ДОКЛАДЫ АКАДЕМИИ НАУК, 2005, том 401, № 1, с. 30–33

ИНФОРМАТИКА

УДК 519.713.8

УНИВЕРСАЛЬНЫЕ СЕТИ ПЕТРИ И СИСТЕМЫ ПЕРЕПИСЫВАНИЯ ПРОЦЕССОВ, ДОПОЛНЕННЫЕ ПРОЦЕДУРАМИ

© 2005 г. И. А. Ломазова

Представлено академиком Ю.И. Журавлевым 16.06.2004 г.

Поступило 22.06.2004 г.

В теории вычислимости понятие универсальной в некотором классе вычислительных формализмов машины (программы) играет важную роль. Универсальная программа получает на вход код некоторой программы *PR* из рассматриваемого класса и входные данные для нее и вычисляет результат работы программы *PR* на этих входных данных. В случае реактивных программ или автоматных устройств, когда важен не только результат работы, но и поведение системы, от универсальной программы естественно также требовать симуляции поведения заданной программы. Наряду с универсальной машиной Тьюринга, которая, получая на вход код произвольной машины Тьюринга, моделирует ее работу, универсальные машины существуют и в некоторых ограниченных классах машин, в частности для машин Тьюринга с некоторыми сложностными ограничениями [8]. Однако для многих классов более узких, чем машины Тьюринга, вычислительных формализмов универсальные машины сами не принадлежат этому классу. В частности, в [7] доказано, что в классе конечных автоматов универсального автомата не существует.

В связи с этим представляется интересным, существует ли универсальная сеть Петри. В работе [5] сделано предположение, что в классе обыкновенных сетей Петри универсальной сети не существует и поставлен вопрос о том, какие расширения сетей Петри позволяют получить класс сетей, для которого имеется универсальная сеть из этого же класса. Для решения этой задачи в данной работе мы рассматриваем не собственно сети Петри, а более общий формализм систем переписывания процессов (Process Rewrite Systems – PRS-систем) [9]. В [9] построена классификация PRS-систем, включающая конечные автоматы, алгебры процессов и сети Петри. Обыкновенным сетям Петри в этой классификации в точности соответствует подкласс PRS-систем, называемый (P,P)-PRS. Заметим, что системы взаимодействующих автоматов [4] могут быть представлены в этой классификации подклассом (2, 2)-PRS.

Институт программных систем Российской Академии наук, Москва

В настоящей работе определяется расширение класса (P,P)-PRS за счет добавления конструкции, которая содержательно соответствует описанию процедуры. Сначала мы определяем класс PPRS, который по выразительности также совпадает с классом обыкновенных сетей Петри, но при этом содержит универсальную в этом классе программу. Процедуры в PPRS не содержат параметров и являются по существу операторами. Затем определяется и исследуется расширение этого класса систем за счет введения переменных и соответственно процедур с параметрами, в том числе с процедурными параметрами любого уровня вложенности. Такое расширение мы называем системами переписывания процедур высокого уровня (обозначение HPRS). В работе показано, что класс HPRS также содержит универсальную в нем самом программу, при этом HPRS строго слабее класса вычислимых по Тьюрингу программ. В частности, для систем из HPRS разрешима проблема останова.

Перейдем к определению PPRS-систем. Пусть Act = $\{a, b, \ldots\}$ – бесконечное множество имен действий и Atom = $\{\epsilon, A, B, \ldots\}$ – бесконечное множество атомов. Термы строятся из атомов с помощью операции $\|$ параллельной композиции. Множество термов в PPRS будем обозначать через $\mathcal{T}_0$. Полагаем:

$\varepsilon \in \mathcal{T}_0$, где ε – пустой терм;

для любого $A \in$ Atom $A \in \mathcal{T}_0$;

если $t_1, t_2 \in \mathcal{T}_0$, то $(t_1 \| t_2) \in \mathcal{T}_0$.

Предполагается, что параллельная композиция ассоциативна и коммутативна. Также определено, что $P \| \varepsilon = P$.

Процессы в PPRS строятся из термов с помощью двух операций: параллельной композиции $\|$ и операции образования процедуры $\overset{a}{\mapsto}$, где a – имя действия (процедуры). Множество Π_0 процессов в PPRS определим по индукции:

Для любого $t \in \mathcal{T}_0$ полагаем $t \in \Pi_0$;

если $t_1, t_2 \in \mathcal{T}_0$, то $(t_1 \overset{a}{\mapsto} t_2) \in \Pi_0$;

если $P_1, P_2 \in \Pi_0$, то $(P_1 \| P_2) \in \Pi_0$.

Определение 1. PPRS-система есть пара $\mathcal{P} = (P_0, \Delta)$, где $P_0 \in \Pi_0$ – процесс, задающий начальное состояние системы, Δ – множество правил (подстановок) вида $(t_1 \xrightarrow{a} t_2)$, где t_1 и t_2 – термы, a – имя действия.

Поведение системы задается с помощью отношения перехода $\xrightarrow{a}$. Запись $(P_1 \xrightarrow{a} P_2)$ означает, что из состояния, заданного процессом P_1, посредством действия a система переходит в состояние P_2.

Отношение перехода определяется аксиомой $\mathbf{A}_0$:

$$((t_1 \overset{a}{\mapsto} t_2) \| t_1) \xrightarrow{a} ((t_1 \overset{a}{\mapsto} t_2) \| t_2)$$

и двумя правилами вывода:

$$\mathbf{R}_1\colon \frac{(t_1 \xrightarrow{a} t_2) \in \Delta}{(t_1 \xrightarrow{a} t_2)}, \quad \mathbf{R}_2\colon \frac{t_1 \xrightarrow{a} t_1'}{t_1 \| t_2 \xrightarrow{a} t_1' \| t_2}.$$

Аксиома $\mathbf{A}_0$ соответствует вызову процедуры. Правило $\mathbf{R}_1$ задает применение подстановок из Δ, а правило $\mathbf{R}_2$ позволяет применять подстановку к аргументам параллельной композиции.

Семантика PPRS-системы $\mathcal{P} = (P_0, \Delta)$ задается системой помеченных переходов – бесконечным (вообще говоря) ориентированным деревом, вершины которого представляют достижимые состояния системы и помечены процессами, а дуги задаются отношением $\xrightarrow{a}$ и помечены соответствующими действиями. Корень этого дерева помечен начальным процессом P_0.

Утверждение 1. *Пусть* $\mathcal{P}$ – PPRS-*система с начальным процессом* P_0 *и множеством подстановок* $t_1 \xrightarrow{a_1} t_1', \ldots, t_k \xrightarrow{a_k} t_k'$.

Тогда $\mathcal{P}$ *поведенчески эквивалентна* PPRS-*системе с начальным процессом* $(P \| (t_1 \overset{a_1}{\longmapsto} t_1') \| \ldots \| (t_k \overset{a_k}{\longmapsto} t_k'))$ *и пустым множеством подстановок, а именно системы помеченных переходов, соответствующие* PPRS-*системам* $\mathcal{P}$ *и* $\tilde{\mathcal{P}}$, *изоморфны.*

Таким образом, PPRS-системы $\mathcal{P}$ и $\tilde{\mathcal{P}}$ задают одно и то же вычисление как в смысле достижимости состояний, так и с точки зрения поведенческой (бисимуляционной) эквивалентности. Из приведенного утверждения получаем, что справедлива

Теорема 1. PPRS-*система с пустым множеством подстановок является универсальной в классе* PPRS-*систем.*

Сравним по выразительности PPRS-системы с (обыкновенными) сетями Петри [2].

Определение 2. Сеть Петри – конечный ориентированный граф с вершинами двух типов, называемыми позициями и переходами. Дуга в сети Петри направлена или от позиции к переходу, или от перехода к позиции. Для перехода t позиция p называется входной, если некоторая дуга направлена от p к t, и выходной, если дуга направлена от t к p. Разметка M сети Петри определяется как отображение множества позиций P в множество неотрицательных целых чисел. Говорят, что позиция p содержит k фишек при разметке M, если $M(p) = k$.

Теперь определим поведение сети Петри. Переход t называется активным при разметке M, если любая входная позиция перехода t содержит хотя бы одну фишку при разметке M. Активный переход t может сработать, забирая по одной фишке из каждой своей входной позиции и добавляя по одной фишке в каждую свою выходную позицию. Переходы в сети Петри помечаются именами действий. Семантика сети Петри задается системой помеченных переходов – деревом с корнем, помеченным начальной разметкой сети, и дугами, соответствующими срабатываниям переходов, с пометками – именами этих переходов.

Утверждение 2. *Для любой сети Петри можно эффективно построить* PPRS-*систему* $\mathcal{P}$, *симулирующую поведение сети* PN (*с изоморфной системой помеченных переходов*) *и, обратно, для любой* PPRS-*системы* $\mathcal{P}$ *можно эффективно построить сеть Петри* PN, *симулирующую поведение* $\mathcal{P}$.

Сеть Петри моделируется PPRS-системой следующим образом. Разметка сети, содержащая, скажем, две фишки в позиции A и три фишки в позиции B моделируется термом $(A\|A\|B\|B\|B)$. Переходу сети с входными позициями A, B, выходной позицией C и пометкой a сопоставляется подстановка $A\|B \xrightarrow{a} C$.

При моделировании PPRS-системы $\mathcal{P}$ сетью Петри процедуры из начального процесса и подстановки из $\mathcal{P}$ моделируются переходами, а атомам, входящим в начальный процесс системы $\mathcal{P}$, сопоставляются позиции сети Петри. Таким образом, по вычислительным возможностям PPRS-системы эквивалентны обыкновенным сетям Петри.

Перейдем к определению систем переписывания процессов высокого уровня (HPRS-систем). Пусть наряду с множеством Act имен действий и множеством Atom атомов имеется также счетное множество Var = $\{X, Y, \ldots\}$ переменных. Термы строятся из атомов и переменных с помощью операции параллельной композиции $\|$. По-прежнему предполагается, что параллельная композиция ассоциативна и коммутативна. Также определено, что $P\|\epsilon = P$. Множество термов в HPRS будем обозначать через $\mathcal{T}$.

Процессы в HPRS строятся из термов с помощью операций параллельной композиции $\|$, образования процедуры $\overset{a}{\mapsto}$ и операции образования блоков $\langle\ldots\rangle$. Переменные в описании процедуры играют роль формальных параметров. Блоки определяют область действия процедур.

HPRS-процессы определим по индукции. Параллельно для каждого процесса P будем определять множество $\mathrm{Proc}(P)$ входящих в него описаний процедур. Через $\mathrm{Var}(P)$ будем обозначать множество всех переменных, входящих в процесс P. Множество всех процессов в HPRS обозначим через Π. Полагаем:

для любого $t \in \mathcal{T}$ $t \in \Pi$; $\mathrm{Proc}(t) = \varnothing$;

если $P_1, P_2 \in \Pi$, $\mathrm{Var}(P_2) \subseteq \mathrm{Var}(P_1)$ и $\mathrm{Proc}(P_2) \subseteq \subseteq \mathrm{Proc}(P_1)$, то

$(P_1 \overset{a}{\mapsto} P_2) \in \Pi$; $\mathrm{Proc}(P_1 \overset{a}{\mapsto} P_2) = \{(P_1 \overset{a}{\mapsto} P_2)\}$;

если $P_1, P_2 \in \Pi$, то $(P_1 \| P_2) \in \Pi$; $\mathrm{Proc}(P_1 \| P_2) = \mathrm{Proc}(P_1) \cup \mathrm{Proc}(P_2)$;

если $P \in \Pi$, то $\langle P\rangle \in \Pi$; $\mathrm{Proc}(\langle P\rangle) = \mathrm{Proc}(P)$.

Условие $\mathrm{Var}(P_2) \subseteq \mathrm{Var}(P_1)$ в определении процессов необходимо для того, чтобы исключить бесконечное ветвление в дереве помеченных переходов (новая переменная в результате вычисления могла бы принимать любое из бесконечного множества значений). Условие $\mathrm{Proc}(P_2) \subseteq \mathrm{Proc}(P_1)$ исключает порождение в процессе вычисления новых процедур.

Определение 3. HPRS-система есть пара $\mathcal{P} = (P_0, \Delta)$, где $P_0 \in \Pi$ – процесс, задающий начальное состояние системы, Δ – множество подстановок вида $(P_1 \xrightarrow{a} P_2)$, где P_1 и P_2 – процессы, при этом требуется, чтобы $\mathrm{Var}(P_2) \subseteq \mathrm{Var}(P_1)$ и $\mathrm{Proc}(P_2) \subseteq \mathrm{Proc}(P_1)$.

Конкретизацией процесса P называется процесс P', полученный из P одновременной подстановкой термов вместо некоторых переменных в P. Будем писать $P' \sqsubseteq_c P$, если P' является конкретизацией P.

Отношение перехода в HPRS определяется следующими правилами вывода:

$$\mathbf{R}_1\colon \frac{(P_1 \xrightarrow{a} P_2) \in \Delta}{(P_1 \xrightarrow{a} P_2)}, \quad \mathbf{R}_2\colon \frac{P_1 \xrightarrow{a} P_1'}{P_1 \| P_2 \xrightarrow{a} P_1' \| P_2},$$

$$\mathbf{R}_3\colon \frac{(P_1 \overset{a}{\mapsto} P_2) \sqsubseteq_c (P_1' \overset{a}{\mapsto} P_2')}{((P_1 \overset{a}{\mapsto} P_2) \| P_1') \xrightarrow{a} ((P_1 \overset{a}{\mapsto} P_2) \| P_2')},$$

$$\mathbf{R}_4\colon \frac{P_1 \xrightarrow{a} P_2}{\langle P_1\rangle \xrightarrow{a} \langle P_2\rangle},$$

$$\mathbf{R}_5\colon \frac{((P_1 \overset{a}{\mapsto} P_2) \| P_3) \xrightarrow{a} ((P_1 \overset{a}{\mapsto} P_2) \| P_4)}{((P_1 \overset{a}{\mapsto} P_2) \| \langle P_3\rangle) \xrightarrow{a} ((P_1 \overset{a}{\mapsto} \langle P_2\rangle) \| \langle P_4\rangle)}.$$

Правила $\mathbf{R}_1$ и $\mathbf{R}_2$ здесь такие же, как и для PPRS-систем. Правило $\mathbf{R}_3$ задает применение процедуры с конкретизацией переменных (формальных параметров). Правила $\mathbf{R}_4$ и $\mathbf{R}_5$ определяют область действия процедур так, что блочная структура "закрывает" данные вне блока для процедуры, находящейся внутри блока.

Семантика HPRS-системы $\mathcal{P} = (P_0, \Delta)$, как и в случае PPRS-системы, задается системой помеченных переходов.

Для HPRS-систем имеет место утверждение, аналогичное утверждению 1 для PPRS-систем. Отличие состоит в необходимости блокирования процедур в начальном процессе системы.

Утверждение 3. *Пусть $\mathcal{P}$ – это* HPRS-*система с начальным процессом P_0 и множеством правил $P_1 \xrightarrow{a_1} P_1', \ldots, P_k \xrightarrow{a_k} P_k'$.*

Тогда $\mathcal{P}$ поведенчески эквивалентна HPRS-*системе с начальным процессом $(P \| \langle (P_1 \overset{a_1}{\longmapsto} P_1') \| \ldots \| (P_k \overset{a_k}{\longmapsto} P_k')\rangle)$ и пустым множеством правил.*

Из этого утверждения вытекает следующая

Теорема 2. *В классе* HPRS-*систем* HPRS-*система с пустым множеством правил является универсальной.*

Далее покажем, что HPRS-вычислимость строго слабее вычислимости по Тьюрингу, поскольку для HPRS-систем разрешима проблема останова. Для этого на множестве процессов определим частичный порядок $\sqsubseteq$. Полагаем:

для любого атома A и переменной X

$$A \sqsubseteq A,\ X \sqsubseteq X;\ P_1 \sqsubseteq P_1 \| P_2;$$

если $P_1 \sqsubseteq P_2$, то $\langle P_1\rangle \sqsubseteq \langle P_2\rangle$;

если $(P_1 \overset{a}{\mapsto} P_2) \sqsubseteq_c (P_3 \overset{a}{\mapsto} P_4)$, то $(P_1 \overset{a}{\mapsto} P_2) \sqsubseteq \sqsubseteq (P_3 \overset{a}{\mapsto} P_4)$.

Определение 4 [1]. Квазипорядок $\le$ на множестве X называется правильным, если для любой бесконечной последовательности $x_0, x_1, x_2, \ldots$ элементов из X существуют индексы $i < j$ такие, что $x_i \le x_j$.

Утверждение 4. *Пусть $\mathcal{P}$ – это* HPRS-*система.*

Тогда частичный порядок $\sqsubseteq$ является правильным квазипорядком на множестве достижимых процессов системы $\mathcal{P}$.

Доказательство этого утверждения основано на конечности множеств переменных, атомов и процедур, встречающихся во всех достижимых состояниях (процессах) HPRS-системы, и использует теорему 1.2 из [3] о правильной упорядоченности множества помеченных деревьев ограниченной высоты.

Для доказательства разрешимости проблемы останова используем понятие вполне структурированной системы переходов [6].

Определение 5. Вполне структурированной системой переходов называется система помеченных переходов с множеством состояний S и отношением перехода $\xrightarrow{t}$, дополненная отношением квазипорядка $\leq$ на S так, что:

1) отношение $\leq$ является правильным квазипорядком;

2) квазипорядок $\leq$ совместим с отношением перехода $\xrightarrow{t}$, а именно для любых состояний $s \leq q$ и перехода $s \xrightarrow{t} s'$ существует переход $q \xrightarrow{t} q'$ с той же пометкой t такой, что $s' \leq q'$.

Можно показать, что частичный порядок $\sqsubseteq$ на множестве Π процессов совместим с отношением перехода $\xrightarrow{a}$ для HPRS-систем. Тогда справедлива следующая

Теорема 3. *Всякая* HPRS-*система является вполне структурированной системой переходов относительно частичного порядка* $\sqsubseteq$ *на множестве ее достижимых состояний.*

Напомним, что проблема останова состоит в проверке истинности того, что любое вычисление с данным начальным состоянием завершается (приводит к состоянию, когда никакое действие неприменимо). В [6] было доказано, что для вполне структурированных систем переходов с разрешимым отношением квазипорядка $\leq$ и вычислимым отношением перехода проблема останова разрешима.

Нетрудно проверить, что отношение $\sqsubseteq$ на множестве процессов HPRS-системы разрешимо, а отношение перехода $\xrightarrow{a}$ вычислимо. Отсюда получаем, что справедлива

Теорема 4. *Для любой* HPRS-*системы проблема останова разрешима.*

В заключение заметим, что введение переменных позволяет более компактно представлять PRS-системы и расширяет их выразительные возможности. Поэтому системы переписывания процессов высокого уровня могут быть полезны при моделировании и анализе параллельных и распределенных систем, в частности систем с динамической сетевой структурой.

Работа поддержана программой Президиума РАН (проект 2.23) и Российским фондом фундаментальных исследований (грант 03–01-00804).

СПИСОК ЛИТЕРАТУРЫ

1. *Дистель Р.* Теория графов. Новосибирск: Ин-т математики, 2002. 336 с.
2. *Котов В.Е.* Сети Петри. М.: Наука, 1984. 160 с.
3. *Ломазова И.А.* Вложенные сети Петри: Моделирование и анализ распределенных систем с объектной структурой. М.: Науч. мир, 2004. 208 с.
4. *Ломазова И.А.* // ДАН. 2003. Т. 393. № 3. С. 317–320.
5. *Farwer B., Kudlek M.* // Fundam. Inform. 2004. V. 60. № 1/4. P. 131–142.
6. *Finkel A., Schnoebelen Ph.* // Theor. Comp. Sci. 2001. V. 256. № 1/2. P. 63–92.
7. *Kudlek M.* // Topics in Comput. Math. 2003. V. 9. P. 163–170.
8. *Kudlek M., Margenstern M.* In: Proc. Intern. Conf. “Automata and Formal Languages VIII”. Publ. Math. Debrecen, 1999. V. 53. P. 895–904.
9. *Mayr R.* // Inform. and Comput. 2000. V. 156. № 1. P. 264–286.