ЛОГИЧЕСКИЕ МЕТОДЫ В ПРОГРАММИРОВАНИИ
(Вычислительные системы)
1987 год Выпуск 120

УДК 519.68

ОБ ОДНОМ ОБОГАЩЕНИИ ЯЗЫКА Σ^+-ПРОГРАММ

Д.И. Свириденко

В работе [I] в качестве языковой основы определения Σ^+-программ использовался язык $L_0(\sigma^*)$. Для теоретических целей этот язык вполне удовлетворителен. Однако его практическое использование даже в достаточно простых ситуациях приводит к появлению весьма громоздких конструкций. Поэтому, естественно, возникает желание ввести в язык дополнительные средства, которые существенно улучшили бы его выразительные возможности. В настоящей статье обсуждается один из таких вариантов расширения языка $L_0(\sigma^*)$, что позволяет писать не только более лаконичные Σ^+-программы, но и строить более эффективные алгоритмы их исполнения. В то же время данное расширение, которое мы будем обозначать $L_1(\sigma^*)$, допускает простую трансляцию в язык $L_0(\sigma^*)$, а следовательно, и трансляцию расширенных Σ^+-программ в Σ^+-программы языка $L_0(\sigma^*)$. Тем самым фактически определяется и процедурная семантика расширенных Σ^+-программ.

Пусть σ_0, σ и σ^* - сигнатуры, введенные в [I]. Дадим определение термов и формул языка $L_1(\sigma^*)$ (обращаем внимание читателя на то, что введение данных синтаксических категорий взаимообусловлено). Множество термов Term сигнатуры σ^* определяется следующим образом:

I) если t - σ-терм (см. [I]), то $t \in$ Term;

2) если $\varphi(x)$ - Δ^+-формула языка $L_1(\sigma^*)$ и $t \in$ Term, то конструкция $\langle x \mid x \in t \wedge \varphi(x) \rangle$ есть терм типа list; свободными переменными этого терма будут переменные терма t и формулы φ, за исключением переменной x;

3) если $F \in \mathbf{F}$ - n-местная функциональная переменная типа $K_1, \ldots, K_n \to K$, а $t_1 : K_1, \ldots, t_n : K_n$ - термы, то $F(t_1, \ldots, t_n) \in$ Term и имеет тип K;

4) других термов нет.

В дальнейшем мы будем предполагать наличие в сигнатуре $\sigma^* \setminus \sigma$ специального унарного предиката D типа I ∪ {list}. Атомарная формула D(t) может понимать как "t есть обозначение некоторого объекта". Класс формул языка $L_1(\sigma^*)$ вводится аналогично тому, как это делается в [I]. Понятие Δ_0^+-, Σ^+- и Δ^+-формул определяется обычным образом включая требование, чтобы формулы вида D(t) входили в Δ^+- и Σ^+-формулы позитивно.

Опишем теперь стандартную семантику языка $L_1(\sigma^*)$. С этой целью вместо модели $\mathcal{N}_0 = \langle \mathcal{M}, HW(\mathcal{M}), \tilde{P}, \tilde{F}_0 \rangle$ языка $L_0(\sigma^*)$ будем рассматривать четверку $\mathcal{N}_1 = \langle \mathcal{M}, HW(\mathcal{M}), \tilde{P}, \tilde{F}_1 \rangle$, где $\tilde{F}_1$ - класс частичных функций (а не класс графиков функций $\tilde{F}_0$ в модели $\mathcal{N}_0$, как это предлагалось в [I]).

Пусть $\theta = \langle \nu, \mu, \eta \rangle$ - некоторая интерпретация предметных, предикатных и функциональных переменных в модели (исключая предикатную переменную D). Прежде всего мы должны в модели $\mathcal{N}_1$ определить значение термов при этой интерпретации. Предварительно введем некоторые определения.

Пусть a - список из $HW(\mathcal{M})$. На множестве $|\mathcal{M}| \cup |HW(\mathcal{M})|$ зададим Δ_0-отношение $\leq_a$:

$$x \leq_a y \leftrightarrows \exists \alpha \sqsubseteq a\ \exists \beta \sqsubseteq a\ \forall \varepsilon \sqsubseteq a[\alpha \neq nil \wedge$$

$$\wedge \beta \neq nil \wedge \alpha \sqsubseteq \beta \wedge x = head(\alpha) \wedge y = head(\beta) \wedge$$

$$\wedge (x = head(\varepsilon) \supset \alpha \sqsubseteq \varepsilon) \wedge (y = head(\varepsilon) \supset \beta \sqsubseteq \varepsilon)].$$

Таким образом, $x \leq_a y$, если x и y - элементы списка a, и первое вхождение элемента x предшествует первому вхождению элемента y. Будем говорить, что список a есть список без повторений, если выполняется следующее Δ_0-свойство:

$$R(a) \leftrightarrows \forall \alpha \sqsubseteq a\ \forall \beta \sqsubseteq a(head(\alpha) = head(\beta) \supset \alpha = \beta).$$

Используя отношение $\leq_a$ и R, вводим бинарное Δ_0-отношение на списках: $a \leq b \leftrightarrows [R(a) \wedge \forall x, y \in a(x \leq_a y \supset x \leq_b y)]$.

Справедлива следующая

ЛЕММА I. Для любых списков a и b выполняются условия:

а) $nil \leq a$;

б) $a \leq b \to a \sqsubseteq b (\doteq \forall x \in a\, x \in b)$;

в) $R(a) \to a \leq a$;

г) $a \leq b \wedge b \leq a \to a = b$;

д) $a \leq b \wedge b \leq c \to a \leq c$;

е) $R(b) \wedge a \sqsubseteq b \to a \leq b$.

Определим индуктивно функцию $ЗН_\theta$: $\underline{Term} \to |\mathcal{M}| \cup |HW(\mathcal{M})|$, вычисляющую значения термов в $\mathcal{N}_1$ при интерпретации θ:

1) если t – σ-терм, то $ЗН_\theta(t)$ определяется обычным образом;

2) если $t = \langle x | x \in q \wedge \varphi \rangle$, то $ЗН_\theta(t) = a$, где $a \leq ЗН_\theta(q)$ и для каждого $y \in ЗН_\theta(q)$ справедливо

$$(\mathcal{N}_1, \theta) \models [(\varphi(y) \supset y \in a) \wedge (\neg \varphi(y) \supset y \notin a)];$$

3) если $t = F(t_1, \ldots, t_n)$ и $\eta(F) = \hat{F} \in \tilde{\mathbf{F}}_1$, то

$$ЗН_\theta(t) = \hat{F}(ЗН_\theta(t_1), \ldots, ЗН_\theta(t_n)).$$

Поскольку определение $ЗН_\theta$ индуктивное, то в качестве $ЗН_\theta$ необходимо взять наименьшую неподвижную точку соответствующего данному определению функционала. Обратим также внимание читателя на то, что определение функции $ЗН_\theta$ зависит от определения семантики формул языка $L_1(\sigma^*)$.

Используя функцию $ЗН_\theta$, можно теперь уточнить и семантику предиката D:

$$(\mathcal{N}_1, \theta) \models D(t) \leftrightarrow ЗН_\theta(t) \text{ определено.}$$

Семантика остальных формул языка $L_1(\sigma^*)$ определяется обычным образом.

Отметим следующее простое, но полезное наблюдение. Будем говорить, что список a есть <u>фактор</u> списка b (в символах $\underline{fact}(a,b)$), если $a \leq b$ и $b \subseteq a$. На роль факторизации списков указывает

ЛЕММА 2. а) Для любого списка a выполняется $(\mathcal{N}_1, \theta) \models \underline{fact}(\langle x | x \in a \wedge x = x \rangle, a)$.

б) Если $(\mathcal{N}_1, \theta) \models \underline{fact}(a,b)$, то для любой формулы φ справедливы

$$(\mathcal{N}_1, \theta) \models \forall x \in a.\varphi \leftrightarrow (\mathcal{N}_1, \theta) \models \forall x \in b.\varphi$$

и

$$(\mathcal{N}_1, \theta) \models \exists x \in a.\varphi \leftrightarrow (\mathcal{N}_1, \theta) \models \exists x \in b.\varphi .$$

Определим теперь трансляцию языка $L_1(\sigma^*)$ в язык $L_0(\sigma^*)$.

Пусть $t \in \underline{Term}$ и x - переменная, не входящая в t. Через $\hat{x}$ обозначим преобразование терма t:

$$\hat{x}(t) \leftrightarrows \begin{cases} x, \text{ если } t \text{ не } \sigma\text{-терм;} \\ t - \text{в противном случае.} \end{cases}$$

Если дан набор термов $t_1 \ldots t_n$ и $x_1 \ldots x_n$ - набор переменных, не встречающихся ни в одном из термов t_i, $i = 1, \ldots, n$, то полагаем $\widehat{x_1 \ldots x_n}(t_1 \ldots t_n) \leftrightarrows \hat{x}_1(t_1) \ldots \hat{x}_n(t_n)$. Обозначим через $Var(t_1, \ldots, t_n)$ набор всех переменных, встречающихся в $\widehat{x_1 \ldots x_n}(t_1 \ldots t_n)$, как элементы этого набора. Таким образом, если $x_{i_j} \in Var(t_1, \ldots, t_n)$, то t_{i_j} - это терм из $t_1 \ldots t_n$, соответствующий x_{i_j}.

Соотнесем каждому не σ-терму t формулу $\varphi_t(x)$ языка $L_0(\sigma^*)$, интуитивно выражающую тот факт, что терм t определен и его значение есть x. Полагаем:

1) если терм имеет вид $f(t_1, \ldots, t_n)$, где $f \in \sigma^*$ - функциональный символ, то

$$\varphi_t(x) \leftrightarrows \exists x_{i_1} \ldots x_{i_m} \Big(\bigwedge_{x_{i_j} \in Var(t_1 \ldots t_n)} \varphi_{t_{i_j}}(x_{i_j}) \wedge$$

$$\wedge\, x = f(\widehat{x_1 \ldots x_n})(t_1, \ldots, t_n)) \Big);$$

2) если терм t имеет вид $\langle z \mid z \in q \wedge \varphi \rangle$, то

$$\varphi_t(x) = \begin{cases} x \le q \wedge \forall y \in q((Tr(\varphi(y)) \wedge y \in x) \vee (Tr(\neg\varphi(y)) \wedge \\ \qquad \wedge\, y \notin x)), \text{ если } q - \sigma\text{-терм;} \\ \exists z[\varphi_q(z) \wedge x \le z \wedge \forall y \in z((Tr(\varphi(y)) \wedge y \in x) \vee \\ \qquad \vee (Tr(\neg\varphi(y)) \wedge y \notin x))] - \text{в противном случае.} \end{cases}$$

Здесь $Tr(\varphi)$ - результат трансляции формулы φ (см. ниже). Сама же трансляция формул из $L_1(\sigma^*)$ в $L_0(\sigma^*)$ задается следующей схемой:

1) Формула φ приводится к негативной нормальной форме φ' (см. [1]).

2) Если в φ встречается подформула вида $D(t)$, где t - σ-терм, то эта подформула из φ удаляется.

3) если t не σ-терм , то вхождение подформулы $D(t)$ заменяется на вхождение подформулы $\exists x.\ \varphi_t(x)$.

4) Пусть $P \in \sigma_0 \cup (\sigma^e \setminus \sigma)$ - предикатный символ. Если $P(t_1,\ldots,t_n)$ входит в φ' позитивно, то это вхождение заменяется на подформулу

$$\exists x_{i_1}\ldots x_{i_m}\Big(\bigwedge_{x_{i_j}\in Var(t_1,\ldots,t_n)} \varphi_{t_{i_j}}(x_{i_j}) \wedge P(\widehat{x_1\ldots x_n}(t_1,\ldots,t_n))\Big).$$

Если же $P(t_1,\ldots,t_n)$ входит негативно, то $\neg P(t_1,\ldots,t_n)$ заменяется на подформулу

$$\exists x_{i_1}\ldots x_{i_m}\Big(\bigwedge_{x_{i_j}\in Var(t_1,\ldots,t_n)} \varphi_{t_{i_j}}(x_{i_j}) \wedge \neg P(\widehat{x_1\ldots x_n}(t_1,\ldots,t_n))\Big).$$

5) Пусть $P \in \{\in,\sqsubseteq\}$. Трансляция соответствующих атомных формул осуществляется в точном соответствии с п.4 , кроме случая $s\in t$ и $s\sqsubseteq t$, где s не σ-терм,а t - σ-терм.В этих случаях $s\in t$ вначале заменяется на $\exists x\in t\ (x = s)$, а далее применяется п. 4 . Аналогично $s\sqsubseteq t$ вначале заменяется на $\exists x \sqsubseteq t(x= s)$ и затем применяется п.4 .

Далее будем обозначать результат трансляции формулы φ через $Tr(\varphi)$. Тогда для более сложных формул трансляция определяется так:

6) $Tr(\varphi\wedge\psi) = Tr(\varphi)\wedge Tr(\psi)$;
$Tr(\varphi\vee\psi) = Tr(\varphi)\vee Tr(\psi)$:
$Tr(\forall x.\varphi) = \forall x.Tr(\varphi)$;
$Tr(\exists x.\varphi) = \exists x.\ Tr(\varphi)$;

7) Пусть $\varphi = \forall x\in t.\varphi_1$.

7.1) Если t - σ-терм, то $Tr(\forall x\in t.\ \varphi_1) = \forall x\in t.\ Tr(\varphi_1)$.

7.2) Если $t = \langle x|x\in q\wedge\psi\rangle$,то $Tr(\forall x\in t.\varphi_1)=Tr(\forall x\in q.(\varphi_1\supset\psi))$.

7.3) Если $t = F(t_1,\ldots,t_n)$,где $F\in\sigma^e\setminus\sigma$, то
$Tr(\forall x\in t.\varphi_1) = \exists y(\varphi_t(y)\wedge\forall x\in y.\ Tr(\varphi_1))$.

Аналогично выглядит схема трансляции для формул вида $\forall x\sqsubseteq t.\ \varphi_1$.

8) Пусть $\varphi = \exists x\in t.\ \varphi_1$.

8.1) Если t - σ-терм, то $Tr(\exists x\in t.\ \varphi_1) = \exists x\in t.\ Tr(\varphi_1)$.

8.2) Если $t = \langle x|x\in q\wedge\psi\rangle$, то
$Tr(\exists x\in t.\varphi_1) = Tr(\exists x\in q.\ (\varphi_1\wedge\psi))$.

8.3) Если $t = F(t_1, \ldots, t_n)$, где $F \in \sigma^* \setminus \sigma$, то $Tr(\exists x \in t.\varphi_1) = \exists y(\varphi_t(y) \wedge \exists x \in y.\, Tr(\varphi_1))$. Аналогично определяется схема трансляции для формул вида $\exists x \subseteq t.\varphi_1$.

Непосредственно из определений вытекает следующая

ЛЕММА 3. Пусть $\varphi \in L_1(\sigma^*)$.

а) Если $\varphi \in \Sigma^+_{L_1}$, то $Tr(\varphi) \in \Sigma^+_{L_0}$;

б) если $\varphi \in \Delta^+_{L_1}(\Delta_{L_1})$, то $Tr(\varphi) \in \Delta^+_{L_0}(\Delta_{L_0})$;

в) $\varphi \in \Sigma^+_{L_0} \Leftrightarrow \varphi \in \Sigma^+_{L_1} \wedge Tr(\varphi) = \varphi$.

Пусть $\mathcal{N}_0 = (\mathcal{M}, HW(\mathcal{M}), \tilde{F}, \tilde{F}_0)$ – модель языка $L_0(\sigma^*)$ и $\mathcal{N}_1 = (\mathcal{M}, HW(\mathcal{M}), \tilde{F}, \tilde{F}_1)$ – соответствующая ей модель языка $L_1(\sigma^*)$.

Справедлива следующая

ТЕОРЕМА. Пусть $\varphi \in L_1(\sigma^*)$. Тогда

$$\mathcal{N}_1 \models \varphi \Leftrightarrow \mathcal{N}_0 = Tr(\varphi).$$

Этот результат естественно интерпретируется как утверждение о корректности вышеприведенной схемы трансляции. В рамках языка $L_1(\sigma^*)$, аналогично тому, как это делалось в [1], можно определить конструкцию (расширенной) Σ^+-программы, а используя преобразование Tr, легко построить соответствующую трансляцию Σ^+-программ языка $L_1(\sigma^*)$ в Σ^+-программы языка $L_0(\sigma^*)$. Для этой схемы трансляции Σ^+-программ можно также доказать аналог сформулированной выше теоремы.

Отметим, что в языке $L_1(\sigma^*)$ более удобно выразимы функциональные схемы определений. Более того, для достаточно широкого класса, скажем, ЛИСП-подобных определений, можно указать их естественное представление в виде расширенных Σ^+-схем. В качестве примера можно указать один из возможных способов записи в языке $L_1(\sigma^*)$ широко используемой функциональной конструкции вида

$$g(\bar{x}) = \underline{\text{если}}\ \varphi(x)\ \underline{\text{то}}\ f(\bar{x})\ \underline{\text{иначе}}\ h(\bar{x}).$$

Если φ – Δ^+-формула языка $L_1(\sigma^*)$, а t_1, t_2 – термы, то полагаем

$\underline{\text{Если}}\ \varphi\ \underline{\text{то}}\ t_1\ \underline{\text{иначе}}\ t_2 \Rightarrow$

$$head(\langle x | x \in \langle t_1, t_2 \rangle \wedge [(\varphi \wedge x = t_1) \vee (\neg\varphi \wedge x = t_2)]).$$

Если же в сигнатуре σ_0 предусмотреть наличие функции – генератора (псевдо)случайных чисел rand(n) (значением является произвольное (случайно выбранное) число из $[1, \ldots, n]$), то в $L_1(\sigma^*)$

легко моделируется и хорошо известная конструкция Дейкстры:

$$\text{IF } \varphi_1 \to t_1 \,\square\, \ldots \,\square\, \varphi_n \text{ FI}$$

(здесь $\varphi_1,\ldots,\varphi_n$ – Σ^+-формулы, $t_1,\ldots,t_n$ – термы).

В классе расширенных Σ^+-программ оказываются Σ^+-представимыми параметрические логические программы с равенством (см.[I, §4]).

В рамках языка $L_1(\sigma^*)$ оказывается возможным определить (функциональный) стиль программирования, позволяющий оперировать с объектами (списками), которые можно интерпретировать как бесконечные. Более того, оказывается возможным определить механизм вычислений, аналогичный механизму "ленивых вычислений", а также ввести в рассмотрение понятие процесса и определять системы взаимодействующих процессов.

В заключение сформулируем одну проблему. Заметим, что теорема позволяет вопрос об изучении свойств Σ^+-формул и Σ^+-программ языка $L_1(\sigma^*)$ эффективно сводить к аналогичным вопросам, касающимся уже формул и программ языка $L_0(\sigma^*)$. И здесь одним из инструментов исследования может служить теория списочных надстроек GES_1^+.. Однако можно поставить задачу построения соответствующей теории, но уже для языка $L_1(\sigma^*)$, учитывающей особенности этого языка и, в частности, таких термообразующих конструкций, как $\langle x | x \in t \wedge \varphi \rangle$ и Σ^+-определимых функций. При этом необходимо учесть, что в языке $L_1(\sigma^*)$ мы уже работаем с частичными объектами. Для подобных объектов, помимо равенства, может понадобиться следующее отношение эквивалентности: $t \cong s = (D(t) \supset t = s) \wedge (D(s) \supset t = s)$. Используя это отношение, можно к (возможно, модифицированной) GES_1^+ добавить следующую аксиому:

$$(t \cong s \wedge \varphi(t)) \supset \varphi(s).$$

Если $R \in \sigma^*$ – n-местный предикатный символ, то необходимо также позаботиться и о том, чтобы выполнялась импликация:

$$R(t_1,\ldots,t_n) \supset \bigvee_{i=1}^{n} D(t_i) .$$

Естественно, что если t – σ-терм, то всегда имеет место $D(t)$. Полезными могут оказаться следующие импликации:

$$\varphi(x) \wedge D(t) \supset \varphi(t),$$

$$(\varphi)_t^x \wedge D(t) \supset \exists x.\, \varphi$$

и т.п.

Другими словами, ставится проблема построения обобщенной вычислимости на списочных надстройках, учитывающая частичность определяемых объектов.

Литература

I. ГОНЧАРОВ С С., СВИРИДЕНКО Д.И. Σ^+-программы и их семантика. - Настоящий сборник. - С.24-5I.

Поступила в ред.-изд.отд.
23 июня 1987 года