ПРИКЛАДНЫЕ АСПЕКТЫ МАТЕМАТИЧЕСКОЙ ЛОГИКИ
(Вычислительные системы)
1987 год Выпуск 122

УДК 519.47

СВЯЗЬ СЕТЕВОГО И ЛОГИЧЕСКОГО ПОДХОДОВ К ОПИСАНИЮ ПРОЦЕССОВ

В.Е.Котов, Л.А.Черкасова

Для описания функционирования дискретных динамических систем и процессов большое использование получили сети Петри, при этом переходы в сети выполняют роль событий в системе, а места соответствуют условиям наступления событий. При анализе и синтезе дискретных систем обработки информации возникает задача установления взаимосвязи между структурой системы и характером протекающих в ней процессов. В работах [1,2] и др. предложен и развит сетевой подход к описанию систем и процессов, при котором системы и порождаемые ими процессы описываются в рамках единого формализма сетей Петри. При таком определении процесс задает не единственный временной протокол (историю реализации событий), а некоторый класс временных протоколов, обусловленных одним и тем же набором причинно-следственных отношений между действиями и условиями.

Сложное поведение параллельных асинхронных систем и трудности, связанные с их правильным моделированием, выдвигают на первый план проблему верификации. Базовым набором отношений при сетевом описании обобщенных процессов являются отношения предшествования, параллелизма и альтернативы, определяемые по топологической структуре сети-процесса. По существу, данный набор "сетевых" отношений может быть описан и в терминах другого формализма, а именно при помощи логики процессов. Однако в динамической логике [3,4], в логике процессов [5] логические средства привлекаются лишь для описания свойств систем и процессов. Описание же самих систем и процессов делается в рамках совершенно иного формализма. Этот контраст средств, используемых при описании программ, процессов и их свойств, становится еще более резким при рассмотрении

временной логики [6], предложенной для верификации и описания свойств параллельных программ и процессов. Синтаксис построения параллельных программ и процессов в последнем случае практически отсутствует. Функционирование задается некоторой совокупностью возможных последовательностей вычислений.

В настоящее время в ряде советских и зарубежных работ активно развивается новое направление - логическая спецификация параллельных программ и процессов, - которое может рассматриваться как создание математического аппарата для развития логических методов верификации. В рамках классической логики, дополненной временными операторами, оказывается возможным описание не только свойств параллельных процессов, но и точное их задание. Такая унификация средств позволяет по описанию процесса выводит его свойства, и таким образом решать проблему верификации.

В работе рассматривается логический подход к описанию конечных обобщенных параллельных процессов и их свойств. Формализуется базовый набор логических операций, устанавливается их связь с понятиями параллелизма, предшествования и альтернативы. Вводится понятие подпроцесса, являющегося базовым при выводе "частичных" свойств исходного процесса. Приводится список эквивалентных формульных преобразований при описании процессов и их свойств.

Для иллюстрации некоторых основных понятий будет использовано графическое изображение процессов в формализме сетей Петри [7].

Синтаксис

Пусть $\mathcal{A} = \{ a, b, c, \ldots \}$ - некоторый конечный алфавит символов для действий. Обозначим через $\overline{\mathcal{A}} = \{ \overline{a}, \overline{b}, \overline{c}, \ldots \}$ двойственный по отношению к $\mathcal{A}$ алфавит для обозначения "не реализовавшихся действий", т.е. событий, смысл которых заключается в том, что действие из $\mathcal{A}$ не произошло.

Специальным выделенным символом для обозначения пустого действия (дедлока, ошибки) служит δ.

Базовыми логическими операциями (связками) являются: & ("конъюнкция"), $\vee$ ("дизъюнкция"), $\neg$ ("не выполнение"), ∇ ("исключительное или" или "альтернатива") и / ("предшествование").

Интуитивно: семантика & и $\vee$ определяется традиционным образом; операция $\neg$ является модифицированным отрицанием: формула $\neg A$ обозначает тот факт, что процесс A не выполнился, т.е. ни од-

но действие процесса A не реализовалось; операция альтернативы ∇ является модификацией традиционного "исключительного или", формула $A \nabla B$ определяет процесс, в котором взаимно исключены выполнения процессов A и B, т.е. при выполнении процесса A процесс B не выполняется, и наоборот; операция предшествования / - единственная операция временной логики, которая служит для упорядочения действий в процессе (все остальные операции не накладывают никаких ограничений на порядок выполнения действий), формула A/B описывает процесс, в котором действия процесса B начинают выполняться только после того, как выполнение действий процесса A завершилось.

Правила построения формул:

1) a, $\bar{a}$ и δ, где $a \in \mathcal{A}$, $\bar{a} \in \bar{\mathcal{A}}$ являются формулами;

2) если A и B - формулы, то $A\&B$, $A \vee B$, $\neg A$, $A \nabla B$, A/B - формулы.

Семантика

Параллельный (обобщенный) процесс может быть охарактеризован при помощи множества цепочек всех его реализаций.

Введем вспомогательные обозначения.

Пусть $\hat{\mathcal{A}} = \mathcal{A} \cup \bar{\mathcal{A}}$ и n - количество символов в алфавите $\mathcal{A}$.

Обозначим через $\hat{\mathcal{A}}^n$ множество строк, удовлетворяющих требованию $\forall \sigma \in \hat{\mathcal{A}}^n$, $\forall a \in \mathcal{A}$ выполнено $(a \in \sigma$ or $\bar{a} \in \sigma)$, т.е. любая цепочка σ из $\hat{\mathcal{A}}^n$ имеет одну и ту же длину n и любой символ a из $\mathcal{A}$ входит в σ или явно (характеризуется тот факт, что действие a в процессе реализовалось), или под знаком отрицания $\bar{a}$ (обозначая, тот факт, что действие a в процессе не реализовалось, не выполнилось).

Каждой формуле A (в базисе действий $\mathcal{A}$) сопоставим некоторое множество цепочек $C_{\mathcal{A}}(A)$ (т.е. некоторый процесс) по следующим правилам.

I. Пусть $a, \bar{a}$ - элементарные действия. Тогда $C_{\mathcal{A}}(a) \overset{\text{def}}{=} (\hat{\mathcal{A}} \setminus \{\bar{a}\})^n$, $C_{\mathcal{A}}(a) \overset{\text{def}}{=} (\hat{\mathcal{A}} \setminus \{a\})^n$, т.е. формула a в базисе действий $\mathcal{A}$ описывает процесс, в каждой реализации которого действие a выполнилось, аналогично формула $\bar{a}$ описывает процесс, в котором действие a не реализовалось. Процесс δ соответствует пустой процесс $C_{\mathcal{A}}(\delta) \overset{\text{def}}{=} \emptyset$.

ПРИМЕР. Пусть $\alpha = \{a,b\}$, тогда формула a описывает процесс $C_\alpha(a) = \{ab, ba, a\bar{b}, \bar{b}a\}$, аналогично формула b задает процесс $C_\alpha(b) = \{ab, ba, b\bar{a}, \bar{a}b\}$.

2. Если же формула имеет вид A & B, то $C_\alpha(A\&B) \stackrel{def}{=} C_\alpha(A) \cap C_\alpha(B)$, т.е. в результирующий процесс, описанный формулой A & B, входят только те цепочки, на которых выполнены условия A и B одновременно, причем порядок выполнения действий из A в B произволен.

ПРИМЕР. Пусть $\alpha = \{a,b\}$, тогда формула a & b описывает следующий процесс $C_\alpha(a\&b) = C_\alpha(a) \cap C_\alpha(b) = \{ab, ba\}$, т.е. процесс, в котором действия a и b выполняются и выполняются в произвольном порядке.

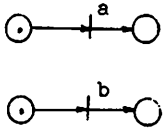

Рис.I

На рис.I при помощи сети Петри изображен параллельный процесс, описанный формулой a & b.

Как это следует из определения, в формуле A & B конъюнкция моделирует параллельное выполнение подпроцессов A и B.

3. Если формула имеет вид A ∨ B, то $C_\alpha(A \vee B) \stackrel{def}{=} C_\alpha(A) \cup C_\alpha(B)$, т.е. в процесс, описанный формулой A ∨ B, входят как цепочки, на которых выполнено условие A, так и те цепочки, для которых истинно условие B.

4. Для определения семантики формулы A/B необходимо ввести ряд вспомогательных понятий.

Обозначим через "·" операцию конкатенации двух строк. Расширим операцию конкатенации на множества строк следующим образом:

$$A \cdot B \stackrel{def}{=} \{ x \cdot y \mid x \in A, y \in B \}.$$

Обозначим через "‖" операцию параллельной композиции двух строк, определяемую следующим набором правил:

$$a \| \lambda = \lambda \| a \stackrel{def}{=} \{a\},$$

$$a \cdot x \| b \cdot y \stackrel{def}{=} \{a\} (x \| b \cdot y) \cup \{b\} \cdot (a \cdot x \| y),$$

где a, b – символы, λ – символ пустой строки, x, y – произвольные строки.

Естественным образом операция параллельной композиции "‖" расширяется на произвольные множества строк:

$$A \| B \overset{def}{=} \{x \| y \mid x \in A,\ y \in B\}.$$

Пусть A - формула, обозначим через $\alpha(A)$ множество символов действий, входящих в A . Заметим, что $\alpha(\bar{a}) \overset{def}{=} \{a\}$ для любого символа нереализованного действия. Соответственно $\bar{\alpha}(A)$ обозначает двойственный по отношению к $\alpha(A)$ алфавит

$$\hat{\alpha}(A) = \alpha(A) \cup \bar{\alpha}(A).$$

Итак, если формула имеет вид A/B (причем $\alpha(A) \cap \alpha(B) = \emptyset$), то $C_\alpha(A/B) \overset{def}{=} (C_{\alpha(A)} \cdot C_{\alpha(B)}(B)) \| (\hat{\alpha} \setminus (\hat{\alpha}(A) \cup \hat{\alpha}(B)))$, т.е. формула A/B описывает процесс, в котором выполнение любого действия из A предшествует выполнению любого действия из B , в противном случае $C_\alpha(A/B) = \emptyset$.

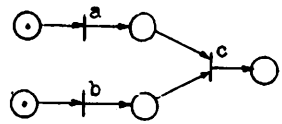

Рис. 2

ПРИМЕР. Пусть $\alpha = \{a,b,c\}$, тогда формула $((a\&b)/c)$ описывает процесс $C_\alpha((a\&b)/c) = \{abc, bac\}$, изображенный сетью Петри на рис.2.

5. Если формула имеет вид $\neg A$, то $C_\alpha(\neg A) \overset{def}{=} (\hat{\alpha} \setminus \alpha(A))^n$, т.е. формула $\neg A$ описывает процесс, в котором ни одно действие из A не выполнилось. В целях эквивалентных преобразований формул часто будет использоваться другое определение операции $\neg$, заданное следующим набором правил:

$$\neg a = \bar{a},$$

$$\neg \bar{a} = a,$$

$$\neg(A \circ B) = \neg A \,\&\, \neg B,$$

где $\circ = \{\&, \vee, /\}$. Эквивалентность двух приведенных определений для операции $\neg$ очевидна.

ПРИМЕР. $\neg((a \vee b)/c) = \neg(a \vee b) \,\&\, \neg c = \bar{a} \,\&\, \bar{b} \,\&\, \bar{c}$.

6. Семантика формулы $A \triangledown B$ задается через введенные выше операции следующим образом: $A \triangledown B \overset{def}{=} A \,\&\, \neg B \vee \neg A \,\&\, B$, т.е. формула $A \triangledown B$ описывает процесс, в котором выполнения процессов A и B взаимно исключены; если выполнился процесс A , то процесс B не выполняется, и наоборот. Из определения следует, что если $\alpha(A) \cap \alpha(B) \neq \emptyset$, то $C_\alpha(A \triangledown B) = \emptyset$.

ЗАМЕЧАНИЕ. Как легко можно убедиться, порядок "невыполнения" действий не влияет на функционирование процесса. В этом смысле цепочки $\sigma_1 = a\bar{b}\bar{c}$ и $\sigma_2 = a\bar{c}\bar{b}$ различаются только условно, формально. В дальнейшем, для облегчения манипуляций со строками, будем считать, что все строки приведены к каноническому виду, т.е. все символы нереализованных действий выписаны в конце строки в соответствии с порядком, заданным на алфавите $\bar{\mathcal{A}}$. Например, пусть алфавит $\bar{\mathcal{A}}$ упорядочен следующим образом: $\bar{\mathcal{A}} = \{\bar{a}, \bar{b}, \bar{c}\}$, тогда каноническим видом строк $\sigma_1 = \bar{c}a\bar{b}$, $\sigma_2 = \bar{c}\bar{b}a$, $\sigma_3 = \bar{b}\bar{c}a$, $\sigma_4 = \bar{b}a\bar{c}$ и т.д. будет строка $\sigma_0 = a\bar{b}\bar{c}$.

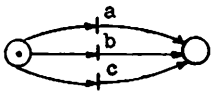

Рис. 3

Заметим, что из определений операций $\neg$ и ∇ вытекает следующее правило: $\neg(A \nabla B) = \neg A \,\&\, \neg B$.

ПРИМЕР. Пусть $\mathcal{A} = \{a, b, c\}$, тогда формула $((a \nabla b) \nabla c)$ описывает следующий процесс $C_{\mathcal{A}}((a \nabla b) \nabla c) = \{a\bar{b}\bar{c}, b\bar{a}\bar{c}, c\bar{a}\bar{b}\}$, изображенный сетью Петри на рис.3.

Связь алгебры регулярных сетей и алгебры процессов

В работах [1,2] и др. предложен и развит сетевой подход к описанию семантики параллельных и обобщенных процессов. В свою очередь, сети Петри могут быть описаны специальной алгеброй [7]. В чем принципиальная разница алгебры сетей и предлагаемой в данной работе алгебры (логики) процессов? В алгебре сетей конструирование происходит на уровне структуры, при этом часто возникает неадекватность структуры сети и описываемого данной сетью процесса. При сетевом подходе такого рода ситуации исключается введение требований K-, L-, M-плотностей [1,2] на структуру сети.

В предлагаемой алгебре процессов конструирование происходит на ином уровне, а именно на уровне функционирования.

ПРИМЕР. Формула $1((a \square b), (a, b))$ в алгебре сетей описывает сеть на рис.4. Операция исключения "$\square$" интуитивно аналогична операции альтернативы "∇", операция наложения "," в некотором смысле близка к операции конъюнкции "&". Однако попытка аналогичного описания в алгебре процессов $(a \nabla b) \,\&\, (a \,\&\, b)$ приводит к порождению пустого процесса. Это происходит по той причине, что

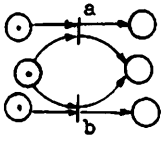

Рис. 4

сеть Петри на рис.4 не М-плотна, т.е. структура сети не адекватна описываемому данной сетью процессу.

Рассмотрим класс конечных сетей-процессов $\tilde{\mathcal{N}}$, построенных при помощи операций "$\square$" (исключение), ";" (присоединение), "," (наложение) и засылки стандартной единичной начальной разметки в головные места сети (не будем изображать эту операцию в формульной записи). Выделим в данном классе подкласс $\tilde{\mathcal{N}}_0$ М-плотных сетей процессов. Каждой формуле N сети $\tilde{N}$ из подкласса $\tilde{\mathcal{N}}_0$ сопоставим формулу алгебры процессов $\varphi(N)$ по следующим правилам:

$\varphi(A;B) = \varphi(A)/\varphi(B)$,

$\varphi(A \square B) = \varphi(A) \triangledown \varphi(B)$,

$\varphi(A,B) = \varphi(A) \& \varphi(B)$,

$\varphi(a) = a$, где a - символ элементарной сети.

Имеет место следующее

УТВЕРЖДЕНИЕ. Рассмотрим формулу N сети $\tilde{N} \in \tilde{\mathcal{N}}_0$, тогда формула алгебры процессов $\varphi(N)$ и сеть $\tilde{N}$ описывают один и тот же обобщенный процесс.

Это утверждение является следствием из условия М-плотности рассмотренных сетей-процессов, поскольку в М-плотных сетях синтаксическое (структурное) описание адекватно порождаемому данной сетью процессу.

Выводимость свойств процессов

Формула во введенной выше алгебре процессов может служить как для описания процесса, так и для описания свойств процесса. Аналогичная ситуация возникает и в традиционных логиках. Предикат P, определяемый следующим образом:

$$P(x) = \begin{cases} \text{истина, если } x \text{ четно;} \\ \text{ложь - в противном случае,} \end{cases}$$

с одной стороны, задает множество четных чисел, с другой стороны, описывает свойство "быть четным".

Соответственно описание процесса в каком-то смысле является самым полным описанием свойства, имеющего место для данного процесса.

Для обобщенных параллельных процессов можно выделить две основные группы свойств: глобальные (имеющие место при любой реализации процесса) и частичные (справедливые на подмножестве реали - заций). Второй подкласс свойств возникает из-за наличия операции альтернативы, а следовательно, возможности альтернативного выполнения подпроцессов. Интуитивно глобальные свойства соответствуют понятию истинности на модели, а частичные свойства связаны с понятием выполнимости.

Рассмотрим процесс, заданный формулой $(a \triangledown b)\ \&\ (c \triangledown d)$ и изображенный сетью Петри на рис.5. Свойство $(a \triangledown b)$ (действия a и b альтернативны) является глобальным, т.е. при любой реализации процесса либо выполняется действие a и b не не выполняется, либо наоборот; тогда как свойство $(a\ \&\ b)$ (действия a и b параллельны) является частичным, т.е. существует подпроцесс, в котором действия a и b выполняются в любом порядке, т.е. справедлива формула $(a\ \&\ b)$.

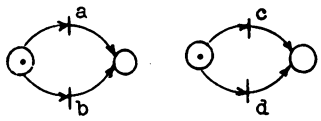

Рис. 5

В данной работе будут рассмотрены логические средства вывода как частичных, так и глобальных свойств процессов.

Эквивалентные преобразования формул

Две формулы Φ_1 и Φ_2 в базисе $\mathcal{A}$ называются эквивалентными $(\Phi_1 = \Phi_2)$, если $C_{\mathcal{A}}(\Phi_1) = C_{\mathcal{A}}(\Phi_2)$.

Ниже приведен список эквивалентных преобразований формул, связанных с коммутативностью, ассоциативностью, дистрибутивностью и некоторыми свойствами операций.

1. Ассоциативность

1.1. $A\ \&\ (B\ \&\ C) = (A\ \&\ B)\ \&\ C$

1.2. $A \vee (B \vee C) = (A \vee B) \vee C$

1.3. $A \triangledown (B \triangledown C) = (A \triangledown B) \triangledown C$

1.4. $A/(B/C) = (A/B)/C$.

2. Коммутативность

2.1. $A\ \&\ B = B\ \&\ A$

2.2. $A \vee B = B \vee A$

2.3. $A \triangledown B = B \triangledown A$

3. Дистрибутивность

3.1. $(A\ \&\ B)/C = (A/C)\ \&\ (B/C)$

3.2. $A/(B\&C) = (A/B)\&(A/C)$

3.3. $(A\vee B)/C = (A/C)\vee(B/C)$, если $\mathcal{A}(A) = \mathcal{A}(B)$

3.4. $A/(B\vee C) = (A/B)\vee(A/C)$, если $\mathcal{A}(B) = \mathcal{A}(C)$

3.5. $A\nabla(B\&C) = (A\nabla B)\&(A\nabla C)$.

4. Аксиомы для ∇ и $\neg$

4.1. $A\nabla B = A\&\neg B\vee\neg A\&B$

4.2. $\neg(A\&B) = \neg A\&\neg B$

4.3. $\neg(A\vee B) = \neg A\&\neg B$

4.4. $\neg(A/B) = \neg A\&\neg B$

4.5. $\neg a = \bar{a}$

4.6. $\neg\bar{a} = \bar{a}$.

5. Другие

5.1. $\bar{a}/A = \bar{a}\&A$

5.2. $A/\bar{a} = \bar{a}\&A$

5.3. $A\&(A/B) = A/B$

5.4. $A/B/C = (A/B)\&(B/C)$

6. Дополнительные

6.1. $A\&A = A$

6.2. $A\vee A = A$

6.3. $a\&\bar{a} = \delta$

6.4. $A/B/A = \delta$

6.5. $A\nabla A = \delta$

6.6. $\delta\&A = \delta$

6.7. $\delta/A = \delta$

6.8. $A/\delta = \delta$

6.9. $\delta\nabla A = \delta$

6.10. $\delta\nabla A = A$

6.11. $A = A\&a\vee A\&\bar{a}$, где $a\in\mathcal{A}\setminus\mathcal{A}(A)$.

Выводимость частичных свойств процесса

Введем ряд вспомогательных понятий и обозначений.

Две цепочки σ и σ' называются перестановочными ($\sigma\approx\sigma'$), если σ' получается из σ при помощи перестановки отдельных символов.

Формула Φ' называется подпроцессом формулы Φ, если выполняются требования:

1) $C(\Phi)\vdash_{\hat{\mathcal{A}}(\Phi')} \geq C(\Phi')$,

2) $\forall \sigma \in C(\Phi)\,\forall \sigma' \in C(\Phi'): (\sigma \approx \sigma' \Rightarrow \sigma \in C(\Phi'))$,

где $C(\Phi)\restriction_{\hat{\mathcal{A}}(\Phi')}$ – проекция цепочек из $C(\Phi)$ на множество символов $\hat{\mathcal{A}}(\Phi')$.

Благодаря требованию 2, в подпроцессе Φ' элементы связаны теми же отношениями (предшествования, альтернативы или параллелизма), что и в исходном процессе Φ.

В терминах сетей данное выше определение подпроцесса совпадает с понятием согласованной компоненты [7].

Сеть $N' = (P', T', F')$ является согласованной компонентой сети $N = (P, T, F)$, если $P \supseteq P'$, $T \supseteq T'$ и $\forall x, y \in P' \cup T'$: $F'^{+}(x,y) = F^{+}(x,y)$, где F^{+} – транзитивное замыкание F. Таким образом, элементы согласованной компоненты связаны тем же набором причинно-следственных отношений, что и в исходной сети.

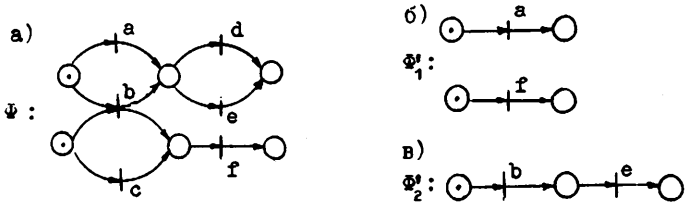

Рис. 6

Так, для процесса $\Phi = ((a \triangledown b)/(d \triangledown e)) \& (c \triangledown b)/f)$, изображенного сетью Петри на рис.6,а, $\Phi'_1 = a \& f$ и $\Phi'_2 = b/e$ – подпроцессы. Их сетевые изображения на рис.6,б,в соответственно являются согласованными компонентами исходной сети.

Свойство Φ' выполнимо для процесса Φ ($\Phi \Vdash \Phi'$) тогда и только тогда, когда Φ' является подпроцессом Φ.

ПРИМЕР. $((a \triangledown b)/(d \triangledown e)) \& ((b \triangledown c)/f) \Vdash a \& f$.

Для логического вывода частичных свойств имеют место следующие правила вывода:

I. $A \triangledown B \Vdash A$, если $\mathcal{A}(A) \cap \mathcal{A}(B) = \emptyset$

$\tilde{I}$. $A \triangledown B \Vdash A' \triangledown B'$, где $A \Vdash A'$ $B \Vdash B'$, если $\mathcal{A}(A) \cap \mathcal{A}(B) = \emptyset$

П. $A/B \Vdash A$ и $A/B \Vdash B$, если $\mathcal{A}(A) \cap \mathcal{A}(B) = \emptyset$

$\tilde{\text{П}}$. $A/B \Vdash A'/B'$, где $A \Vdash A'$ и $B \Vdash B'$, если $\mathcal{A}(A) \cap \mathcal{A}(B) = \emptyset$

Ш. $A \& B \Vdash A$, если $\mathcal{A}(A) \cap \mathcal{A}(B) = \emptyset$

$\tilde{\text{Ш}}$. $A \& B \Vdash A' \& B'$, где $A \Vdash A'$, $B \Vdash B'$, и если $\mathcal{A}(A) \cap \mathcal{A}(B) = \emptyset$.

Состоятельность этих правил очевидна. В силу контекстного ограничения $\mathcal{A}(A) \cap \mathcal{A}(B) = \emptyset$ во всех трех случаях утверждение о том, что в правой части вывода стоит формула-подпроцесс, непосредственно следует из определения подпроцесса.

Для введения еще одного содержательного правила вывода, используется следующий алгоритм приведения произвольной формулы к специальному каноническому виду.

На первом шаге применением эквивалентностей из групп I-6 (см. с. 104) любая формула Φ приводится к виду: $\Phi = \Phi_1 \vee \Phi_2 \vee \ldots \vee \Phi_k$, где Φ_i содержит только операции & и / над элементарными действиями из $\hat{\mathcal{A}}$.

На втором шаге проводится нормализация формул Φ_i применением правил 3.I, 3.2, 5.I-5.4, 6.I-6.I0. Нормализация направлена на "поглощение" в формуле Φ_i "избыточной" информации и связана в основном с применением правила 5.3, т.е. мы добиваемся, чтобы из формулы, например $(a \& b) \& (a/b)$, осталась ее "содержательная" часть соответственно (a/b).

Интуитивно набор нормализованных конъюнктов $\{\Phi_i\}_{i=1}^{k}$ является совокупностью возможных (максимальных) параллельных подпроцессов.

Имеет место следующее правило вывода:

IУ. $\Phi \Vdash \Phi_i$, где Φ_i - нормализованный конъюнкт.

Заметим, что к нормализованным конъюнктам типа Φ_i правило вывода Ш применимо без контекстных ограничений.

ПРИМЕР.

$\Phi = ((a \triangledown b) \& c)/d) \& (c/e)$

$\Phi = \Phi_1 \vee \Phi_2 = (((a \& \bar{b} \& c)/d) \& (c/e) \vee (((\bar{a} \& b \& c)/d) \& (c/e)$

$\Phi \Vdash \Phi_1 = ((a \& \bar{b})/d) \& (c/d) \& (c/e) =$

$= ((a \& \bar{b})/d) \& (c/(d \& e)) \Vdash c/(d \& e) \Vdash d \& e.$

Поскольку в статье рассмотрена логика конечных процессов, то разрешимость и конечная аксиоматизируемость такой логики очевидна.

Однако на содержательном уровне для вывода свойств предшествования (A/B) и параллелизма (A&B) достаточно набора аксиом I–6 и приведенных выше правил вывода I–IУ.

Выводимость глобальных свойств

Свойство Φ' и с т и н н о для процесса Φ ($\Phi \vdash \Phi'$) тогда и только тогда, когда $C_{\mathcal{A}}(\Phi)\restriction_{\hat{\mathcal{A}}(\Phi')} = C_{\mathcal{A}}(\Phi')$, где $C_{\mathcal{A}}(\Phi)\restriction_{\hat{\mathcal{A}}(\Phi')}$ – проекция цепочек из $C_{\mathcal{A}}(\Phi)$ на множество символов $\hat{\mathcal{A}}(\Phi')$.

Для логического вывода глобальных свойств имеют место следующие правила вывода:

I. $(A \triangledown B) \vdash A' \vee \overline{\Pi}A'$, где $A \vdash A'$ и если $\mathcal{A}(A) \cap \mathcal{A}(B) = \emptyset$.

$\tilde{I}$. $(A \triangledown B) \vdash A' \triangledown B'$, где $A \vdash A'$, $B \vdash B'$, если $\mathcal{A}(A) \cap \mathcal{A}(B) = \emptyset$.

II. $(A/B) \vdash A$ и $A/B \vdash B$, если $\mathcal{A}(A) \cap \mathcal{A}(B) = \emptyset$.

$\tilde{II}$. $(A/B) \vdash A'/B'$, где $A \vdash A'$, $B \vdash B'$, если $\mathcal{A}(A) \cap \mathcal{A}(B) = \emptyset$.

III. $(A \& B) \vdash A$, если

1) $\mathcal{A}(A) \cap \mathcal{A}(B) = \emptyset$, или

2) $A \& B$ не содержит знака $\triangledown$ и является нормализованным конъюнктом.

$\tilde{III}$. $(A \& B) \vdash A' \& B'$, где $A \vdash A'$, $B \vdash B'$ и если $\mathcal{A}(A) \cap \mathcal{A}(B) = \emptyset$.

IУ. Пусть $\Phi = \bigvee_{i=1}^{k} \Phi_i$ – дизъюнктивная нормальная форма.

Если для любого i $(1 \le i \le k)$ имеет место $\Phi_i \vdash A$, то $\Phi \vdash A$.

ПРИМЕРЫ.

1) $\Phi = (a/b \& c)/d) \triangledown e$,

$a/(b \& c)/d \vdash a/(b \& c) \vdash b \& c$,

$\Phi \vdash (b \& c) \vee \overline{\Pi}(b \& c) = b \& c \vee \bar{b} \& \bar{c}$.

2) $\Phi = (a \triangledown b) \& (c \triangledown d)$,

$a \triangledown b \vdash a \vee \overline{\Pi}a = a \vee \bar{a}$,

$c \triangledown d \vdash c \vee \overline{\Pi}c = c \vee \bar{c}$,

$\Phi \vdash (a \vee \bar{a}) \& (c \vee \bar{c}) = a \& c \vee \bar{a} \& c \vee a \& \bar{c} \vee \bar{a} \& \bar{c}$.

Заключение

В работе предложен унифицированный логический подход к описанию процессов и их свойств. Формализован набор базовых операций и понятий, связанных с отношениями параллелизма, предшествования и альтернативы. Введены и обоснованы правила вывода частичных и глобальных свойств процесса. Рассмотрен алгоритм приведения к специальной канонической форме, где в качестве конъюнкта выбирается максимальный параллельный подпроцесс. Это позволяет сделать логический вывод свойств более эффективным, поскольку вывод происходит не на уровне цепочек слов, характеризующих исходный процесс (как это было традиционно), а на следующем, более высоком уровне параллельных подпроцессов.

Литература

I. PETRI C.A. Non-sequential processes: ISF-Report-77.05. - St.Augustin: esellschaft für Mathematik und Datenverarbeitung, 1971.- 31 P.

2. KOTOV V.E., CHERKASOVA L.A. On structural properties of generalized processes// Lecture Notes in Computer Science.-Berlin a.o.,1984.-Vol.188.-P.288-306.

3. HAREL D. First-order dynamic logic// Lecture Notes in Computer Science.-Berlin a.o.,1979.-Vol.68.-133 P.

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

5. PRATT V.R. Process logic// Proc.ACM Symp.on Principles of Programming Languages.1979.- P.93-100.

6. MANNA Z., PNUELI A. Verification of concurrent programs: temporal proof principles// Lecture Notes in Computer Science. - Berlin a o.,1981.- Vol.131. - P.200-252.

7. КОТОВ В.Е. Сети Петри. -М.: Наука, 1984. - 158 с.

Поступила в ред.-изд. отд.
22 апреля 1986 года