разметок, полученная при порождении бесконечного или тупикового слова в сети N.

3. Пропозициональная временная логика

Пропозициональная временная логика (PTL) является классической пропозициональной логикой, расширенной унарными временными операторами: o, $\Diamond$, $\Box$. В отличие от формул классической логики, истинность которых определяется на состоянии, временные операторы позволяют выражать некоторые свойства путей вычисления в целом. Интуитивно утверждение: $\Box\varphi$ истинно для вычисления, если φ истинно во всех будущих состояниях вычисления; $\Diamond\varphi$ истинно, если φ истинно в одном из будущих состояний; $o\varphi$ истинно, если φ истинно в следующем состоянии вычисления.

Формулы PTL строятся из следующих элементов:

а) множество Φ предикатных переменных для элементарных высказываний: $\varphi_1, \varphi_2, \varphi_3, \ldots$;

б) логические связки $\wedge$, $\neg$;

в) временные операторы: o (следующий), $\Box$ (всегда), $\Diamond$ (возможно).

Правила образования формул следующие:

а) предикатная переменная $\varphi \in \Phi$ является формулой PTL;

б) если φ_1 и φ_2 – формулы, то $\varphi_1 \wedge \varphi_2$, $\neg\varphi_1$, $o\varphi_1$, $\Box\varphi_1$, $\Diamond\varphi_1$ также формулы PTL.

В дальнейшем будем использовать $\vee$, $\supset$ и $\equiv$ как обычные сокращения для известных формул, построенных из $\wedge$ и $\neg$, и использовать скобки для исключения двусмысленностей.

Формулы, использующие только обычные логические связки ($\wedge$, $\neg$, $\vee$, $\supset$, $\equiv$), называются п р о с т ы м и в ы с к а з ы в а н и я м и.

Истинность простого высказывания φ, не содержащего временных операторов, зависит только от состояния s и конкретной интерпретации предикатных переменных. Обозначим через $s \models \varphi$ тот факт, что простое высказывание φ истинно на состоянии s.

В отличие от простых высказываний, которые являются функциями от состояния в последовательности вычисления, в р е м е н н ы е в ы с к а з ы в а н и я (т.е. те формулы, которые содер-

жат $\circ, \Box, \Diamond$) являются функциями от всей последовательности вычисления. Будем обозначать через $\sigma \models \varphi$ тот факт, что временное высказывание φ истинно на последовательности вычисления σ.

В дальнейшем будем использовать обозначение σ для произвольной бесконечной последовательности вычисления, построенной по начальному вычислению системы $\tilde{\sigma}$: $s_0 \xrightarrow{t_1} s_1 \xrightarrow{t_2} s_2 \xrightarrow{t_3} \dots$, следующим образом. Если начальное вычисление $\tilde{\sigma}$ бесконечно, то соответствующая последовательность вычисления σ имеет вид $\sigma: s_0, s_1, s_2, \dots$ Если начальное вычисление $\tilde{\sigma}$ конечно и его последнее состояние s_k является заключительным, то σ строится следующим образом $\sigma: s_0, s_1, s_2, \dots, s_k, s_k, \dots$, т.е. последнее заключительное состояние s_k повторяется бесконечное количество раз.

Состояние s_i характеризует i-й шаг вычисления (или, другими словами, вычисление в i-й момент времени). Мы будем говорить о моменте времени 0, как о настоящем, и о любом моменте времени, большем 0, как о будущем.

Простое высказывание φ может быть проинтерпретировано как временное высказывание, относящееся к настоящему, более точно $\sigma \models \varphi \Leftrightarrow s_0 \models \varphi$.

Временные высказывания относятся как к настоящему, так и к будущему. Обозначим через σ_i суффикс последовательности вычисления σ после выполнения первых i шагов: $s_i, s_{i+1}, \dots$ Тогда семантика временных операторов задается следующим образом:

$$\sigma \models \Box\varphi \Leftrightarrow \forall i \geq 0: \ \sigma_i \models \varphi;$$

$$\sigma \models \Diamond\varphi \Leftrightarrow \exists i \geq 0: \ \sigma_i \models \varphi.$$

Заметим, что $\Box$ и $\Diamond$-дуальные операторы, т.е. имеет место $\Diamond\varphi \equiv \neg\Box\neg\varphi$;

$$\sigma \models \circ\varphi \Leftrightarrow \sigma_1 \models \varphi.$$

Поскольку формулы PTL образуются из простых высказываний при помощи временных операторов, а также обычных логических связок, то определение $\sigma \models \varphi$ для произвольного временного высказывания может быть дополнено следующим образом:

$$\sigma \models \varphi_1 \wedge \varphi_2 \Leftrightarrow \sigma \models \varphi_1 \text{ и } \sigma \models \varphi_2;$$

$$\sigma \models \varphi_1 \vee \varphi_2 \Leftrightarrow \sigma \models \varphi_1 \text{ или } \sigma \models \varphi_2;$$

$$\sigma \models \neg\varphi \Leftrightarrow \text{неверно } \sigma \models \varphi.$$

Поскольку нас в большей степени интересуют свойства и утверждения о функционировании сети Петри в целом (а не отдельных ее последовательностей вычисления) определим семантику временных высказываний на сети. Обозначим через $\Sigma(N)$ множество всех последовательностей вычислений в сети N (будем отождествлять модель сети и сеть в тех случаях, когда это не приводит к возникновению двусмысленностей). Будем говорить, что временное утверждение φ истинно на сети N, если φ истинно на любой последовательности вычисления этой сети, т.е. $N \models \varphi \Leftrightarrow \forall \sigma \in \Sigma(N): \sigma \models \varphi$.

В дальнейшем PTL для сети Петри N будет обозначаться через PTL_N.

Формула φ называется выполнимой на модели N, если существует такая последовательность вычисления $\sigma \in \Sigma(N)$, что $\sigma \models \varphi$. Формула φ называется N-истинной, если выполняется $N \models \varphi$. Формула φ называется общезначимой, если для любой модели N имеет место $N \models \varphi$ (будем писать в этом случае $\models \varphi$).

Примерами общезначимых формул будут:

$$\Box \varphi = \neg \Diamond \neg \varphi ,$$

$$\Box(\varphi \wedge \psi) \equiv \Box\varphi \wedge \Box\psi .$$

Проблемой истинности (N-истинности) для PTL_N называется проблема определения по заданной формуле $\varphi \in PTL_N$, имеет место или нет $\models \varphi$ ($N \models \varphi$) ?

Проиллюстрируем, как в терминах временной логики выражаются самые известные свойства сетей Петри. Рассмотрим сеть $N=(P_N, T_N, F_N, M_{0N})$. Поскольку приведенные ниже свойства не связаны с помечающей функцией A, то будем ее в данном случае рассматривать как тождественную, тогда множество переходов T в модели совпадает с множеством переходов T_N в сети. Определим интерпретацию следующих предикатных переменных на состояниях:

$$\text{Terminal}(s) = \bigwedge_{t \in T} (f_t(s) = \emptyset).$$

Данная формула характеризует множество заключительных состояний (тупиковых разметок).

Формула

$$\text{Enabled}(T')(s) = \bigwedge_{t \in T'} (f_t(s) \neq \emptyset)$$

характеризует множество переходов T', которые могут сработать в сети при данном состоянии (разметке).

Оба предиката выражаются бескванторной формулой первого порядка.

Следующие свойства сетей Петри выражаются при помощи операторов временной логики соответствующими формулами:

переход t достижим в сети N ⇔ формула $\Diamond$ Enabled (t) выполнима на N;

переход t живой в сети N ⇔ $N \models \Box \Diamond$ Enabled (t);

сеть N жива ⇔ $N \models \bigwedge_{t \in T} (\Box \Diamond \text{Enabled } (t))$;

сеть N безопасна ⇔ $N \models \Box (\sum_{p_i \in P_N} M(p_i) = 1)$;

тупиковый язык $L_0(N)$ не пуст ⇔ формула $\circ \Diamond$ Terminal выполнима на N.

Зачастую для того, чтобы выразить более широкий спектр утверждений о возможном поведении сети, полезно проделать небольшие дополнительные построения (изначально), которые не влияют на работу сети. Например, каждый переход t снабжается дополнительным выходным местом p_t, выполняющим функцию счетчика ($^\bullet p_t = \{t\}$, $p_t^\bullet = \emptyset$). Естественным образом состояние-разметка дополняются кортежом мест p_{t_i}, $t_i \in T$. Тогда утверждение: переход t может сработать в сети ровно n раз записывается следующей выполнимой формулой:

$$\Diamond(M(p_t) = n) \wedge \neg \Diamond (M(p_t) = n+1).$$

4. Неразрешимость PTL_N

В данном параграфе будет установлено, что проблема включения (эквивалентности) языков $\mathcal{L}_0^\omega$ сетей Петри, которая неразрешима, сводится к проблеме N-истинности формулы PTL_N, что, в свою очередь, приводит к неразрешимости последней.

ЛЕММА I. Проблемы включения и эквивалентности для языков из $\mathcal{L}_0^\omega$ неразрешимы.

ДОКАЗАТЕЛЬСТВО. Возьмем произвольные сети N_1 и N_2. Нужно показать, имеет место или нет включение $L_0^\omega(N_1) \subseteq L_0^\omega(N_2)$? Заметим, что из справедливости включения $L_0^\omega(N_1) \subseteq L_0^\omega(N_2)$ непосредственно следует справедливость включения префиксных языков данных сетей $L(N_1) \subseteq L(N_2)$. Но проблема включения для языков из $\mathcal{L}$ неразреши-

ма [7], следовательно, неразрешима проблема включения языков и для $\mathcal{L}_0^\omega$.

Аналогичным образом показывается неразрешимость проблемы эквивалентности для $\mathcal{L}_0^\omega$. □

Рассмотрим произвольную сеть N и ее модель. Выберем произвольное начальное вычисление $\sigma \in \Sigma(N)$. Обозначим через $T(\sigma)$ последовательность помеченных переходов вдоль σ, т.е. если $\sigma: s_0 \xrightarrow{t_1} s_1 \xrightarrow{t_2} s_2 \xrightarrow{t_3} \dots$, то $T(\sigma)=t_1t_2t_3\dots$ Заметим, что множество $\{T(\sigma) \mid \sigma \in \Sigma(N)\}$ совпадает с определенным выше языком $L_0^\omega(N)$.

Чтобы для произвольных сетей $N_1=(P_1,T_1,F_1,M_{01})$ и $N_2=(P_2,T_2,F_2,M_{02})$ написать формулу PTL_N, которая истинна тогда и только тогда, когда $L_0^\omega(N_1) \subseteq L_0^\omega(N_2)$, нужно построить общую для N_1 и N_2 сеть-модель N, поведение которой определялось бы поведением сетей N_1 и N_2.

Сеть $N = (P,T,F,M_0)$ строится при помощи специального наложения сетей N_1 и N_2. При этом $P = P_1 \cup P_2$ и

$$M_0(p) = \begin{cases} M_{01}(p), & \text{если } p \in P_1, \\ M_{02}(p), & \text{если } p \in P_2. \end{cases}$$

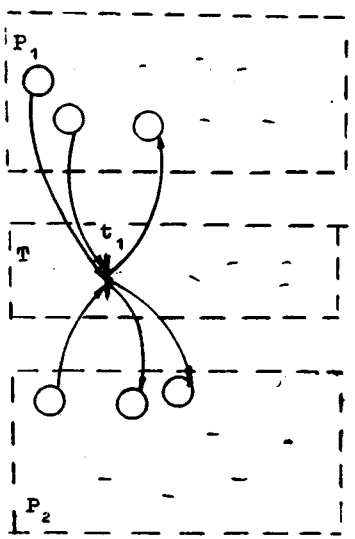

Рис.2

Множество переходов T в сети N строится следующим образом. Если в сетях N_1 и N_2 каждый переход имеет свое уникальное имя, то осуществляется простое наложение по этим переходам (рис.2), в противном случае наложение производится следующим образом. Пусть $t_1^1,\dots,t_n^1$ – переходы сети N_1, помеченные символом a, и $t_1^2,\dots,t_k^2$ – переходы сети N_2, помеченные тем же символом a. В сети n заводятся n x k переходов $t_{11}, t_{12},\dots,t_{1k},\dots,t_{n1},t_{n2},\dots,t_{nk}$, помеченных символом a. Переход t_{ij} получается от наложения перехода t_i^1 на t_j^2, т.е. $\cdot t_{ij} = \cdot(t_i^1) \cup \cdot(t_j^2)$ и $t_{ij}^\cdot = (t_i^1)^\cdot \cup (t_j^2)^\cdot$.

ЛЕММА. 2. Если $L_0^\omega(N_1) \subseteq L_0^\omega(N_2)$, то $L_0^\omega(N) = L_0^\omega(N_1)$.

ДОКАЗАТЕЛЬСТВО. Достаточно установить справедливость следующих двух включений: $L_0^\omega(N_1) \subseteq L_0^\omega(N)$ и $L_0^\omega(N) \subseteq L_0^\omega(N_1)$.

Возьмем $\tau \in L_0^\omega(N_1)$. Обозначим через τ^i префикс слова τ длины i. Покажем, что $\forall i \geq 1$, $\tau^i \in L(N)$. Это устанавливается индукцией по i.

Рассмотрим случай, когда $i = 1$. Префикс состоит из имени одного перехода, например, a. По условию в сетях N_1 и N_2 могут сработать переходы (например, t_i^1 и t_j^2 соответственно), помеченные символом a, но тогда по построению в сети N переход t_{ij}, помеченный символом a, также может сработать, следовательно, $\tau^1 = a \in L(N)$.

Предположим, что для $i = k$ утверждение доказано. Покажем для $i = k+1$. Пусть $\tau^{k+1} = \tau^k t$, причем по предположению $\tau^k \in L(N)$. Аналогично предыдущим рассуждениям, в сетях N_1 и N_2 могут сработать переходы (например, t_l^1 и t_m^2 соответственно), помеченные символом t, тогда по построению в сети N переход t_{lm}, помеченный символом t, также может сработать, следовательно, $\tau^{k+1} \in L(N)$.

Если слово τ - бесконечное, то $\tau \in L_\omega(N)$ и соответственно $\tau \in L_0^\omega(N)$.

Если слово τ - конечное (предположим, его длина равна n), то $\tau = \tau^n \in L(N)$. Поскольку по условию леммы $L_0^\omega(N_1) \subseteq L_0^\omega(N_2)$, то слово τ тупиковое как в сети N_1, так и в сети N_2, т.е. ни один переход как в сети N_1, так и в сети N_2 после порождения слова τ не может сработать. Но тогда, по построению, в сети N после порождения слова τ также ни один переход не может сработать, т.е. τ - тупиковое, следовательно, $\tau \in L_0^\omega(N)$.

Аналогичным образом показывается и второе включение: $L_0^\omega(N) \subseteq L_0^\omega(N_1)$. □

Рассмотрим модель сети N, построенной из сетей N_1 и N_2 вышеописанным способом. Обозначим через $f_t^i: S \to 2^S$ частичную функцию, сопоставляющую помеченному переходу t все возможные выходы из состояния $s \in S$ в сети $N_i (i = 1,2)$.

Определим следующие предикаты. Формула

$$\mathrm{Enabled}_i(t)(s) = (f_t^i(s) \neq \emptyset),\ i=1,2\ ,$$

характеризует возможность срабатывания переходов, помеченных символом t, на состоянии $s \in S$ в сети N_i $(i=1,2)$.

Формула

$$\mathrm{Terminal}_i(s) = \bigwedge_{t \in T_i} (f_t^i(s) = \emptyset), \quad i=1,2 ,$$

характеризует множество заключительных (тупиковых) состояний в сети N_i $(i=1,2)$.

ЛЕММА 3.

$$N \models \Box\Big(\bigwedge_{t \in T} (\mathrm{Enabled}_1(t) \supset \mathrm{Enabled}_2(t)) \wedge$$

$$\wedge\ (\mathrm{Terminal}_1 \supset \mathrm{Terminal}_2)\Big) \Rightarrow L_0^{\omega}(N_1) \subseteq L_0^{\omega}(N_2) .$$

ДОКАЗАТЕЛЬСТВО. Возьмем $\tau \in L_0^{\omega}(N_1)$, требуется установить, что $\tau \in L_0(N_2)$. Обозначим через τ^i префикс слова τ длины i. Покажем, что $\forall\, i \geq 1$, $\tau^i \in (N_2)$. Будем доказывать это индукцией по i.

Пусть $i = 1$. Префикс состоит из имени одного перехода, например, t. По условию первой половины формулы, поскольку помеченный переход t может сработать в сети N_1, то переход t может сработать и в сети N_2, т.е. $\tau^1 = t \in L(N_2)$.

Предположим, для $i = k$ утверждение доказано.

Покажем справедливость для $i = k+1$. Пусть $\tau^{k+1} = \tau^k t$, по предположению $\tau^k \in L(N_2)$. По условию первой половины формулы, поскольку помеченный переход t может сработать в сети N_1, то t может сработать и в сети N_2, т.е. $\tau^{k+1} = \tau^k t \in L(N_2)$. Причем если слово τ бесконечно, то не существует такого i, для которого τ^i являлось бы тупиковым словом в N_2. Предположим противное, пусть τ^k – тупиковое слово в N_2. Поскольку τ – бесконечное слово в N_1, то существует переход t, который срабатывает после τ^k в сети N_1, но тогда по условию первой половины формулы переход t может сработать и в сети N_2, что противоречит тупиковости слова τ^k. Следовательно, $\tau \in L_{\omega}(N_2)$ и $\tau \in L_0^{\omega}(N_2)$.

Если слово τ – конечное тупиковое слово в сети N_1 (предположим, его длина равна n), то $\tau = \tau^n \in L(N_2)$, кроме того, по условию второй половины формулы следует, что в сети N_2 после порождения слова τ достигнуто заключительное состояние (тупиковая разметка), т.е. τ – тупиковое слово, следовательно, $\tau \in L_0^{\omega}(N_2)$.

Покажем, что если $L_0^{\omega}(N_1) \subseteq L_0^{\omega}(N_2)$, то выполнена формула в условии леммы. Заметим, что по лемме 2 $L_0^{\omega}(N) = L_0^{\omega}(N_1) \subseteq L_0^{\omega}(N_2)$.

Возьмем произвольное начальное вычисление σ и покажем, что для любого суффикса σ_i выполнена данная формула. Обозначим через

τ последовательность переходов вдоль вычисления σ. Как было замечено, $\tau \in L_0^{\omega}(N)$. Обозначим через τ^i префикс слова τ длины i, который порождается после исполнения i шагов начального вычисления σ и перед началом допустимого вычисления σ_{i+1}.

Истинность формулы будем доказывать индукцией по i, где i-индекс допустимого вычисления.

При i=0 допустимое вычисление σ_0 совпадает с начальным вычислением σ, и истинность формулы проверяется на начальной разметке M_0. Если переход, помеченный символом t, может сработать в сети N_1, то существует бесконечное или тупиковое слово $w \in L_0^{\omega}(N_1)$, начинающееся с t, то тогда по условию леммы $w \in L_0^{\omega}(N_2)$, и, следовательно, переход, помеченный символом t, может сработать и в сети N_2. Таким образом, истинность первой половины формулы для i=0 доказана. Истинность второй половины формулы при i=0 тривиальна.

Предположим, для i=k утверждение доказано.

Пусть i = k+1. Понятно, что если $\sigma_{k+1} \models \mathrm{Terminal}_1$, то слово τ^k - тупиковое в сети N_1 и, следовательно, $\tau^k \in L_0^{\omega}(N_1)$. Поскольку $L_0^{\omega}(N_1) \subseteq L_0^{\omega}(N_2)$, то $\tau^k \in L_0^{\omega}(N_2)$. Таким образом, $\sigma_{k+1} \models \mathrm{Terminal}_2$, т.е. истинность второй половины формулы доказана.

Предположим, что $\sigma_{k+1} \models \mathrm{Enabled}_1(t)$. Покажем, что тогда $\sigma_{k+1} \models \mathrm{Enabled}_2(t)$. По предположению, слово $\tau^k t$ порождается сетью N_1. Понятно, что существует слово $w \in L_0^{\omega}(N_1)$, тупиковое или бесконечное, префиксом которого является $\tau^k t$. По условию леммы тогда $w \in L_0^{\omega}(N_2)$, и, следовательно, в сети N_2 после слова τ^k на k+1 шаге может сработать переход t, т.е. $\sigma_{k+1} = \mathrm{Enabled}_2(t)$. Доказательство леммы закончено. □

ТЕОРЕМА. Проблема N-истинности формулы PTL_N неразрешима.

ДОКАЗАТЕЛЬСТВО непосредственно следует из лемм I и 3.□

Заключение

В работе рассмотрена временная логика как средство описания и изучения свойств сетей Петри. Из литературы известны примеры полных систем аксиом [8] для более простых моделей, в которых оператор o (следующий) детерминирован, т.е. каждый следующий шаг

вычисления определен однозначно. Имеются примеры полных правил вывода [6] и для абстрактной модели, рассмотренной в данной работе, которые переносятся и на случай сетей Петри.

Дальнейший шаг использования временной логики в сетях Петри авторы видят в разработке правил вывода для подклассов регулярных сетей [9], т.е. из истинности временного утверждения, характеризующего свойства и поведение сетей N_1 и N_2, выводить истинные утверждения о поведении и свойствах сети N, конструируемой из сетей N_1 и N_2 при помощи операций алгебры сетей Петри [9].

Литература

1. HAREL D. First-Order Dynamic logic. - In: Lecture Notes in Computer Science,v.68, Springer Verlag,Berlin,1979.

2. FISHER M.J., LADNER R.E. Propositional dynamic logic of regular programs. - J.Comput.on System Science,1979,v.18, N 2 , p.197-211.

3. PRATT V.R. Process logic.- In: Proc.ACM Symp.on Principles of Programming Languages,1979,p.93-100.

4. PARIKH R. A decidability result for second order process logic.- In: Proc.19th FOCS, October,1978,p.177-183.

5. PNUELI A. The temporal semantics of concurrent programs. - In: Lecture Notes in Computer Science,Springer Verlag, Berlin, 1979,v.70,p.1-20.

6. MANNA Z., PNUELY A. Proving precedence properties: the temporal way.- In: Lecture Notes in Computer Science,v.154,Springer Verlag,Berlin,1983,p.491-512.

7. HACK M. Petri net languages.- In: TR-159,MIT, Lab.Computer Science, Cambridge, Mass.,1976.- 128 p.

8. WOLPER P. Temporal logic can be more expressive. - Information and Control,1983,v.56,N 1/2,p.72-99.

9. KOTOV V.E. An algebra for parallelism based on Petri Nets.- In: Lecture Notes in Computer Science,v.64, Springer Verlag,Berlin,1978,p.39-55.

Поступила в ред.-изд.отд.
26 сентября 1984 года