ИЕРАРХИЗИРОВАННЫЕ ВЫЧИСЛИМЫЕ

Σ–СПЕЦИФИКАЦИИ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ

ГЛУШКОВА В.Н.
Кафедра “Математика”
Донской государственный технический университет
344000, г. Ростов-на-Дону, пл. Гагарина 1
РОССИЯ
lar@aaanet.ru

The polynomial realized $\Delta_0 T$ –formulas are considered with quantifiers acting on hierarchy lists described by CF-grammars. These formulas are interpreted on a many-sorted model with the hierarchy list superstructure. The constant model is constructed for the Noether, confluent theory based on quasi-identities. The signature predicates and functions are interpreted on input CF-list extended on the interpretation process. The $\Delta_0 T$ –formulas theories might be used to model real-time systems.

Logic formulas with restricted quantifiers, many-sorted model, time complexity of an execution, CF-grammars, logic specification , modelling of discrete systems.

1 Введение

Для спецификации моделируемых систем используется класс Δ_0 - формул с ограниченными кванторами многосортного языка ИП, выделенных в концепции Σ - программирования [1]. Рассматриваемые формулы являются частным случаем Σ - формул, которые допускают описание вычислимых функций на списочных структурах. В этой концепции, основанной на теоретико-модельном подходе, особую роль играют многосортные модели с надстройкой из конечных списков, формируемых из элементов исходной модели. В сигнатуру модели явно вводится сорт “*list*” и стандартные операции и отношения для элементов этого сорта. Списки используются для ограничения области изменения квантифицируемых переменных.

Целью исследования является выделение специального класса Δ_0-формул, которые позволяют организовать эффективные вычисления на деревьях с полиномиальной оценкой сложности относительно “размера” дерева. Дерево – самая распространенная структура данных, допускающая эффективную реализацию. Рассмотрим модели, списочная надстройка которых иерархизируется посредством произвольной КС-грамматики. Это позволяет выделить класс $\Delta_0 T$ - формул, интерпретируемых на списках, представляющих деревьях, за полиномиальное время относительно их “размера” [2]. Выделенный класс формул использовался для проверки контекстных ограничений в системе построения трансляторов с описанием контекстных связей в аппарате ИП 1-го порядка [3]. Класс $\Delta_0 T$ - формул можно применять не только для спецификации статических свойств программ, но и для верификации свойств асинхронных параллельных программ, взаимодействующих посредством разделяемых переменных [4].

Спецификация поведения моделируемой системы состоит из двух взаимосвязанных частей. На первом уровне иерархизируется пространство действий системы посредством грамматики G с правилами для действий вида $Act \to Act_1|...|Act_n$, $Act_i \to St_1 \ldots . St_n$. При выполнении действия Act_i система переходит от состояния St_{j-1} к состоянию St_j, $2\le j\le n$. Правила для состояний St_j, определяются спецификой действия, в них может входить переменная сорта “время”, значениями которой являются константы c – «моментального» времени или интервалы вида $< c_1, c_2 >$, где c_1, c_2 – константы, < заменяется на (или [в зависимости от того, включена левая граница в сегмент времени или нет, аналогично для >.

На втором уровне задается логическая спецификация (теория) поведения системы. Язык ИП позволяет в явном виде выразить свойства о количественных характеристиках объектов системы, зависящих от времени. Для построения модели используются квазитождества специального вида с ограниченными кванторами над *КС*-списками. Алгоритм интерпретации реализует прямой логический вывод для заданной теории, исходя из начальных фактов, причем предикаты, функции и аксиомы теории согласованы с нетерминальными символами и правилами грамматики. Это позволяет при интерпретации теории построить дерево исполнения действий. Для верификации построенной модели используются произвольные $\Delta_0 T$- формулы, специфицирующие ее корректное поведение.

2 Используемые понятия

Пусть M - многосортная модель со списочной надстройкой сигнатуры $\sigma = <\hat{I}, C_M, F, P>$, где $\hat{I} = I \cup \{list\}$ – множество сортов, C_M, F, P– множества констант, функций и предикатов соответственно. Универс U модели M включает семейство носителей $\{U_i\}_{i\in I}$ (U_i – множество элементов сорта i) и надстройку $S^{fin}(M)$ сорта *list*, состоящую из списков, формируемых из элементов множеств U_i. Надстройка $S^{fin}(M) = \bigcup_{n\geq 0} S^n(M)$, где $S^0(M) = \{<\alpha_1,...,\alpha_n> | \alpha_j \in B, 1\leq j\leq n\}$, $B = \bigcup_{i\in I} U_i$, $S^{i+1}(M) = S^i(M) \cup \{<\alpha_1,...,\alpha_n> | \alpha_j \in S^{i-1}(M) \cup B, 1\leq j\leq n\}$. К надстройке добавляется пустой список $nil = <>$. На $S^{fin}(M)$ определяются основные операции и отношения для списков: *conc* (конкатенация списков), *cons* (присоединение к списку нового элемента), $\in$ (“быть элементом списка”) и др.

Упорядоченный набор $<s_1, \ldots, s_n>$, $n \geq 1$, $s_j \in \hat{I}$ или $s_j = <i, list>$, $i \in I$, $1\leq j\leq n$ называется типом над $\hat{I}$. Тип $<s_1>$ будем обозначать s_1 и считать, что $<i, list>$ является подтипом типа *list* и все элементы универса модели типа $<i, list>$ являются также объектами типа *list*. Всем сигнатурным символам, термам и переменным приписывается тип. Термы определяются традиционно. Переменные и константы типа $s \in \hat{I} \cup \{<i, list> | i\in I\}$ будут термами этих же типов. Пусть $f\in F$ функциональный символ типа $<s_1, \ldots, s_n, s>$, тогда $f(t_1,...,t_n)$ терм типа s, если $t_1,...,t_n$ – допустимые для f термы. Аналогичным образом определяются формулы с использованием всех логических связок и ограниченных кванторов вида $\forall x\in t$, $\exists x\in t$, $\forall x \subseteq t$, $\exists x \subseteq t$, где x – переменная произвольного типа , t – терм сорта *list* или $<i, list>$ (t не содержит x).

Иерархизируем списочную надстройку модели M. Пусть $G=(N, T, Pr, A)$ – произвольная *КС*-грамматика, где N, T, Pr – множества нетерминальных, терминальных символов и правил соответственно; A – главный нетерминальный символ, $V = N \cup T$ – множество всех символов G. Определим многосортное множество *КС* - списков. Для этого зададим функцию ρ: $V\rightarrow S_G$, приписывающую всем символам $X\in V$ атрибут сорт. Пусть $C_{\rho(X)}$ не более чем счетное множество констант сорта $\rho(X)$, $C = \{C_{\rho(X)}\}_{X\in V}$ – семейство констант.

Поставим в соответствие произвольному дереву вывода Tr в грамматике G список:

1. По определению $Tr(r)$ – помеченное (символами из V) упорядоченное дерево с выделенным корнем r с меткой A . Если корень r является листом, то ему соответствует константа c сорта $\rho(A)$.

2. Пусть $Tr_1, \ldots, Tr_k$ – поддеревья с корнями $r_1,...,r_k$, которые являются прямыми потомками корня r, помеченными символами $X_1,\ldots, X_k$ (соответственно), тогда $A\rightarrow X_1 \ldots X_k \in Pr$. Деревья Tr_i, $1\leq i\leq k$, являются деревьями разбора в грамматике $G(X_i) = (N, T, Pr, X_i)$. В этом случае дереву Tr соответствует список $<\alpha_1,...,\alpha_k>$ сорта $\rho(A)$. Если Tr_i состоит из единственной вершины, помеченной символом X_i, то α_i – новая константа сорта $\rho(X_i)$, в противном случае α_i – список, соответствующий дереву Tr_i с корнем r_i сорта $\rho(X_i)$.

3. Если корень r имеет единственного потомка, помеченного пустой цепочкой ε, то дереву Tr соответствует список $<c>$ сорта $\rho(A)$, с константой c сорта $\rho(\varepsilon)$.

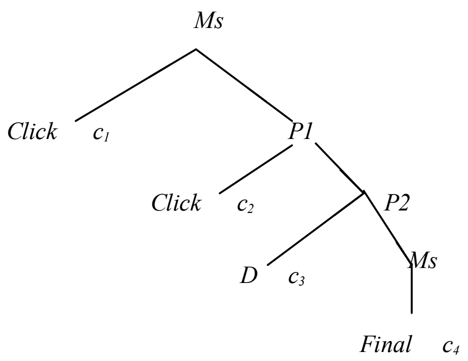
Обозначим через $K(Tr(r))$ список, сопоставляемый дереву $Tr(r)$. По построению этого списка всем узлам-листьям дерева $Tr(r)$ соответствуют уникальные константы – элементы списка. Легко доказать, что при таком представлении дерева отношению непосредственного подчинения его узлов соответствует отношение принадлежности для соответствующих элементов списка $K(Tr(r))$.

<u>*Определение 1*</u>. Для грамматики G множеством *КС*-списков $D_G(C)$ над семейством констант $C = \{C_{\rho(X)}\}_{X\in V}$ называется множество всех списков, сопоставляемых деревьям разбора во всех грамматиках $G(X_i) = (N, T, Pr, X_i)$, $X_i \in N$.
Пусть $\widetilde{K}(X) = \{K(Tr(r)) \mid \textit{метка}(r) = X, X \in N\}$ – множество списков, соответствующих деревьям выводов с корнем X, тогда $D_G(C) = \bigcup\limits_{X\in N} \widetilde{K}(X)$.
Для модели M будем считать, что $S_G \subseteq I$, $C \subseteq C_M$, $D_G(C) \subseteq S^{fin}(M)$. Все элементы $D_G(C)$ имеют сорт *list*, структурируем множество $D_G(C)$ и представим как семейство $\{D_G^X(C)\}_{X\in N}$, где $D_G^X(C)$ – множество списков сорта $\rho(X)$. Списки из $D_G^X(C)$ имеют тип $<list, \rho(X)>$.

<u>*Пример*</u>. Грамматика G_{ms} для манипулятора "мышь" имеет правила:

(1) $Ms \to Click\ \ P1 \mid Final$; (2) $P1 \to Click\ \ P2 \mid S\ \ Ms$; (3) $P2 \to D\ \ Ms$;
(4) $S \to St$; (5) $D \to St$; (6) $Click \to St$; (7) $Final \to St$; (8) $St \to Dtime$.

Перечислим нетерминальные символы: *Ms* – главный; *Click* – имя действия; *S, D, Final* – характеристики результатов действия (простой, двойной щелчок, "финал"); *St* – состояние; *P1, P2* – вспомогательные символы. Особым является символ *Dtime* – указывающий класс лексем, в качестве которого выступает множество N – натуральных чисел. В процессе интерпретации логической спецификации манипулятора (пример ниже) значение n добавляется в дерево в качестве лексемы символа *Dtime*, n указывает номер того действия *Click*, для которого $Click \Rightarrow^* Dtime$.
Дереву вывода: $Ms \Rightarrow Click\ \ P1 \Rightarrow Click\ \ Click\ \ P2 \Rightarrow Click\ \ Click\ \ D\ Ms \Rightarrow Click\ \ Click\ \ D\ \ final$, изображенному на Фиг.1, соответствует список $\delta = < c_1, < c_2, < c_3, < c_4 >>>>$.



Фиг.1

Константы c_i сопоставляются висячим узлам, сорта констант c_i и списков определяются метками соответствующих узлов дерева и обозначаются мнемонично строчными буквами с добавлением s в конец имени сорта: $\rho(\delta) = mss$, $\rho(c_1) = \rho(c_2) = clicks$, $\rho(< c_2, < c_3, < c_4 >>>) = p1s$, $\rho(< c_3, < c_4 >>>) = p2s$, $\rho(c_3) = ds$, $\rho(< c_4 >) = mss$, $\rho(c_4) = finals$.

Приведем определение Δ_0–формул с "древесным" префиксом, в которых используются лишь ограниченные кванторы вида $\forall\ v \in r$, $\forall\ v \overset{*}{\in} r$, $\forall\ u \prec v$, где u, v – переменные любого сорта; r – переменная сорта *list*. Отношение $\overset{*}{\in}$ – рефлексивное транзитивное замыкание $\in$; $\prec$ – отношение «левее» для элементов списочных объектов. Пусть $\bar{x}$ обозначает индексированную последовательность переменных x, $\dot{\in}$ – отношение $\in$ или $\overset{*}{\in}$.

<u>*Определение 2*</u>. Δ_0–формула вида

$$(\forall v_1 \dot{\in} r_1) \ldots (\forall v_m \dot{\in} r_m)\ (w_1 \prec u_1) \ldots\ (w_k \prec u_k)\ \Phi(\bar{v}, \bar{r}),\ m \geq 1,\ k \geq 0 \qquad (1)$$

называется $\Delta_0 T$ – формулой, если w_j, $u_j \in (\bar{v}, \bar{r})$, $1 \le j \le k$, и переменные префикса удовлетворяют условию “древесности”: для всех $1 \le i \le m$, $l \le i$ выполняется $r_{i+1} = r_i$ или $r_{i+1} = v_l$. Если $r_{i+1} = v_l$, то для всех $p \le i$ выполняется $v_{i+1} \ne v_p$, $v_{i+1} \ne r_p$.

Если представить все переменные префикса узлами с дугами, ведущими от r_i к v_i, то получится дерево с корнем r_1.

3 Интерпретация

Модель M строится по теории Th из квазитождеств с отрицаниями вида:

$$(\forall v_1 \dot{\in} r_1)\ldots(\forall v_m \dot{\in} r_m)\ (w_1 \prec u_1)\ \ldots\ \ (w_k \prec u_k)\ (\varphi(\overline{\alpha}) \to \psi(\overline{\alpha})) \qquad (2)$$

Здесь $\overline{\alpha}$ – последовательность переменных $\bar{v}, \bar{r}$; формула $\varphi(\psi)$ – конъюнкция атомных формул или их отрицаний вида: p, $f=t$, $(t_1=t_2)$, t_1, t_2 – термы сигнатуры σ. В формулу ψ отрицания не входят.

Теория Th рассматривается как логическая спецификация для вычисления функций и отношений на списках носителя модели. Если Th обладает свойством нётеровости и конфлюентности, можно построить индуктивно вычислимую модель из констант C модели M на основе правила вывода “modus ponens”, исходя из начального списка δ_0 и известных фактов (формул вида $r(\bar{c})$, $f(\bar{c})=d$, $\bar{c}$ – последовательность констант, d – константа). Иерархичность списков позволяет синтаксически ориентированным способом найти все элементы, удовлетворяющие префиксу формулы. А именно, любому списку однозначно сопоставляется дерево вывода в грамматике G, которое является корневым деревом с произвольным ветвлением. Такие деревья можно представлять по принципу “левый ребенок-правый сосед” с хранением сорта узла в специальном поле. Поставим в соответствие каждому списку корень соответствующего ему дерева, т.е. закодируем все узлы дерева константами соответствующих сортов с сохранением этого кода в поле узла. Кортеж значений $\bar{c}$, удовлетворяющих префиксу формулы, строится за один просмотр списка по принципу “сверху-вниз, слева-направо” [2].

На следующем шаге реализуется логический вывод: проверяется истинность левых частей всех аксиом вида (2), в случае истинности $\varphi(\bar{c})$ добавляются новые следствия – все значения предикатов и функций, входящих в $\psi(\bar{c})$. Процесс продолжается до получения неподвижной точки вычисления, т.е. до того шага вывода, при котором новые факты уже не появляются. При этом все аксиомы с позитивными левыми частями интерпретируются до аксиом, содержащих негативные вхождения. Арифметические операции при интерпретации считаются встроенными, равенство интерпретируется как слабое, т.е. ложное, если одно из значений терма неопределенно. Отрицание для предикатов трактуется по принципу "замкнутого мира".

Обозначим $Lex(G)$ – множество нетерминальных - классов лексем, а множество нетерминальных, из которых выводятся эти символы через $Ld(G) = \{X \mid X \Rightarrow^* \alpha Y \beta,\ \alpha, \beta \in V^*,\ Y \in Lex(G)\}$. Все символы из $Ld(G)$ согласованы с различными предикатными символами p сигнатуры σ; будем считать, что они совпадают по имени: $Ld(G) \subseteq P$. Тогда имена областей определения этих предикатов p являются классами лексем, для $p \subseteq D_1 \times \ldots \times D_n$, $имя(D_j) \in Lex(G)$, $1 \le j \le n$ и $имя\ (D_j) = D_j$. Всем аксиомам ax, в правой части которых появляется такой предикат p, сопоставляется последовательность правил $pr \in Pr^*$ грамматики G. Пусть дереву вывода, полученному до интерпретации аксиомы ax, соответствует цепочка $\alpha \in V^*$. Последовательность правил pr подобрана так, что с ее применением $\alpha \ \Rightarrow_l^* \alpha_l\, p \alpha_2$; $\alpha_l, \alpha_2 \in V^*$; $p \Rightarrow_l^* D_1 \ldots D_n$, причем тип предиката p – $< s_1, \ldots, s_n >$, $s_j = \rho\ (D_j)$, $1 \le j \le n$. Пусть при интерпретации устанавливается истинность предиката $p\ (\bar{d})$ для последовательности констант $d_1, \ldots, d_n$. Тогда при применении правил pr в выводе в качестве лексем класса D_j выбираются константы d_j. Конечным результатом интерпретации является дерево, отражающее последовательность шагов вычисления предикатов и функций, в результате которой область их определения расширяется до получения ”неподвижной точки” вычисления.

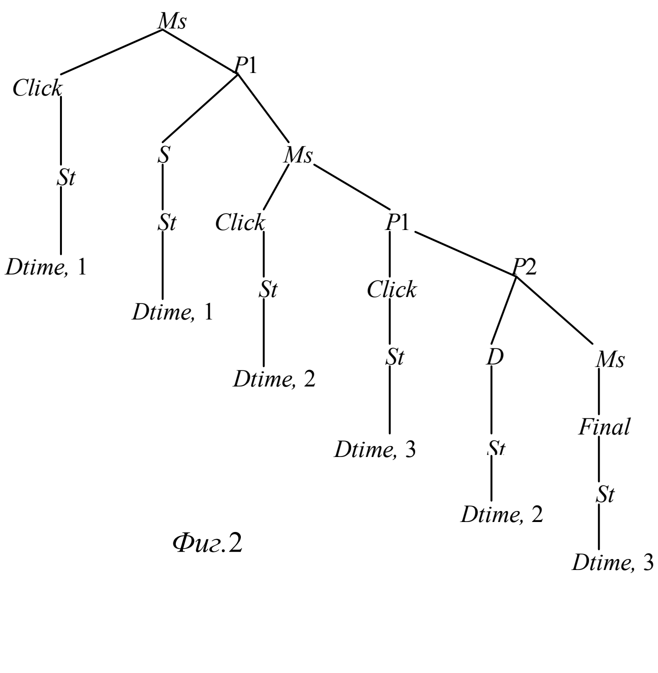
4 Пример

Приведем логическую спецификацию поведения манипулятора “мышь”, в которой различаются одинарный и двойной щелчок. Ниже используются одноместные предикаты: $Click \subseteq Dtime$ (“нажатие кнопки мыши”); S, $D \subseteq Dtime$ (события “одинарный” и “двойной” щелчки); $Final \subseteq Dtime$ (определяет номер последнего щелчка); функция T: $Dtime \to S^0(M)$, для $T(0) == <t_1,...,t_n>$ константа t_i – время исполнения i-го действия “*Click”*. Область $Dtime \subset N$, где N– множество натуральных чисел, значение $n \in N$ – это номер щелчка; l – граница временного интервала, ниже которой два последовательных нажатия считаются одним двойным щелчком. Переменные обозначаются мнемонично в соответствии с их сортом: *st*– переменная сорта *sts* (“состояние”), ее значением является список, представляющий дерево вывода с корнем, помеченным символом *St*, $\rho(n)$ =*dtimes*. Все аксиомы интерпретируются на заданном списке *ms*, который расширяется в процессе интерпретации в результате применения к соответствующему выводу последовательности правил грамматики G_{ms}, приведенных в аксиомах в квадратных скобках. Все выводы грамматики являются левыми. Исходный список для переменной *ms* определяется аксиомами 2, 3.

Используются стандартные функции: $h(<x_1,...x_n>) = x_1$; $tail(<x_1,...x_n>) = <x_2,...x_n>$. Фактами теории являются значения $h\,(T(0)) = t_1$ и $h(tail(T(0))) = t_2$, если $tail(T(0)) \neq nil$.

1. $tail(T(0)) = nil \to Click\,(1), S(1)$ [1.1; 6; 8; 2.2; 4; 8]
2. $h(tail(T(0))) - h\,(T(0)) < l \to Click\,(1), Click\,(2), D(1)$ [1.1; 6; 8; 2.1; 6; 8; 3; 5; 8]
3. $h(tail(T(0))) - h\,(T(0)) \geq l \to Click\,(1), S(1)$ [1.1; 6; 7; 2.2; 4; 8]
4. $(\forall st \,\dot{\in}\, ms)\ (\forall n \,\dot{\in}\, st)\,(S(n) \to\ T(n) = tail(T(n-1)))$
5. $(\forall st \,\dot{\in}\, ms)\ (\forall n \,\dot{\in}\, st)\,(D(n) \to\ T(n+1) = tail\,(tail(T(n-1))))$
6. $(\forall click \,\dot{\in}\, ms)\ (\forall n \,\dot{\in}\, click)(\ Click(n), T(n) = nil \to Final(n))$ [1.2; 7; 8]
7. $(\forall click \,\dot{\in}\, ms)\ (\forall n \,\dot{\in}\, click)\ \ (Click(n), tail(T(n)) = nil \to Click\,(n+1), S(n+1))$ [1.1; 6; 8; 2.2; 4; 8]
8. $(\forall click \,\dot{\in}\, ms)\ (\forall n \,\dot{\in}\, click)\ \ (h(tail(T(n))) - h\,(T(n)) < l \to Click\,(n), Click\,(n+1), D(n))$
   [1.1; 6; 8; 2.1; 6; 8; 3; 5; 8]
9. $(\forall click \,\dot{\in}\, ms)\ (\forall n \,\dot{\in}\, click)\ \ (h(tail(T(n))) - h\,(T(n)) \geq l \to Click\,(n), S(n)\ )$ [1.1; 6; 8; 2.2; 4; 8]
10. $(\forall st \,\dot{\in}\, ms)\ (\forall n \,\dot{\in}\, st)\ \ (S(n) \to T(n) = tail(T(n-1))$
11. $(\forall st \,\dot{\in}\, ms)\ (\forall n \,\dot{\in}\, st)\,(\ D(n) \to\ T(n+1) = tail\,(tail(T(n-1)))$

Пусть $T(0) = <t_1,\ t_2,\ t_3>$ и $t_1 - t_2 \geq l$, $t_3 - t_2 < l$. Исходным для заданных фактов является дерево, соответствующее выводу: $Ms \Rightarrow Click\ \ P1 \Rightarrow St\ P1 \Rightarrow Dtime\ P1 \Rightarrow S\ Ms \Rightarrow St\ Ms \Rightarrow Dtime\ Ms$ с лексемой 1 для символа *Dtime*. В результате интерпретации получаем дерево (*Фиг*.2) и следствия: *Click* (1), *S*(1), *Click* (2), *Click* (3), *D*(2), $T(2) = <t_1,\ t_2>$, $T(3) = < >$, *Final* (3).



Фиг.2

Литература:

[1] Goncharov S.S., Ershov Yu.L., and Sviridenko D.I. Semantic programming. Information Processing, 86, 1093-1100, North-Holland, Amsterdam,1986

[2] Glushkova V.N. Estimating the execution complexity of logical specifications based on context-free grammars. Kibernetika i Sistemnyi Analiz. No. 4, July-August, 1996. pp.50-58.

[3] Glushkova V.N. , Il'icheva O.A. Computer aided parsing and context analysis in compiler construction system. Kibernetika, No.4, 1985, pp. 26-28.

[4] Глушкова В.Н. Гибридные логические модели. Научное программное обеспечение в образовании и научных исследованиях. Труды научно-технической конференции, 30-31 января, 2008 г. Санкт-Петербург, Изд-во Политехнического ун-та, 2008, стр. 21-28.