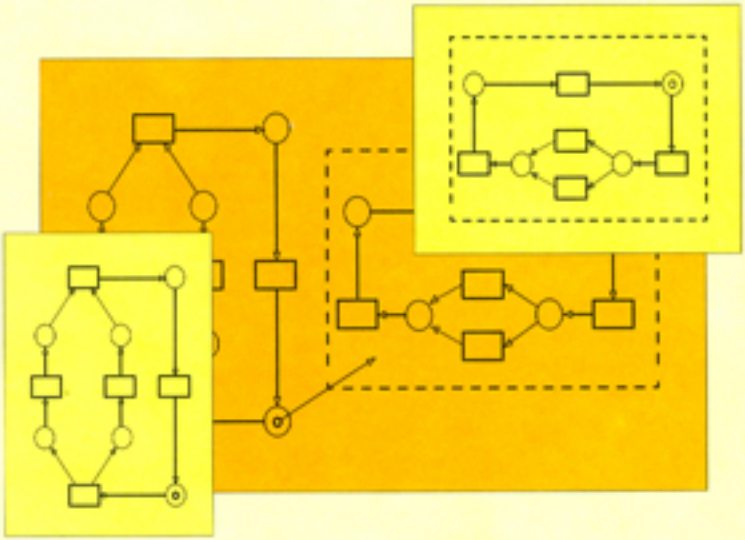
И.А. ЛОМАЗОВА

Вложенные сети Петри:

моделирование и анализ распределенных систем с объектной структурой



НАУЧНЫЙ МИР

И. А. Ломазова

Вложенные сети Петри:
моделирование и анализ распределенных систем с объектной структурой

Москва
Научный мир
2004

УДК 681.3
ББК 22.18
Л74

Л74 *Ломазова И. А.* Вложенные сети Петри: моделирование и анализ распределенных систем с объектной структурой. — М.: Научный мир, 2004. — 208 с.
ISBN 5-89176-247-1

В книге представлено описание формализма вложенных сетей Петри, предназначенного для моделирования и анализа поведения распределенных систем со сложной объектной структурой. Вложенные сети Петри представляют собой расширение стандартного формализма сетей Петри, в котором фишки, представляющие локальные ресурсы в позициях сети, сами могут быть сложными объектами с сетевой структурой. Дается описание формальной семантики таких сетей. Приводятся алгоритмы верификации некоторых поведенческих свойств.

Книга предназначена для научных работников, преподавателей, аспирантов и студентов, интересующихся формальными моделями параллельных и распределенных систем.

Илл. 38. Библ. 95 назв.

РФФИ *Публикуется при финансовой поддержке Российского фонда фундаментальных исследований (проект 03-01-14080)*

Lomazova I. A. Nested Petri nets: Modeling and analysis of distributed systems with object structure. — Moscow: Scientific World, 2004. — 208 p.

The book represents the nested Petri nets formalism intended for modeling and analysis of distributed systems with complex object structure. Nested Petri nets extend the standard Petri net formalism by allowing tokens residing in places of a net to be objects with a net structure. The book presents a formal semantics of such nets and some algorithms for verification of their behavioral properties.

The book is designed for scientists, professors and students interested in formal models of concurrent and distributed systems.

Pict. 38. Bibl. 95 titles.

ISBN 5-89176-247-1

Оглавление

Предисловие

Разработка средств моделирования и анализа поведения распределенных систем является актуальной задачей, особенно в связи с постоянным возрастанием сложности проектируемых и исследуемых систем.

Одним из наиболее распространенных современных формализмов для моделирования и анализа параллельных распределенных систем являются сети Петри. В настоящее время ведется большое число как теоретических исследований в этой области, так и реализаций практических инструментариев для разработки распределенных алгоритмов, основанных на сетях Петри. Разрабатывается международный стандарт по сетям Петри. Ежегодно проводится большая международная конференция "Теория и приложения сетей Петри". Практически все конференции по теории и практике параллелизма включают секции по сетям Петри. В России во многих университетах и академических институтах также ведутся исследования в этой области.

Вместе с тем специалистами ощущается необходимость развития формализма с целью более адекватного и удобного представления систем со сложной структурой. Современные системы часто являются мультиагентными и имеют иерархическую, многоуровневую структуру. В связи с этим в последнее время ведутся исследования по расширению формализма сетей Петри за счет идей объектно-ориентированного подхода с целью получения моделей,

явно отражающих иерархическую и мультиагентную структуру системы (см., например, обзоры [95, 20, 62]).

В настоящее время также разрабатывается целый ряд программных систем, основанных на сетях Петри и дополненных конструкциями объектно-ориентированного программирования [18]. При этом добавление тех или иных языковых конструкций часто производится исходя из требований приложений без определения их формальной семантики. Таким образом, существует необходимость теоретического обоснования интеграции сетей Петри и объектно-ориентированного подхода.

В настоящей монографии излагается один из подходов к решению этой задачи. Здесь представлен новый формализм для моделирования и анализа распределенных систем с объектной структурой — вложенные сети Петри. Вложенные сети Петри представляют собой расширение стандартного формализма сетей Петри, в котором фишки, представляющие локальные ресурсы в позициях сети, сами могут быть сложными объектами с сетевой структурой. Имеются механизмы синхронизации объекта с системной сетью и двух объектов, находящихся в одной позиции системной сети.

Название "вложенные сети Петри" выбрано потому, что элементы сетей в них сами являются сетями, подобно тому, как в системе вложенных множеств элементами некоторого множества могут быть множества.

В книге рассматриваются двухуровневые, многоуровневые и рекурсивные вложенные сети Петри. Дается описание формальной семантики таких сетей. Приводятся алгоритмы решения проблем останова, покрытия и некоторых других проблем для многоуровневых вложенных сетей. Для рекурсивных сетей с автономными элементами доказана разрешимость проблемы останова. Доказано, что вложенные сети более выразительны, чем обыкновенные сети Петри, но сохраняют многие достоинства базовой модели. В частности, более слабая по сравнению с универсальными вычислителями выразительность вложенных сетей Петри де-

лает их перспективными с точки зрения верификации. Имеются модельные примеры, которые иллюстрируют возможность естественного отображения структуры системы и наглядность модели.

Представленная работа выполнена автором в Исследовательском центре искусственного интеллекта Института программных систем РАН.

Книга может быть использована в качестве основы семестрового спецкурса для студентов старших курсов, магистрантов и аспирантов, специализирующихся в областях теоретической информатики, программирования и вычислительной техники. По ее материалам автором в течение ряда лет читается спецкурс для магистрантов кафедры Теоретической информатики Ярославского государственного университета им. П.Г. Демидова.

Глава 1

Предварительные сведения

1.1 Мультимножества

Понятие мультимножества есть обобщение понятия множества. В мультимножестве допускается вхождение нескольких экземпляров одного и того же элемента.

Определение 1.1. Пусть X – некоторое множество. Мультимножеством M над X называется функция $M : X \to \mathbb{N}$, где $\mathbb{N} = \{0, 1, 2, \dots\}$ – множество натуральных чисел.

Полагаем $x \in M$, если $M(x) > 0$ и $x \notin M$, если $M(x) = 0$. Для $x \in X$ значение $M(x)$ называют кратностью x в M.

В том случае, когда $\forall x \in X : M(x) \leq 1$, M является обычным множеством. Мощность мультимножества определяется как $|M| = \Sigma_{x \in X} M(x)$. Мультимножество конечно, если $M(x) = 0$ для всех $x \in X$ за исключением, может быть, конечного их числа. В дальнейшем будем рассматривать только конечные мультимножества. Множество всех конечных мультимножеств над множеством X будем обозначать через $\mathcal{M}(X)$.

Операцию вычитания до нуля для натуральных чисел определим следующим образом: $\forall n, m \in \mathbb{N} : n \ominus m =_{\text{def}} \max(0, n - m)$.

Операции и отношения теории множеств естественно расширяются на конечные мультимножества. Пусть $M_1, M_2 \in \mathcal{M}(X)$ и $x \in X$. Полагаем

$$(M_1 + M_2)(x) = M_1(x) + M_2(x);$$
$$(M_1 - M_2)(x) = M_1(x) \ominus M_2(x);$$
$$(M_1 \cup M_2)(x) = \max(M_1(x), M_2(x));$$
$$(M_1 \cap M_2)(x) = \min(M_1(x), M_2(x));$$
$$M_1 \subseteq M_2 \Leftrightarrow \forall x \in X : M_1(x) \leq M_2(x).$$

Вместо $M + \{x\} - \{y\}$ обычно будем писать $M + x - y$. Символ $\emptyset$ обозначает пустое мультимножество, т.е. такое $M \in \mathcal{M}(X)$, что $M(x) = 0$ для всех x.

1.2 Квазиупорядоченные множества

Напомним ([9]), что бинарное отношение R называется отношением *частичного порядка*, если оно рефлексивно, транзитивно и антисимметрично. Антисимметричность означает, что

$$(xRy) \wedge (yRx) \rightarrow (x = y).$$

Если отношение только рефлексивно и транзитивно, то оно называется отношением *квазипорядка* или *квазиупорядочиванием*.

Вместо xRy обычно пишут $x \leq_R y$ или просто $x \leq y$. Будем также говорить, что множество X, являющееся носителем отношения R, частично упорядочено (или квазиупорядочено), не указывая явно отношения R, если из контекста ясно, о каком упорядочении идет речь. Если $x \leq y$ и $x \neq y$, то пишут $x < y$ и говорят, что x строго меньше y. Очевидно, что строгий порядок есть транзитивное и иррефлексивное отношение.

Для квазипорядка $x \leq_R y$ *обратным* называют квазипорядок $x \geq_R y$ такой, что $x \geq_R y \Leftrightarrow y \leq_R x$. Аналогично, пишем $x >_R y \Leftrightarrow y <_R x$. Квазипорядок R_1 на множестве X называется *расширением* квазипорядка R_2 на X, если для любых $x_1, x_2 \in X$

из $x_1 \leq_{R_2} x_2$ следует $x_1 \leq_{R_1} x_2$, соответственно, R_2 называется *сужением* R_1.

Частично упорядоченное множество, в котором любые два элемента сравнимы, называется *линейно упорядоченным* или *цепью*.

Подмножество Y квазиупорядоченного множества $\langle X, \leq \rangle$ называется *антицепью*, если элементы Y попарно несравнимы, т.е. для любой пары элементов $x \neq y$ из Y не выполняется ни $x \leq y$, ни $y \leq x$.

Элемент a квазиупорядоченного множества X называется *минимальным элементом* этого множества, если в X нет ни одного элемента x, удовлетворяющего условию $x < a$.

1.2.1 Правильные квазиупорядочения

Квазипорядок R на множестве X называется *фундированным* или квазипорядком с условием минимальности (well-founded), если всякое непустое подмножество множества X имеет минимальный (в этом подмножестве) элемент. При наличии аксиомы выбора это эквивалентно тому, что в X нет бесконечных строго убывающих цепей вида $x_0 > x_1 > x_2 > \ldots$ [10].

Квазипорядок $\leq$ на множестве X называется *правильным* (well-quasi-ordering), если для любой бесконечной последовательности $x_0, x_1, x_2, \ldots$ элементов из X существуют индексы $i < j$ такие, что $x_i \leq x_j$ [6].

Непосредственно из определений следует, что:

- Всякое расширение правильного квазипорядка является правильным.
- Всякое сужение фундированного квазипорядка является фундированным.

Далее приводятся некоторые другие важные свойства правильных квазипорядков.

Утверждение 1.1. *Квазипорядок $\leq$ на множестве X является правильным в том и только том случае, когда:*

1) *X не содержит бесконечных строго убывающих цепей $x_0 > x_1 > x_2 > \ldots$;*

2) *всякая содержащаяся в X антицепь конечна.*

Доказательство. Легко заметить, что для правильного квазипорядка выполняется условие минимальности. Действительно, если нарушается условие минимальности и существует бесконечная строго убывающая последовательность $x_0 > x_1 > x_2 > \ldots$, то для этой последовательности для любой пары индексов $i < j$ не выполняется $x_i \leq x_j$.

Аналогично, если существует бесконечная антицепь Y, то по аксиоме выбора из элементов Y можно построить бесконечную последовательность, не удовлетворяющую условию правильности.

Докажем, что если условия 1, 2 выполняются, то $\leq$ — правильный квазипорядок. Предположим противное. Пусть $x_0, x_1, x_2, \ldots$ — некоторая бесконечная последовательность элементов из X, для которой условие правильности не выполняется, но выполняются условия 1 и 2. Итак, для любых $j > i$ не выполняется $x_i \leq x_j$. Тогда в силу конечности антицепей из этой последовательности можно выбрать подпоследовательность

$$x_{i_1} > x_{j_1}, x_{i_2} > x_{j_2}, \ldots, \tag{1.1}$$

где $i_1 < j_1 < i_2 < j_2 < i_3 < \ldots$.

Рассмотрим теперь последовательность $x_{i_1}, x_{i_2}, x_{i_3}, \ldots$. Повторяя предыдущие рассуждения, из нее можно выбрать подпоследовательность

$$x_{i'_1} > x_{j'_1}, x_{i'_2} > x_{j'_2}, \ldots, \tag{1.2}$$

где $i'_1 < j'_1 < i'_2 < j'_2 < i'_3 < \ldots$. Тогда, объединяя 1.1 и 1.2, имеем

$$x_{i'_1} > x_{j'_1} > x_{k_1}, x_{i'_2} > x_{j'_2} > x_{k_2}, \ldots,$$

где $i'_1 < j'_1 < k_1 < i'_2 < j'_2 < k_2 < i'_3 < \dots$.

Применяя аналогичные рассуждения, из данной последовательности $x_0, x_1, x_2, \dots$ можно построить сколь угодно длинную строго убывающую подпоследовательность, что противоречит условию 2. □

Следствие 1.1. *1) Всякий правильный квазипорядок является фундированным.*

2) Для линейного порядка условия правильности и фундируемости эквивалентны.

Утверждение 1.2. *Пусть на множестве X задан правильный квазипорядок $\leq$. Тогда всякая бесконечная последовательность $x_0, x_1, x_2, \dots$ элементов из X содержит бесконечную возрастающую подпоследовательность $x_{i_0} \leq x_{i_1} \leq x_{i_2} \dots$, где $i_0 < i_1 < i_2 \dots$.*

Доказательство. Рассмотрим бесконечную последовательность x_0, x_1, $x_2, \dots$ и множество $M = \{i \in \mathbb{N} \mid \forall j > i : x_i \not\leq x_j\}$. Множество M не может быть бесконечным, так как в этом случае из его элементов можно было бы выбрать бесконечную подпоследовательность, для которой не выполняется условие правильности. Тогда множество M ограничено, и, следовательно, любая подпоследовательность, начинающаяся с индекса $i \notin M$, является бесконечной возрастающей подпоследовательностью. □

Определение 1.2. Пусть на множестве X задан правильный квазипорядок $\leq$. Подмножество $C \subseteq X$ называется *верхним конусом*, если для любых $x \in C$ и $y \in X$ из $y \geq x$ следует, что $y \in C$.

Каждый элемент $x \in X$ порождает верхний конус $\uparrow x =_{\text{def}} \{y \mid y \geq x\}$. *Базисом* верхнего конуса C называется множество C^b такое, что $C = \cup_{x \in C^b} \uparrow x$. Следующее свойство было впервые установлено Хигманом [46].

Утверждение 1.3. *Если $\leq$ — правильный квазипорядок на множестве X, то всякий верхний конус C имеет конечный базис.*

Доказательство. Поскольку $\leq$ — фундированный порядок, множество минимальных элементов конуса образует его базис. Этот базис может содержать только конечное число элементов, так как в противном случае из этих минимальных элементов можно было бы построить бесконечную последовательность попарно несравнимых элементов, что противоречит условию правильности. □

В дальнейшем нам понадобится следующее свойство верхних конусов.

Утверждение 1.4. *Если $\leq$ — правильный квазипорядок на множестве X, то всякая бесконечно возрастающая (по отношению вложения множеств) последовательность $C_0 \subseteq C_1 \subseteq C_2 \subseteq \ldots$ верхних конусов стабилизируется, т.е. найдется такое $k \in \mathbb{N}$, что $C_k = C_{k+1} = C_{k+2} = \ldots$.*

Доказательство. Предположим противное, т.е. что для некоторой последовательности условия теоремы не выполняются. Выберем из этой последовательности строго возрастающую подпоследовательность $C_{n_0} \subsetneq C_{n_2} \subsetneq \ldots$. Тогда для каждого $i > 0$ найдется элемент $x_i \in C_{n_i} \setminus C_{n_{i-1}}$. В силу правильности вазипорядка среди элементов бесконечной последовательности $x_1, x_2, \ldots$ найдется пара $x_i \leq x_j$ при $i < j$. Но x_i принадлежит конусу C_{n_i}, и, следовательно, $x_j \in C_{n_i}$, что противоречит условию $x_j \notin C_{n_{j-1}}$. □

Далее будет рассмотрено распространение квазипорядков на различные структуры — векторы, мультимножества и деревья.

1.2.2 Упорядочение мультимножеств

Определение 1.3. Пусть $\leq$ — некоторый квазипорядок на множестве X. Отношение $\leq^n$ на множестве X^n векторов размерности

n с элементами из X определим, полагая для $\bar{x} = (x_1, x_2, \ldots, x_n)$, $\bar{y} = (y_1, y_2, \ldots, y_n) \in X^n$, $\bar{x} \leq^n \bar{y}$ в том и только том случае, когда $(\forall i, 1 \leq i \leq n) : x_i \leq y_i$.

Утверждение 1.5. *Пусть $\leq$ — правильный квазипорядок на множестве X. Тогда отношение $\leq^n$ на множестве векторов размерности n из X^n также является правильным квазипорядком.*

Доказательство. Пусть имеется некоторая бесконечная последовательность $\bar{x}_0, \bar{x}_1, \bar{x}_2, \ldots$ векторов из множества X^n. Выберем из этой последовательности бесконечную подпоследовательность так, чтобы все первые координаты векторов следовали в возрастающем порядке (относительно порядка $\leq$). Это возможно в силу леммы 1.2. Далее из полученной подпоследовательности выберем в свою очередь бесконечную подпоследовательность, в которой вторые координаты следуют в возрастающем порядке. Повторив этот процесс для всех n координат, получим подпоследовательность, возрастающую в каждой координате, т. е. подпоследовательность векторов $\bar{x}_{i_0} \leq \bar{x}_{i_1} \leq \bar{x}_{i_2}, \ldots$. В этой подпоследовательности любая пара векторов удовлетворяет условию, что $\bar{x}_i \leq \bar{x}_j$ для $i < j$. □

Замечание 1.1. Заметим, что $\leq^n$ не является линейным порядком на X^n даже если $\leq$ — линейный порядок на X.

При упорядочении векторов важен порядок расположения элементов в наборе (векторе). Если же необходимо рассматривать неупорядоченные конечные наборы элементов, то используется понятие мультимножества.

Рассмотрим множество $\mathcal{M}(X)$ всех конечных мультимножеств над множеством X. Мультимножества будем обозначать с помощью квадратных скобок. Так, $[x, x, y]$ — мультимножество, содержащее два элемента x и один элемент y (порядок перечисления элементов, естественно, не существен).

В дальнейшем нам понадобится следующий специальный случай упорядочения мультимножеств.

Определение 1.4. Пусть $\geq$ — некоторый квазипорядок на множестве X. Отношение $\geq_{\mathcal{M}^1}$ на $\mathcal{M}(X)$ определим следующим образом:

1) $[x] \geq_{\mathcal{M}^1} \emptyset$ и $[x] \geq_{\mathcal{M}^1} [y]$, если $x \geq y$;

2) если $M_1 \geq_{\mathcal{M}^1} M_2$, то $M_1 + M \geq_{\mathcal{M}^1} M_2 + M$;

3) если $M_1 \geq_{\mathcal{M}^1} M_2$ и $M_2 \geq_{\mathcal{M}^1} M_3$, то $M_1 \geq_{\mathcal{M}^1} M_3$, т.е. отношение $\geq_{\mathcal{M}^1}$ транзитивно.

Отношение $\geq_{\mathcal{M}^1}$ будем называть *1-упорядочением мультимножеств.*

Другими словами, $M_1 \geq_{\mathcal{M}^1} M_2$, если M_2 может быть получено из M_1 последовательным удалением некоторого элемента x или заменой его на меньший (относительно $\geq$) элемент y. Так, например, $[5, 5, 3, 0, 7] \geq_{\mathcal{M}^1} [5, 6, 2, 0]$.

Замечание 1.2. В [31] Н. Дершовитцем и З. Манной для доказательства завершаемости систем переписывания термов было введено отношение $>_{\mathcal{M}}$ упорядочения мультимножеств. Отношение $>_{\mathcal{M}}$ определяется следующим образом.

Пусть $>$ — некоторый строгий частичный порядок на множестве X. Для мультимножеств $M_1, M_2 \in \mathcal{M}(X)$ полагают $M_1 >_{\mathcal{M}} M_2$, если M_2 может быть получено из M_1 заменой хотя бы одного элемента $x \in M_1$ на некоторое (возможно, нулевое) число меньших (относительно $>$) элементов $y_1, y_2, \ldots, y_n$, при этом каждый из элементов y_i меньше соответствующего заменяемого элемента x.

Таким образом, 1-упорядочение мультимножеств есть сужение упорядочения мультимножеств, при котором каждый элемент заменяется не более чем на один меньший элемент.

Известно, что упорядочение мультимножеств является линейным порядком, если базовый порядок линеен. Для 1-упорядочения мультимножеств это не так. Например, мультимножества $[5,5,3,0,7]$ и $[0,1,7,8]$ не сравнимы.

Н. Дершовитцем и З. Манной в [31] доказано, что отношение $<$ — фундированный частичный порядок на множестве X в том и только том случае, когда $<_{\mathcal{M}}$ — фундированный частичный порядок на множестве $\mathcal{M}(X)$.

Из фундируемости упорядочения мультимножеств следует и фундируемость 1-упорядочения мультимножеств (как сужения). Далее будет доказано более сильное свойство, а именно, что 1-упорядочение мультимножеств является правильным квазиупорядочением при условии, что базовый квазипорядок — правильный.

Теорема 1.1. *Квазипорядок $\leq_{\mathcal{M}^1}$ на множестве $\mathcal{M}(X)$ конечных мультимножеств с элементами из X является правильным тогда и только тогда, когда правильным является квазипорядок $\leq_X$.*

Доказательство. Нетрудно заметить, что $\leq_X$ есть проекция квазипорядка $\leq_{\mathcal{M}^1}$ на множество одноэлементных мультимножеств. Поэтому из правильности квазиупорядочения $\leq_{\mathcal{M}^1}$ на множестве $\mathcal{M}(X)$ следует правильность квазиупорядочения $\leq_X$ на множестве X.

Докажем обратное утверждение. Бесконечную последовательность, удовлетворяющую условию правильности (относительно некоторого порядка $\leq$), т. е. такую последовательность $a_0, a_1, \ldots$, для которой найдется ("хорошая") пара элементов $a_i \leq a_j$ при $i < j$, будем называть "хорошей", а последовательность, для которой это условие не выполняется, — "плохой".

Доказательство проведем от противного. Пусть квазипорядок $\leq_X$ на множестве X является правильным, а соответствующий ему квазипорядок $\leq_{\mathcal{M}^1}$ на множестве $\mathcal{M}(X)$ правильным не является, т.е. существуют "плохие" последовательности.

Построим "плохую" последовательность $A_0, A_1, A_2, \ldots$ в X^* следующим образом. В качестве индукционного предположения полагаем, что для $i < n$ элементы A_i уже построены и существует "плохая" последовательность с началом $A_0, A_1, \ldots, A_{n-1}$. Это, очевидно, верно для $n = 0$, так как по начальному предположению множество $\mathcal{M}(X)$ содержит "плохую" последовательность, и эта последовательность имеет пустой начальный фрагмент. Выберем A_n так, что некоторая "плохая" последовательность начинается с $A_0, A_1, \ldots, A_n$, и мультимножество A_n имеет минимальную из возможных мощность (число элементов с учетом их кратности).

По построению последовательность $A_0, A_1, A_2, \ldots$ является "плохой" и, в частности, $A_n \neq \emptyset$ для всех n. Для каждого n выберем некоторый произвольный элемент $a_n \in A_n$. Пусть $B_n =_{\text{def}} A_n \setminus [a_n]$. В силу правильности отношения $\leq_X$ последовательность $(a_n)_{n\in\mathbb{N}}$ содержит бесконечную возрастающую подпоследовательность $(a_{n_i})_{i\in\mathbb{N}}$. Рассмотрим бесконечную последовательность

$$A_0, \ldots, A_{n_0-1}, B_{n_0}, B_{n_1}, B_{n_2}, \ldots.$$

В силу минимальности выбора A_{n_0} эта последовательность является "хорошей". Тогда она содержит "хорошую" пару элементов, т.е. такую пару, что элемент пары с большим индексом больше (относительно квазипорядка $\leq_{\mathcal{M}^1}$) другого элемента. Эта "хорошая" пара не может быть вида (A_i, A_j) или (A_i, B_j), так как последовательность $(A_i)_{i\in\mathbb{N}}$ — "плохая" и $B_j \leq_{\mathcal{M}^1} A_j$. Тогда эта "хорошая" пара имеет вид (B_i, B_j). Имеем $B_i \leq_{\mathcal{M}^1} B_j$. Но, с другой стороны, $[a_i] \leq_{\mathcal{M}^1} [a_j]$, $A_i = B_i + [a_i]$, $A_j = B_j + [a_j]$ и, следовательно, $A_i \leq_{\mathcal{M}^1} A_j$, т.е. (A_i, A_j) — "хорошая" пара, что противоречит первоначальному предположению. □

Замечание 1.3. Приведенная выше теорема является вариацией известного утверждения, которое было впервые доказано в [46] и известно под названием *леммы Хигмана*. Г. Хигман доказал правильность квазиупорядочения $\preceq_H$ на множестве X^* конеч-

ных последовательностей элементов из X, при условии, что базовое множество X правильно упорядочено. Отношение $\preceq_H$, называемое *вложением Хигмана*, определяется следующим образом. Пусть $\leq_X$ — некоторый квазипорядок на множестве X. Для конечных последовательностей $(x_1, \ldots, x_n), (y_1, \ldots, y_m) \in X^*$ полагаем $(x_1, \ldots, x_n) \preceq_H (y_1, \ldots, y_m)$ в том и только том случае, когда существует инъективное вложение $\phi : [1 \ldots n] \to [1 \ldots m]$ такое, что $\forall i \in [1 \ldots m], x_i \leq y_{\phi(i)}$, где ϕ — строго монотонная функция.

1.2.3 Упорядочение деревьев

Определение 1.5. Пусть $\langle X, \leq_X \rangle$ — некоторое частично упорядоченное множество. Помеченным корневым коммутативным деревом над множеством X (или просто деревом) $t \in \mathbb{T}(X)$ называется пара $(\langle D, \leq, \alpha_0 \rangle, \mathcal{L})$, в которой $\langle D, \leq, \alpha_0 \rangle$ есть частично упорядоченное отношением $\leq$ конечное множество $D = \{\alpha_0, \beta, \gamma \ldots\}$ вершин с выделенным элементом α_0, называемым корнем дерева t, $\mathcal{L} : D \to X$ — функция пометки вершин. При этом требуется выполнение следующих двух условий:

1) $\alpha_0 \geq \beta$ для любого $\beta \in D$,

2) Если $\beta \leq \gamma$ и $\beta \leq \delta$, то либо $\gamma \leq \delta$, либо $\delta \leq \gamma$ для всех $\beta, \gamma, \delta \in D$.

Далее множество вершин D дерева t будем обозначать через $Nodes(t)$. Через $\alpha \wedge \beta$ будем обозначать наименьшую верхнюю грань элементов $\alpha, \beta \in D$, т.е. минимальное δ такое, что $\alpha \leq \delta$ и $\beta \leq \delta$. Смежными будем называть такие две вершины $\alpha, \beta \in Nodes(t)$, что $\alpha \leq \beta$ и не существует вершины δ, для которой $\alpha \leq \delta$ и $\beta \leq \delta$.

Нетрудно заметить, что это определение задает дерево как ориентированный односторонне связный ациклический помеченный граф с выделенной вершиной (корнем), в котором

- дуги направлены от корня к вершинам,
- отношение $\alpha \leq \beta$ соответствует существованию ориентированного пути от α к β,
- для каждой вершины α существует единственный путь от корня к α,
- смежность вершин соответствует обычной смежности вершин в графе,
- каждая вершине в качестве пометки приписан некоторый элемент частично упорядоченного множества X.

Определение 1.6. Пусть $t, t' \in \mathbb{T}(X)$. Будем говорить, что дерево t *(изоморфно) вкладывается* в t', обозначается $t \precsim_X t'$, если существует отображение $\varphi : \mathit{Nodes}(t) \to \mathit{Nodes}(t')$ такое, что

1) φ — инъективна;
2) φ монотонна и переводит смежные вершины в смежные, т.е. для любых смежных вершин $\alpha, \beta \in \mathit{Nodes}(t)$ если $\alpha \leq \beta$, то $\varphi(\alpha) \leq \varphi(\beta)$ и вершины $\varphi(\alpha), \varphi(\beta)$ также смежны;
3) для любой вершины $\alpha \in \mathit{Nodes}(t)$: $\mathcal{L}(\alpha) \leq_X \mathcal{L}'(\varphi(\alpha))$, где $\mathcal{L}, \mathcal{L}'$ — функции пометки вершин для деревьев t и t' соответственно.

Следующее утверждение следует непосредственно из определений.

Утверждение 1.6. *Если отношение $\leq$ является квазипорядком на множестве X, то отношение $\precsim_X$ является квазипорядком на множестве $\mathbb{T}(X)$. Если $\leq$ — частичный порядок на X, то $\precsim_X$ — частичный порядок на $\mathbb{T}(X)$.*

Через $\mathbb{T}_n(X)$ обозначим множество всех деревьев из $\mathbb{T}(X)$, высота которых не превышает n.

В следующей теореме доказывается, что на множестве деревьев ограниченной высоты можно определить правильный квазипорядок.

Теорема 1.2. *Для любого целого $n \geq 0$ квазипорядок $t \precsim_X t'$ на множестве $\mathbb{T}_n(X)$ является правильным тогда и только тогда, когда правильным является квазипорядок $\leq_X$.*

Доказательство. Любая последовательность элементов из X является одновременно последовательностью деревьев высоты 0. Поэтому из правильности квазипорядка $\precsim_X$ на множестве $\mathbb{T}_n(X)$, $n \geq 0$ следует правильность $\leq_X$ на X.

Обратное утверждение докажем индукцией по n. Для $n = 0$ утверждение следует из сказанного выше. Предположим, что на множестве $\mathbb{T}_{n-1}(X)$ квазипорядок $\precsim_X$ является правильным.

Пусть $T_1, T_2, \ldots$ — некоторая бесконечная последовательность деревьев из $\mathbb{T}_n(X)$. Покажем, что в этой последовательности найдутся два "хороших" элемента $T_i \precsim_X N_j$, где $i < j$.

Если последовательность $T_1, T_2, \ldots$ содержит бесконечную подпоследовательность деревьев с высотой, не превышающей $n-1$, то по индукционному предположению в ней найдутся два "хороших" элемента.

Поэтому можно сразу считать, что все деревья в этой последовательности имеют высоту n. Рассмотрим последовательность $a_1, a_2, a_3, \ldots$ пометок, приписанных корневым вершинам деревьев $t_1, t_2, t_3, \ldots$. В силу правильности квазиупорядочения $\leq_X$ на X из этой последовательности можно выбрать бесконечную возрастающую подпоследовательность $a_{i_1} \geq_X a_{i_2} \geq_X a_{i_3} \geq_X \ldots$.

Через T_i обозначим мультимножество всех поддеревьев дерева t_i, порожденных смежными с корнем дерева t_i вершинами. Очевидно, что для любого $i \geq 1$ деревья, входящие в T_i, имеют высоту не более $n-1$ и не являются пустыми. Рассмотрим последова-

тельность $T_{i_1}, T_{i_2}, \ldots$. В силу индукционного предположения квазипорядок $\precsim_X$ является правильным на множестве $\mathcal{T} = \cup_{k\in\mathbb{N}} T_{i_k}$. Тогда по теореме 1.1 на множестве $\mathcal{M}(\mathcal{T})$ мультимножеств с элементами из $\mathcal{T}$ правильным является квазипорядок $(\precsim_X)_{\mathcal{M}^1}$, т.е. в последовательности $T_{i_1}, T_{i_2}, \ldots$ найдутся два мультимножества T_i, T_j $(i < j)$, для которых существует инъективное вложение $\varphi : T_i \to T_j$ такое, что для любого дерева $t \in T_i$ выполняется $t \precsim_X \varphi(t)$. Нетрудно заметить, что в этом случае выполняется также $t_i \precsim_X t_j$, что доказывает правильность $\precsim_X$ на множестве $\mathbb{T}_n(X)$. □

Определение 1.7. Пусть $t, t' \in \mathbb{T}(X)$. Говорят, что дерево t *гомеоморфно вкладывается* в t', обозначается $t \preceq_X t'$, если существует отображение $\varphi : \mathit{Nodes}(t) \to \mathit{Nodes}(t')$ такое, что

1) φ — инъективна;

2) φ монотонна, т.е. для любых вершин $\alpha, \beta \in \mathit{Nodes}(t)$: $\alpha \leq \beta \Rightarrow \varphi(\alpha) \leq \varphi(\beta)$;

3) для любых вершин $\alpha, \beta \in \mathit{Nodes}(t)$: $\varphi(\alpha \wedge \beta) = \varphi(\alpha) \wedge \varphi(\beta)$;

4) для любой вершины $\alpha \in \mathit{Nodes}(t)$: $\mathcal{L}(\alpha) \leq_X \mathcal{L}'(\varphi(\alpha))$, где $\mathcal{L}, \mathcal{L}'$ — функции пометки вершин для деревьев t и t' соответственно.

Другими словами, $t \preceq_X t'$, если найдется функция, отображающая вершины t в вершины t', такая, что вершины t отображаются в вершины t', имеющие меньшие или равные (относительно частичного порядка на X) пометки, а различным дугам в t соответствуют различные пути в t'.

Отношение $\preceq_X$ называется обычно гомеоморфным вложением Краскала и является частичным порядком на $\mathbb{T}(X)$ при условии, что $\leq_X$ — частичный порядок. Для гомеоморфного вложения деревьев имеет место следующая

Теорема 1.3 (Теорема Краскала, [56]). *Если $\leq_X$ — правильный частичный порядок на множестве пометок X, то частичный порядок $\preceq_X$ (гомеоморфное вложение) на множестве корневых помеченных деревьев $\mathbb{T}(X)$ также является правильным.*

Замечание 1.4. Впервые эта теорема была доказана Б. Краскалом в [56]. Более простое ее доказательство, основанное на идее Нэш-Вильямса [75], можно найти в [30]. Новое доказательство и обобщение теоремы Краскала приведено в [71]. На русском языке теорему Краскала с доказательством можно найти в недавно вышедшем переводе книги Р. Дистеля "Теория графов" [6].

1.3 Системы помеченных переходов

1.3.1 Определение

Системы помеченных переходов [53] — одна из наиболее распространенных моделей для описания поведения систем.

Определение 1.8. *Система помеченных переходов* есть четверка $LTS = (S, T, \rightarrow, s_0)$, где

- S есть множество *состояний* с элементами $s_0, s_1, s_2, \ldots$;
- T — некоторый алфавит пометок (*множество имен действий*);
- $\rightarrow \subseteq (S \times T \times S)$ — *отношение перехода* между состояниями;
- $s_0 \in S$ — выделенное состояние, называемое *начальным состоянием* системы.

Переход (s, t, s') обычно обозначается как $s \xrightarrow{t} s'$, что означает, что действие с именем t переводит состояние s в состояние s'. Состояние s' в этом случае называется *t-последующим*, или просто *последующим* для s, а состояние s — *t-предыдущим*, или

просто *предыдущим* для s'. Состояния, не имеющие последующих состояний, называются *финальными*. Если некоторый переход переводит состояние s в состояние s', то пишем $s \to s'$. Через $Succ(s)$ будем обозначать множество последующих состояний для s, через $Pred(s)$ — множество его предыдущих состояний.

Последовательное исполнение для LTS есть конечная или бесконечная цепочка переходов $s_0 \xrightarrow{t_1} s_1 \xrightarrow{t_2} s_2 \to \cdots$, где s_0 — начальное состояние системы. Запись $s \xrightarrow{*} s'$ означает, что имеется (конечная) последовательность переходов, переводящая состояние s в состояние s'.

Диаграмма переходов для LTS есть помеченный, ориентированный граф G_{LTS}, в котором вершинами являются элементы множества состояний S, а дуги определяются отношением переходов так, что дуга, помеченная t, соединяет вершину s с вершиной s' в том и только том случае, когда $s \xrightarrow{t} s'$.

Очевидно, что каждому последовательному исполнению для LTS соответствует ориентированный путь с началом в вершине s_0 в графе G_{LTS}.

1.3.2 Спецификация поведения: язык темпоральной логики

Удобным формализмом для описания динамических свойств параллельных реактивных систем и, в частности, систем помеченных переходов, является язык темпоральной логики [68]. Язык темпоральной логики определяет предикаты на бесконечных последовательностях состояний. Динамические свойства системы помеченных переходов описываются, таким образом, как свойства последовательностей состояний вида $s_0 \to s_1 \to s_2 \to \cdots$.

Язык темпоральной логики строится над языком обычной статической логики (например, логики предикатов первого порядка), который используется для спецификации свойств состояний системы и обычно зависит от конкретной рассматриваемой модели.

В общем случае, при рассмотрении системы помеченных переходов, предполагается, что базовый язык PL содержит набор атомарных формул, а также стандартный набор логических связок, состоящий из отрицания $\neg$, конъюнкции $\wedge$, дизъюнкции $\vee$ и импликации $\Rightarrow$.

Пусть $LTS = (S, T, \rightarrow, s_0)$ — система помеченных переходов. Мы предполагаем, что для всякой атомарной формулы A базового логического языка PL определено логическое значение $A(s)$ формулы A в состоянии $s \in S$. Логические связки будут интерпретироваться стандартным образом. Будем писать $s \models p$, если значение формулы $p \in PL$ в состоянии s равно истине. Формулы темпоральной логики строятся из формул базового логического языка PL с помощью следующих темпоральных операторов: $\bigcirc$ — оператор следования, $\Box$ — оператор "всегда", $\Diamond$ — оператор "когда-нибудь", Until — оператор "до тех пор пока", Unl — оператор "пока не" (Unless) и $\rightsquigarrow$ — оператор "приводит к".

Определение 1.9. Язык TL темпоральной логики определим по индукции:

1) $PL \subset TL$;

2) Если $\varphi, \psi \in TL$, то $\varphi \wedge \psi, \varphi \vee \psi, \varphi \Rightarrow \psi, \neg\varphi \in TL$;

3) Если $\varphi, \psi \in TL$, то $\bigcirc \varphi, \Box \varphi, \Diamond \varphi, \varphi \,\mathrm{Until}\, \psi, \varphi \,\mathrm{Unl}\, \psi, \varphi \rightsquigarrow \psi \in TL$.

Пусть теперь $w = s_0 \rightarrow s_1 \rightarrow \ldots$ — некоторое последовательное исполнение системы помеченных переходов LTS. Через $\vec{s_i}$ будем обозначать подпоследовательность последовательности w, начинающуюся с состояния s_i, т.е. последовательность $s_i \rightarrow s_{i+1} \rightarrow \ldots$.

Истинность формулы $\varphi \in TL$ на последовательности w (обозначается $w \models \varphi$) определяется согласно следующим правилам.

- Для формулы $p \in PL$ полагаем $w \models p$, если $s_0 \models p$.

- $w \models \neg\varphi$, если не выполняется $w \models \varphi$.
- $w \models (\varphi \wedge \psi)$, если $w \models \varphi$ и $w \models \psi$.
- $w \models \bigcirc\varphi$, если для следующего за начальным состоянием состояния s_1 выполняется $\vec{s_1} \models \varphi$.
- $w \models \Box\varphi$, если для любого состояния s_i из w выполняется $\vec{s_i} \models \varphi$.
- $w \models \Diamond\varphi$, если найдется состояние s_i из w, для которого выполняется $\vec{s_i} \models \varphi$.
- $w \models (\varphi \,\mathrm{Unl}\, \psi)$, если либо для всех состояний s_i из w выполняется $\vec{s_i} \models \varphi$, либо найдется состояние s_j в w такое, что $\vec{s_j} \models \psi$ и для всех состояний s_k $(k < j)$ в w выполняется $\vec{s_k} \models \varphi$.
- $w \models (\varphi \rightsquigarrow \psi)$, если для любого состояния s_i из w такого, что $\vec{s_i} \models \varphi$ в w найдется состояние s_j, где $j \geq i$ такое, что $\vec{s_j} \models \psi$.
- $w \models (\varphi \,\mathrm{Until}\, \psi)$, если $w \models (\varphi \rightsquigarrow \psi)$ и $w \models (\varphi \,\mathrm{Unl}\, \psi)$.

Для спецификации свойств поведения системы помеченных переходов LTS будем использовать формулы вида $(\exists)\varphi$ и $(\forall)\varphi$, где $\varphi \in TL$ — формула темпоральной логики. Будем писать $LTS \models (\exists)\varphi$ (или просто $\models (\exists)\varphi$, если LTS ясно из контекста) для обозначения того, что существует исполнение w для LTS, удовлетворяющее формуле φ, т.е. $w \models \varphi$. Формула $LTS \models (\forall)\varphi$ (или просто $\models (\forall)\varphi$) означает, что для любого исполнения w для LTS выполняется $w \models \varphi$.

Формулы $\varphi, \psi \in TL$ назовем *эквивалентными* (обозначается $\varphi \approx \psi$), если для любой последовательности w состояний их значения на w совпадают.

Приведем некоторые эквивалентности, которые связывают между собой темпоральные модальности.

$\neg\Box\,\varphi \approx \Diamond\,\neg\varphi$;
$\neg\Diamond\,\varphi \approx \Box\,\neg\varphi$;
$\neg(\varphi\,\mathrm{Unl}\,\psi) \approx (\neg\varphi)\,\mathrm{Until}\,(\neg\varphi \wedge \neg\psi)$;
$\neg(\varphi\,\mathrm{Until}\,\psi) \approx (\neg\psi)\,\mathrm{Unl}\,(\neg\varphi \wedge \neg\psi)$.

1.3.3 Вполне структурированные системы переходов

Для решения задач анализа семантических свойств систем переходов полезной оказывается теория вполне структурированных систем переходов [40, 41].

Определение 1.10. *Вполне структурированной системой переходов* называется система переходов $LTS = (S, T, \rightarrow, s_0)$, дополненная отношением квазипорядка $\leq\subseteq S \times S$, удовлетворяющим следующим двум условиям:

1) отношение $\leq$ является правильным квазипорядком;

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

Свойство совместимости представляет собой отношение слабой симуляции в смысле Р. Милнера [73]. Оно может быть представлено в виде следующей диаграммы:

$$\begin{array}{ccc} s & \leq & q \\ \downarrow & & \downarrow \\ s' & \leq & (\exists) q' \end{array}$$

1.3.4 Покрывающее дерево системы переходов

Для анализа некоторых важных свойств вполне структурированных систем переходов используется построение *покрывающего де-*

рева. Эта конструкция была предложена А. Финкелем [39] и является обобщением соответствующей конструкции для обыкновенных сетей Петри (см., например, [8]).

Пусть $\mathcal{S} = \langle LTS, \leq \rangle$ — вполне структурированная система переходов.

Определение 1.11. *Покрывающим деревом* вполне структурированной системы переходов $\mathcal{S}$ для состояния $s_0 \in S$ называется конечное ориентированное дерево (связный ациклический граф), такое что:

1) вершины дерева помечены состояниями системы $\mathcal{S}$;

2) каждая вершина объявлена либо *живой*, либо *мертвой*;

3) корню дерева приписана пометка s_0, и эта вершина объявлена живой;

4) мертвые вершины не имеют потомков;

5) живая вершина с пометкой s имеет по одному потомку, помеченному s', для каждого состояния $s' \in Succ(s)$;

6) если на пути от корня до некоторой вершины с пометкой s' встречается вершина с пометкой s, такой, что $s \leq s'$, то говорят, что s' *покрывает* s, и вершина s' объявляется мертвой; в противном случае s' — живая вершина.

Листья в покрывающем дереве помечены финальными или покрывающими состояниями. Для вполне структурированных систем переходов частичный порядок $\leq$ является вполне упорядочиваемым. Благодаря этому все пути в покрывающем дереве конечны, так как всякий бесконечный путь должен был бы содержать покрывающую вершину. В силу леммы Кенига о конечных деревьях если покрывающее дерево не имеет бесконечных ветвлений, то оно конечно.

Более того, очевидно, имеет место следующее

Утверждение 1.7. *Если отношение порядка $\leq$ разрешимо (т.е. существует эффективная процедура проверки истинности $s \leq s'$ для любых s и s') и отображение Succ вычислимо (т.е. существует эффективная процедура вычисления множества $Succ(s)$ для любого состояния s), то покрывающее дерево для вполне структурированной системы переходов может быть эффективно построено.*

Для построения покрывающего дерева не требуется совместимость между отношениями упорядоченности $\leq$ и перехода $\rightarrow$. Однако именно при выполнении условия совместимости покрывающее дерево содержит полезную информацию о свойствах поведения системы.

Пусть, например, подмножество $C \subseteq S$ является верхним конусом. Для обыкновенных сетей Петри это может быть, например, множество разметок, содержащих фишку в некоторой заданной позиции, или множество разметок, при которых некоторый данный переход является возбужденным. Покрывающее дерево позволяет ответить на вопрос: существует ли такая последовательность переходов системы, что все промежуточные состояния будут принадлежать множеству C?

На языке темпоральной логики это свойство запишется формулой $s \models (\exists)\Box C$.

Утверждение 1.8. *Утверждение $s \models (\exists)\Box C$, где C — некоторый верхний конус, истинно для системы переходов LTS тогда и только тогда, когда покрывающее дерево системы содержит путь от корня к листу, в котором все вершины имеют пометки из C.*

Доказательство. Непосредственно из определений. $\Box$

В том случае, когда система переходов является вполне структурированной и, следовательно, порядок $\leq$ — правильный, всякий верхний конус C имеет конечный базис (см. утверждение 1.3),

т. е. представим в виде $C = \{s \in S \mid s \geq s_1 \vee \cdots \vee s \geq s_k\}$, и, таким образом, если отношение $\leq$ разрешимо, то для любого $s \in S$ эффективно проверяется принадлежность конусу C. Если, к тому же, вычислима функция *Succ*, то проверка того, что все вершины некоторого пути в конечном покрывающем дереве имеют пометки из C, может быть эффективно выполнена.

Проверка свойства $s \models (\exists)\Box C$ для конуса C, заданного своим конечным базисом, называется *проблемой поддержки управляющего состояния* (control state maintainability problem). Таким образом, из утверждения 1.8 следует

Утверждение 1.9 ([16]). *Проблема поддержки управляющего состояния разрешима для вполне структурированных систем переходов с разрешимым отношением порядка $\leq$ и вычислимой функцией Succ.*

Пусть теперь $D \subseteq S$ — *нижний конус*, т.е. для любых $x \in D$ и $y \in S$ если $y \leq x$, то $y \in D$. Очевидно, что если D — нижний конус, то его дополнение $C = S \setminus D$ является верхним конусом. *Ко-базисом* D будем называть базис $C = S \setminus D$.

Двойственной к проблеме поддержки управляющего состояния является *проблема неизбежности* (inevitability problem): по данному конечному ко-базису нижнего конуса D и начальному состоянию s_0 определить, верно ли, что всякое вычисление приводит к некоторому состоянию из D. На языке темпоральной логики это означает проверку утверждения $s \models (\forall)\Diamond D$. В качестве двойственного к утверждению 1.9 получаем

Утверждение 1.10 ([16]). *Проблема неизбежности разрешима для вполне структурированных систем переходов с разрешимым отношением порядка $\leq$ и вычислимой Succ.*

Частным, но очень важным, случаем проблемы неизбежности является *проблема останова*: по данному начальному состоянию s_0 определить, верно ли, что всякое вычисление заверша-

ется, т. е. приводит к некоторому финальному состоянию. Множество финальных состояний является нижним конусом в силу свойства совместимости вполне структурированных систем переходов. Действительно, всякое состояние s, которое меньше некоторого финального состояния s', само является финальным, т. к. в противном случае переход t, который может сработать в s, мог бы сработать и в s'.

Следствие 1.2. *Проблема останова разрешима для вполне структурированных систем переходов с разрешимым отношением порядка $\leq$ и вычислимой Succ.*

Доказательство. Для проверки того, что всякое вычисление в системе переходов завершается, достаточно перебрать все листья в конечном покрывающем дереве системы и убедиться, что все они помечены финальными состояниями. □

1.3.5 Метод насыщения

Одной из основных проблем в семантическом анализе динамических систем является *проблема достижимости*, которая состоит в том, чтобы проверить, выполняется ли $s_0 \xrightarrow{*} s$ для двух данных состояний s_0 и s.

Вместо того, чтобы решать проблему достижимости, часто можно ограничиться проверкой того, что может быть достигнуто состояние, *покрывающее* данное состояние s, т. е. состояние $s' \geq s$.

Проблема покрытия: для данного состояния s_0 и заданного своим конечным базисом верхнего конуса C проверить достижимость из s_0 состояния, попадающего в C, т. е. проверить истинность формулы $s_0 \models (\exists)\Diamond C$. В частности, для $C = \uparrow s$, где s — некоторое состояние системы, проблема покрытия состоит в проверке достижимости из состояния s_0 состояния $s' \geq s$.

Для решения проблемы покрытия в случае вполне структурированных систем переходов используется *метод насыщения* (saturation method, [41, 16]), основанный на свойстве стабилизации воз-

растающей последовательности верхних конусов (см. утверждение 1.4).

Пусть $LTS = \langle S, \to, \leq \rangle$ — вполне структурированная система переходов. Напомним, что $\uparrow X =_{\text{def}} \{y \in S \mid (\exists x \in X) : y \geq x\}$, где $X \subseteq S$. Через $Pred(s)$ будем обозначать множество $\{s' \in S \mid s \to s'\}$ непосредственных предшественников состояния s. Соответственно, для множества состояний $X \subseteq S$ полагаем $Pred(X) =_{\text{def}} \cup_{s \in X} Pred(s)$.

Пусть C — верхний конус. Рассмотрим последовательность $C_0 \subseteq C_1 \subseteq \ldots$, где $C_0 =_{\text{def}} C$, $C_{i+1} =_{\text{def}} C_i \cup Pred(C_i)$. Через $Pred^*(C)$ обозначим предел этой последовательности, то есть $Pred^*(C) =_{\text{def}} \cup_{i \geq 0} C_i$. Решение проблемы покрытия для заданных состояний s_0 и s состоит в проверке того, что $s \in Pred^*(\uparrow s_0)$.

Утверждение 1.11. *Если $C \subset S$ — верхний конус (относительно вполне упорядочиваемого порядка $\leq$), то $Pred^*(C)$ также является верхним конусом (относительно $\leq$).*

Доказательство. Покажем сначала, что если C — верхний конус, то $Pred(C)$ также является верхним конусом. Действительно, пусть состояние $s \in Pred(C)$, т.е. существует $s' \in C$, такое что $s \to s'$, и пусть $r \geq s$. По свойству совместимости найдется состояние r' такое, что $r \to r'$ и $r' \geq s'$, т.е. $r \in Pred(r')$ и $r' \in C$. Это значит, что $r \in Pred(C)$.

Поскольку объединение двух верхних конусов также является верхним конусом, применив индукцию, из доказанного получаем, что для любого $i > 0$ множество C_i, где $C_0 =_{\text{def}} C$, $C_{i+1} =_{\text{def}} C_i \cup Pred(C_i)$, также является верхним конусом.

Пусть теперь $s \in Pred^*(C)$. Тогда для некоторого $i > 0$ выполняется $s \in C_i$. Поскольку C_i — верхний конус, из $r \geq s$ следует $r \in Pred^i(C)$, а, значит, и $r \in Pred^*(C)$. □

Из приведенного выше утверждения и утверждения 1.4 следует, что последовательность $C_0 \subseteq C_1 \subseteq \ldots$, где $C_0 =_{\text{def}} C$,

$C_{i+1} =_{\text{def}} C_i \cup Pred(C_i)$, стабилизируется, т.е. начиная с некоторого k выполняется $C_k = C_{k+1} = \cdots = Pred^*(C)$. Заметим, что стабилизация наступает после того, как в последовательности два последовательных множества оказываются равными.

Множество $Pred^*(C)$ может быть эффективно вычислено при условии еще одного дополнительного предположения.

Определение 1.12. Будем говорить, что вполне структурированная система переходов $\mathcal{S} = \langle LTS, \leq \rangle$ *имеет эффективный предбазис*, если для любого состояния $s \in S$ может быть эффективно вычислен конечный базис $pb(s)$ множества $Pred(\uparrow s)$.

Пусть вполне структурированная система переходов $\mathcal{S} = \langle LTS, \leq \rangle$ имеет эффективный предбазис. Для верхнего конуса C с конечным базисом C^b построим последовательность $B_0, B_1, \ldots$ множеств таких, что $B_0 =_{\text{def}} C^b$ и $B_{i+1} =_{\text{def}} B_i \cup pb(B_i)$. Рассмотрим последовательность $\uparrow B_0 \subseteq \uparrow B_1 \subseteq \ldots$. В силу утверждения 1.4 эта последовательность стабилизируется. Пусть m — первый индекс, для которого $\uparrow B_m = \uparrow B_{m+1}$. Имеет место следующее

Утверждение 1.12. *При определенных выше условиях*

1) $\uparrow B_m = \uparrow \cup_{i \in \mathbb{N}} B_i$;

2) $\uparrow \cup_{i \in \mathbb{N}} B_i = Pred^*(C)$.

Доказательство. Первое утверждение следует из дистрибутивности операций $Pred$ и $\uparrow$ относительно объединения множеств, а также из того, что если для множеств R и R' выполняется $\uparrow R = \uparrow R'$, то $\uparrow pb(R) = \uparrow pb(R')$.

Докажем второе утверждение. Индукцией по n проверяется, что $B_n \subseteq \uparrow B_n \subseteq Pred^*(C) (= \uparrow Pred^*(C))$.

Но, с другой стороны, из определения предбазиса следует, что $\uparrow Pred^n(C) \subseteq \uparrow B_n$ и, следовательно,

$$Pred^*(C) \subseteq \cup_{i \in \mathbb{N}} \uparrow B_i \subseteq \uparrow \cup_{i \in \mathbb{N}} B_i \subseteq \uparrow Pred^*(C). \qquad \square$$

Эффективное решение проблемы покрытия основано на следующем утверждении:

Утверждение 1.13 ([40]). *Пусть вполне структурированная система переходов $\mathcal{S} = \langle LTS, \leq \rangle$ имеет эффективный предбазис и разрешимое отношение $\leq$. Тогда существует эффективная процедура построения конечного базиса множества $Pred^*(C)$ по произвольному верхнему конусу C, заданному своим конечным базисом.*

Доказательство. В силу доказанного выше можно эффективно порождать последовательность $B_0, B_1, \ldots$, поскольку каждое множество B_i конечно и pb эффективно вычисляется. Из разрешимости $\leq$ следует разрешимость проверки $\uparrow B = \uparrow B'$ для конечных множеств B и B', и, следовательно, индекс m, начиная с которого последовательность стабилизируется, может быть эффективно вычислен. В результате получаем B_m — вычислимый базис верхнего конуса $Pred^*(C)$. □

Теорема 1.4 ([40]). *Проблема покрытия разрешима для вполне структурированных систем переходов, имеющих эффективный предбазис и разрешимое отношение порядка $\leq$.*

Доказательство. Проблема покрытия состоит в проверке $s_0 \in Pred^*(\uparrow s)$ для двух данных состояний s_0 и s. Согласно утверждению 1.13 можно эффективно вычислить конечный базис B множества $Pred^*(\uparrow s)$. Тогда проблема сводится к проверке $s_0 \in \uparrow B$. В силу разрешимости $\leq$ проверка $s_0 \in \uparrow B$ для конечного множества B также может быть эффективно выполнена.

В том случае, когда выполняется проверка более общего свойства $s \in Pred^*(\uparrow C)$ для верхнего конуса C, заданного своим конечным базисом $b_1, \ldots, b_k$, имеем $\uparrow C = \cup_{i=1,\ldots,k} \uparrow b_i$, и достаточно проверить, выполняется ли $s \in Pred^*(\uparrow b_i)$ для некоторого $i = 1, \ldots, k$. □

Глава 2

Одноуровневые сети Петри

2.1 Сети Петри

В этом разделе будут даны базовые определения сетей Петри.

2.1.1 Сети

Определение 2.1. *Сеть* $N = (P_N, T_N, F_N)$ есть двудольный ориентированный граф с конечным множеством вершин $P_N \cup T_N$, где $P_N \cap T_N = \emptyset$, и множеством дуг $F_N \subseteq (P_N \times T_N) \cup (T_N \times P_N)$.

Как правило, вместо P_N, T_N, F_N мы будем писать просто P, T, F. Элементы $P = \{p, ...\}$ называются *позициями* и изображаются кружками, элементы $T = \{t, \dots\}$ называются *переходами* и изображаются прямоугольниками, ориентированные дуги из F изображаются стрелками, соединяющими некоторые позиции и переходы. Через (u, v) будем обозначать дугу, соединяющую вершины u и v.

Определение 2.2. Пусть $N = (P, T, F)$ есть сеть. Если $(u, v) \in F$, то говорят, что u является *входным* элементом для v, а v — *выходным* элементом для u в N. В том случае, когда отношение

F ясно из контекста, через ${}^\bullet u$ будем обозначать множество $\{v \in (P \cup T) \mid (v,u) \in F\}$ входных и через $u^\bullet$ — множество $\{v \in (P \cup T) \mid (u,v) \in F\}$ выходных элементов для вершины u.

Для подмножеств $U \subseteq (P \cup T)$ полагаем ${}^\bullet U =_{\text{def}} \bigcup_{u \in U} {}^\bullet x$ и $U^\bullet =_{\text{def}} \bigcup_{u \in U} x^\bullet$.

2.1.2 Обыкновенные сети Петри

Определение 2.3. *Обыкновенной сетью Петри* называется набор $PN = (N, W)$, где $N = (P, T, F)$ есть конечная сеть (т.е. множество $P \cup T$ конечно), $W : F \to \mathbb{N} \setminus \{0\}$ — функция, задающая кратность дуг.

На основе отношения инцидентности F и функции кратности дуг W можно построить функцию инцидентности, которую мы также будем обозначать через F. Полагаем

$$F(x,y) = \begin{cases} n, & \text{если } xFy \wedge W(x,y) = n, \\ 0, & \text{если } \neg(xFy). \end{cases}$$

Текущее состояние системы определяется разметкой сети Петри. Слова ‘состояние’ и ‘разметка’ применительно к сети Петри мы будем употреблять далее как синонимы.

Определение 2.4. Пусть $PN = (P, T, F, W)$ — обыкновенная сеть Петри. *Разметкой* сети PN называется функция вида $M : P \to \mathbb{N}$. Множество всех разметок сети PN будем обозначать через $\mathfrak{M}(PN)$.

Маркированной сетью будем называть пару (PN, M_0) — сеть Петри PN вместе с некоторой выделенной разметкой, называемой *начальной разметкой* сети PN.

Если из контекста ясно, о чем идет речь, маркированную сеть будем называть просто сетью.

Элементы начального состояния сети графически выделяются фишками (черными точками) внутри соответствующих кружков.

Определим поведение сети Петри.

Определение 2.5. Пусть $PN = (P, T, F, W)$ — обыкновенная сеть Петри. Переход $t \in T$ является *активным* при разметке M, если для любой позиции $p \in {}^\bullet t : M(p) \geq F(p, t)$.

Пусть некоторый переход t является активным в состоянии M. Тогда состояние M' такое, что $M'(p) = M(p) - F(p, t)$ для всех $p \in {}^\bullet t$, $M'(p) = M(p) + F(t, p)$ для $p \in t^\bullet$, и $M'(p) = M(p)$ для $p \notin ({}^\bullet t \cup t^\bullet)$, называется *результатом* срабатывания t при разметке M. В этом случае тройка (M, t, M') называется *шагом срабатывания* в PN и обозначается $M[t\rangle M'$. Если не важно, какой именно переход сработал, пишем $M[\,\rangle M'$

Разметка называется *тупиковой*, если при этой разметке не может сработать ни один из переходов сети.

Допустимой последовательностью срабатываний R сети PN называется конечная или бесконечная последовательная комбинация шагов в PN:$M_0[t_1\rangle M_1[t_2\rangle M_2 \ldots$, где M_0 есть начальная разметка сети PN.

Разметка M сети PN называется *достижимой*, если для некоторой допустимой последовательности срабатываний $M_0[t_1\rangle M_1[t_2\rangle M_2 \ldots [t_n\rangle M_n$ выполняется $M_n = M$.

Аналогично, переход $t \in T$ сети PN называется *достижимым*, если существует допустимая последовательность срабатываний $M_0[t_1\rangle M_1[t_2\rangle M_2 \ldots [t_n\rangle M_n$, в которой $t_n = t$.

Последовательным исполнением сети PN называется допустимая последовательностью срабатываний, которая не может быть продолжена.

На рисунке 2.1 приведен пример обыкновенной сети Петри, моделирующей химическую реакцию синтеза воды из водорода и кислорода. В правой части рисунка показана сеть, получающаяся в результате срабатывания перехода. После этого срабатывания новое срабатывание перехода уже невозможно из-за недостаточности ресурсов, представленных фишками во входных позициях.

Таким образом, последовательное исполнение сети PN есть либо бесконечная допустимая последовательность срабатываний,

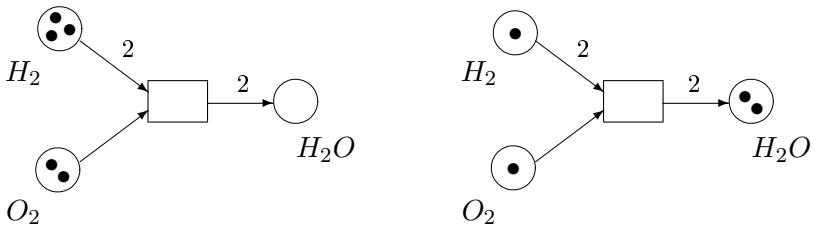


Рис. 2.1: Химическая реакция

либо конечная последовательность $M_0[t_1\rangle M_1[t_2\rangle M_2 \ldots [t_n\rangle M_n$, в которой разметка M_n является тупиковой.

Множество всех конечных и бесконечных допустимых последовательностей срабатывания сети PN можно представить в виде путей в некотором конечном графе.

Определение 2.6. *Графом переходов* G_{PN} сети PN назовем ориентированный помеченный граф с выделенной вершиной, в котором

- вершинами являются все достижимые разметки сети PN;
- начальная разметка сети PN является выделенной вершиной графа G_{PN};
- дуги графа G_{PN} помечены переходами сети PN;
- в графе G_{PN} вершины M_1 и M_2 соединены стрелкой, помеченной переходом t, в том и только том случае, когда $M_1[t_2\rangle M_2$ есть шаг срабатывания сети PN.

Утверждение 2.1. *Пусть PN — сеть, G_{PN} — соответствующий ей граф переходов. Тогда*

1) *Любой ориентированный путь в графе G_{PN} задает допустимую последовательность срабатываний сети PN.*

2) Множество всех максимальных конечных и бесконечных ориентированных путей графа G_{PN} с началом в выделенной вершине совпадает с множеством всех последовательных исполнений сети PN.

2.1.3 Элементарные сети Петри

В элементарных сетях Петри разрешаются только разметки, в которых каждая позиция содержит не более одной фишки. Соответственно, срабатывание перехода может перемещать из некоторой позиции или добавлять в некоторую позицию не более одной фишки. Поэтому веса всех дуг в элементарной сети одинаковы и равны 1, а в определении элементарной сети функция W отсутствует.

Позиция p в элементарной сети может интерпретироваться как некоторое условие (высказывание). В заданной разметке условие p может быть либо истинно (позиция p содержит фишку), либо ложно (позиция p пуста). Таким образом, разметка элементарной сети задается как некоторое подмножество множества позиций (условий, истинных в данном состоянии).

Обычно элементарной сетью называют уже маркированную сеть, т. е. начальная разметка включается в определение элементарной сети.

Определение 2.7. Пара $EN = (N, M_0)$, в которой $N = (P, T, F)$ есть конечная сеть, $M_0 \subseteq P$ — подмножество множества P, называемое *начальной разметкой сети EN*, называется *элементарной сетью Петри* (E-сетью).

Как и в обыкновенных сетях Петри, элементы начальной разметки графически помечаются черными точками в соответствующих позициях.

Поскольку позиция в E-сети может содержать не более одной метки, срабатывание некоторого перехода t возможно только при условии, что выходные для t позиции не содержат фишек (иначе

срабатывание t привело бы к добавлению в некоторую позицию второй фишки).

Определение 2.8. Пусть $PN = (N, M_0)$ — некоторая E-сеть, где $N = (P, T, F)$.

- Любое подмножество $M \subseteq P$ есть *разметка*.
- Переход $t \in T$ является *активным* при разметке M, если ${}^\bullet t \subseteq M$ и $(t^\bullet \setminus {}^\bullet t) \cap M = \emptyset$.
- Пусть некоторый переход t является активным при разметке M. Тогда $M' \equiv (M \setminus {}^\bullet t) \cup t^\bullet$ называется *результатом* срабатывания t при разметке M. В этом случае тройка (M, t, M') называется *шагом* в EN и обозначается $M[t\rangle M'$.

Понятия последовательного исполнения, а также достижимых состояния и перехода для E-сетей определяются так же, как и для обыкновенных сетей Петри.

Таким образом, в отличие от обыкновенных сетей Петри, в элементарных сетях условие активности некоторого перехода t зависит не только от разметки его входных позиций, но и от наличия фишек в выходных позициях. Ситуация, когда переход не может сработать по причине наличия фишки в его выходной позиции, называется контактом.

Определение 2.9. *Контакт* в E-сети EN определяется как пара (M, t), где M — достижимое состояние, t — переход в EN, такие что ${}^\bullet t \subseteq M$ и $(t^\bullet \setminus {}^\bullet t) \cap M \neq \emptyset$.

E-сеть называется *контактно-свободной*, если она не имеет контактов.

На рисунке 2.2 приведен содержательный пример элементарной сети Петри $\mathcal{EN}$, взятый нами из [85]. Эта сеть моделирует работу агентства по продаже автомобилей. Элементарная сеть $\mathcal{EN}$ описывает систему с точки зрения обслуживания агентством

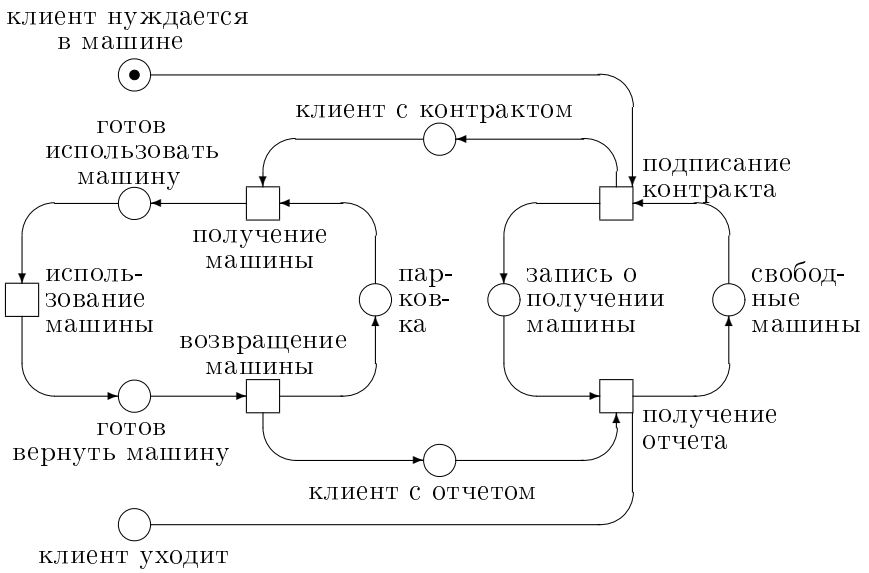


Рис. 2.2: Элементарная сеть $\mathcal{EN}$

одного клиента. Каждая позиция сети соответствует состоянию этого клиента или агентства, которое представляется логическим высказыванием, принимающем значение “истина” или “ложь”. Начальное состояние данной сети задается позицией “клиент нуждается в машине”.

2.1.4 Сети позиций/переходов

Как и в обыкновенных сетях Петри, в сетях позиций/переходов позиции могут содержать несколько фишек, а дуги помечены весами, задающими их кратность. Кратность дуги соответствует числу фишек, одновременно “переносимых” по этой дуге. Отличие

сетей позиций/переходов от обыкновенных сетей состоит в том, что каждой позиции приписана емкость — максимальное число фишек, которые эта позиция может содержать.

Определение 2.10. *Сетью позиций/переходов* (P/T-сетью) называется четверка $PTN = (N, K, W, M_0)$, где $N = (P, T, F)$ есть конечная сеть, $K : P \to \mathbb{N} \setminus \{0\}$, $W : F \to \mathbb{N} \setminus \{0\}$ — функции, задающие емкость позиций и кратность дуг соответственно, $M_0 : P \to \mathbb{N}$ — функция, задающая начальную разметку сети.

Таким образом, элементарная сеть — это такая сеть позиций/переходов, в которой емкости всех позиций и кратности всех дуг равны 1.

Начальная разметка сети графически обозначается соответствующим числом черных точек в позициях сети.

Определение 2.11. Пусть $PTN = (N, K, W, M_0)$ — некоторая P/T-сеть, где $N = (P, T, F)$. *Разметкой* сети PTN называется функция вида $M : P \to \mathbb{N}$.

Переход $t \in T$ является *активным* при разметке M, если для любой позиции $p \in {}^\bullet t : M(p) \geq W(p, t)$ и для любой позиции $q \in t^\bullet : M(q) + W(t, q) \leq K(q)$.

Пусть некоторый переход t является активным при разметке M. Тогда разметка M' такая, что $M'(p) = M(p) - W(p, t)$ для всех $p \in {}^\bullet t$, $M'(p) = M(p) + W(t, p)$ для $p \in t^\bullet$, и $M'(p) = M(p)$ для $p \notin ({}^\bullet t \cup t^\bullet)$, называется *результатом* срабатывания t при разметке M. Тройка (M, t, M') называется *шагом* в PTN и обозначается $M[t\rangle M'$.

Понятия последовательного исполнения, а также достижимого состояния и перехода для P/T-сетей определяются так же, как и для обыкновенных сетей Петри.

Как и для элементарных сетей, для P/T-сетей возможны ситуации контакта.

Определение 2.12. *Контакт* в P/T-сети PTN определяется как пара (M, t), где M — достижимая разметка, t — переход в PTN, такие что для любой позиции $p \in {}^\bullet t : M(p) \geq W(p,t)$ и найдется такая позиция $q \in t^\bullet$, что $M(q) + W(t,q) > K(q)$.

P/T-сеть называется *контактно-свободной*, если она не имеет контактов.

2.2 Сети Петри высокого уровня

Сети Петри высокого уровня характеризуются следующими особенностями:

- разметка сети задается с помощью индивидуальных, т. е. различимых между собой фишек;
- позиции могут содержать кортежи (n-ки) индивидуальных фишек; размер кортежа определяется арностью позиции;
- переходы могут срабатывать в различных режимах, удаляя фишки из одних позиций и добавляя их в другие, при этом единственным априорным ограничением является требование локальности, т. е. в любом режиме переход может удалять фишки только из своих входных позиций и добавлять только в выходные позиции.

2.2.1 Базисные сети Петри высокого уровня

Сначала приведем самое общее определение сети Петри высокого уровня [90].

Определение 2.13. *Базисной сетью Петри высокого уровня* называется набор $BHL = (N, D, \Phi, W, K, M_0)$, где

1) $N = (P, T, F)$ — сеть, при этом каждой позиции $p \in P$ сопоставлено натуральное число $n \geq 1$, называемое *арностью* позиции p.

2) D — функция, сопоставляющая каждой позиции $p \in P$ некоторый непустой *домен* $D(p)$. Элементами $D(p)$ являются кортежи индивидуальных фишек. Размерность всех кортежей из $D(p)$ совпадает с арностью позиции p.

3) *Разметкой* называется отображение $M : P \to \mathcal{M}(D(p))$, т. е. разметка сопоставляет каждой позиции сети некоторое мультимножество (возможно пустое) кортежей — элементов из соответствующего этой позиции домена.

4) $K : P \to \mathcal{M}(D(p))$ — *функция емкости позиций*, которая имеет те же области определения и значений, что и функция разметки, но в отличие от последней значение $K(p)(a)$ может быть не определено для некоторых $a \in D(p)$. В качестве разрешенной разметки сети будут допускаться только разметки M, для которых $M(p)(a) \leq K(p)(a)$, если значение емкости определено. Другими словами, значение $K(p)(a)$ задает максимальное допустимое в позиции p количество экземпляров кортежа a.

5) Φ — функция, сопоставляющая каждому переходу t некоторое непустое множество *режимов срабатывания* $\Phi(t)$.

6) W — функция, областью определения которой является множество F дуг сети N. Дуге (p,t) функция W сопоставляет значение $W(p,t) \in \mathcal{M}(D(p) \times \Phi(t))$, соответственно, $W(t,p) \in \mathcal{M}(\Phi(t) \times D(p))$. Содержательно $W(p,t)$ определяет мультимножество фишек, удаляемых из позиции p, а $W(t,p)$ — мультимножество фишек, добавляемых в позицию p в результате срабатывания перехода t в режиме $\Phi(t)$.

7) M_0 — некоторая выделенная разметка сети, называемая *начальной разметкой*.

Поведение сети *BHL* определяется следующим образом:

Определение 2.14. Пусть $BHL = (N, D, \Phi, W, K, M_0)$ — базисная сеть Петри высокого уровня. Переход t сети BHL является *активным* в режиме $\phi \in \Phi(t)$ при разметке M, если
$\forall p \in {}^\bullet t, \forall a \in D(p) : W(p,t)(a,\phi) \leq M(p)(a)$ и
$\forall p \in t^\bullet, \forall a \in D(p) : M(p)(a) + W(t,p)(\phi,a) \leq K(p)(a)$.
Емкостные ограничения учитываются только в том случае, когда значение $K(p)(a)$ определено.

Если переход является активным, то *срабатывание* t в режиме ϕ переводит разметку M в разметку M', где $M(p) = M'(p)$ для $p \notin {}^\bullet t \cup t^\bullet$ и для $p \in {}^\bullet t \cup t^\bullet$ и $a \in D(p)$ имеем:
$M'(p)(a) = M(p)(a) - W(p,t)(a,\phi) + W(t,p)(\phi,a)$.

Через $M[t[\phi]\rangle M'$ будем обозначать срабатывание перехода t в режиме ϕ, переводящее разметку M в разметку M'.

2.2.2 Предикатные сети Петри

Известно несколько специальных классов сетей Петри высокого уровня. Предикатные сети (PrT-сети) были введены Г. Генрихом и К. Лаутенбахом [43].

Напомним некоторые понятия логики предикатов, необходимые для определения PrT-сетей.

Язык $\mathcal{L}$ определяется как набор символов отношений, функциональных символов и (индивидуальных) символов констант. Каждому символу отношения приписана его арность $n \geq 1$, соответственно, функциональному символу приписано число его аргументов $m \geq 1$.

Предполагается также, что заданы бесконечное множество Var индивидуальных переменных и стандартный набор логических связок и кванторов. *Термы* $\theta, \ldots$ и *формулы* $\varphi, \ldots$ языка $\mathcal{L}$ определяются обычным образом.

Моделью языка $\mathcal{L}$ называется пара $\mathcal{U} = (\mathrm{A}, \mathcal{I})$, где A — домен, $\mathcal{I}$ — функция интерпретации, сопоставляющая символам языка $\mathcal{L}$ соответствующие отношения, функции и константные элементы над A.

Означиванием называется отображение $b : Var \to \mathrm{A}$. *Значение* $\theta[b]$ терма θ при означивании b определяется по индукции обычным образом. Для кортежа термов $\overline{\theta} = (\theta_1, \ldots, \theta_n)$ полагаем $\overline{\theta}[b] =_{\mathrm{def}} (\theta_1[b], \ldots, \theta_n[b])$. Для мультимножества Θ термов (кортежей термов) через $\Theta[b]$ будем обозначать мультимножество значений, получаемых при означивании b всех термов из Θ (в $\Theta[b]$ учитываются все кратные вхождения значений). И, наконец, $\mathcal{U} \models \varphi[b]$ означает истинность формулы φ в модели $\mathcal{U}$ при означивании b и определяется как обычно.

Теперь перейдем к определению предикатных сетей Петри. В предикатной сети каждой позиции $p \in P$ приписано некоторое натуральное число n, задающее арность этой позиции. Содержательно это означает, что фишками, которые могут находиться в позиции p, являются n-ки элементов домена.

Определение 2.15. *Предикатной сетью Петри* (PrT-сетью) называется набор $PrT = (N, \mathcal{L}, \mathcal{U}, W, G, M_0)$, где

1) $N = (P, T, F)$ — сеть.

2) P — множество символов с приписанной им арностью.

3) $\mathcal{L}$ — язык, не содержащий символов из P.

4) $\mathcal{U} = (\mathrm{A}, \mathcal{I})$ — модель языка $\mathcal{L}$.

5) W — функция, сопоставляющая каждой дуге $(x, y) \in F$ некоторое конечное мультимножество $W(x, y)$ n-ок термов $\overline{\theta} = \{\theta_1, \ldots, \theta_n\}$, где $\theta_i \in \mathcal{L}$ $(1 \leq i \leq n)$ и n есть арность позиции, инцидентной дуге (x, y).

6) G — частичная функция с областью определения T, сопоставляющая некоторым $t \in T$ (не обязательно всем) формулы φ_t, называемые *охранами переходов*;

7) M_0 — *начальная разметка сети*, где вообще *разметка* есть отображение M, которое сопоставляет каждой позиции $p \in$

P некоторое мультимножество $M(p)$ над множеством A^n кортежей (n — арность позиции p).

По техническим причинам мы требуем, чтобы $\mathcal{L}$ содержал хотя бы один символ константы (это нужно для существования замкнутых термов). Обычно нет необходимости формально определять язык $\mathcal{L}$, он может быть неявно задан путем указания $\mathcal{U}$ и функций W и G.

Выражения $W(x,y)$, приписанные дугам, будем обычно представлять в виде $\overline{\theta}_1 + \cdots + \overline{\theta}_m$. Здесь каждый $\overline{\theta}_i$ есть кортеж из n термов, где n — арность позиции $p \in (x,y)$. Будем также использовать обозначение $-W(x,y)$ для выражения $-\overline{\theta}_1 - \cdots - \overline{\theta}_m$. Символ 0 будем использовать для обозначения кортежа из нулевых элементов. В некоторых случаях разметки также будем представлять в виде символьных сумм.

Определим поведение предикатной сети. Оно основано на срабатывании означенных переходов вида $t[b]$, которые задаются путем означивания b переменных в охране φ_t и выражениях на инцидентных t дугах. Нежелательные срабатывания могут быть предотвращены с помощью охран φ_t.

Определение 2.16. Пусть $PrT = (N, \mathcal{L}, \mathcal{U}, W, G, M_0)$ — некоторая PrT-сеть.

1) Означивание $b : Var \to A$ является *допустимым* для перехода t, если $\mathcal{U} \models \varphi_t[b]$.

2) *Означенный переход* t (называемый также режимом срабатывания перехода t) есть пара (t,b) (обозначается $t[b]$) составленная из перехода и допустимого означивания.

3) Означенный переход $t[b]$ является *активным* при разметке M, если $W(p,t)[b] \subseteq M(p)$ для всех $p \in {}^\bullet t$ (обозначение $M[t[b]\rangle$).

4) Если $M[t[b]\rangle$, то *срабатывание означенного перехода* $t[b]$ переводит разметку M в разметку M', такую что $M'(p) = M(p) - W(p,t)[b] + W(t,p)[b]$. Обозначение для такого срабатывания: $M[t[b]\rangle M'$. Если нас интересует только изменение разметки, и не важно, какой именно переход сработал, пишем $M[\,\rangle M'$.

Замечание 2.1. В изначальном определении PrT-сетей в работе [43] предполагалось выполнение дополнительного ограничения на емкость позиций: в каждой позиции может находиться не более одного экземпляра индивидуальной фишки (кортежа). В этом случае каждой позиции сети соответствует некоторое n-местное отношение. Означивание перехода считается допустимым, если значения выражений на входных дугах любой позиции из $t^\bullet$ попарно различны. Означенный переход объявляется активным, если его срабатывание не приносит в какую-либо позицию второй экземпляр некоторой индивидуальной фишки. Таким образом, в определении из [43] предикатные сети Петри являются обобщением элементарных сетей на случай, когда в позициях допускаются только различимые между собой индивидуальные фишки. Как и в элементарных сетях, в них не разрешаются разметки, при которых некоторая позиция содержит две или более одинаковые фишки. Известно, что PrT-сети, определенные в [43], моделируются элементарными сетями Петри. Позднее понятие PrT-сетей было обобщено и ограничение на кратные вхождения фишки было снято. В этом случае справедливо утверждение, что PrT-сети моделируются обыкновенными сетями Петри.

2.2.3 Абстрактные предикатные сети Петри

Предикатная сеть Петри определяется над некоторой конкретной моделью. Однако часто бывает удобно иметь одно описание для класса подобных систем, например, зависящих от некоторого параметра. В этих случаях можно вместо конкретной модели рас-

сматривать аксиоматическое описание класса моделей. Такой подход используется в [42], где вводится понятие абстрактных предикатных систем или сетей-схем.

Через Ax обозначим набор аксиом (множество формул первого порядка). $Ax \vdash \varphi$ означает выводимость формулы φ из множества аксиом Ax.

Определение 2.17. *Абстрактной предикатной сетью Петри* называется четверка $APrT = (N, \mathcal{L}, W, G, Ax)$, где $N, \mathcal{L}, W, G$ определяются так же, как и для предикатных сетей Петри, а Ax есть множество формул первого порядка из $\mathcal{L}$.

Заметим, что в это определение не входит начальная разметка сети. Это связано с тем, что абстрактная предикатная сеть рассматривается как общая схема для некоторого набора конкретных сетей.

Обобщение понятия разметки и поведения системы дается в следующем определении.

Определение 2.18. *Разметка* абстрактной предикатной сети $APrT$ есть отображение M, которое каждой позиции $p \in P$ сопоставляет мультимножество $\{\overline{\theta}_1, \ldots, \overline{\theta}_k\}$ n-ок, где компоненты $\theta_{i_1}, \ldots, \theta_{i_n}$ кортежа $\overline{\theta}_i$, $1 \leq i \leq n$, являются замкнутыми термами языка $\mathcal{L}$.

Поведение $APrT$-сети можно определить аналогично поведению предикатной сети. Прежде всего заметим, что для $APrT$-сети означивание b можно определить как синтаксическую замену переменных замкнутыми термами. Тогда условие допустимости означивания b для перехода t переформулируется следующим образом: $Ax \vdash \varphi_t[b]$.

Следующее определение совпадает с определением для предикатных сетей.

Определение 2.19. Означенный переход $t[b]$ является *активным* при разметке M (обозначение $M[t[b]\rangle$), если $W(p,t)[b] \subseteq M(p)$ для всех $p \in {}^\bullet t$.

Если $M[t[b]\rangle$, то *срабатывание означенного перехода* $t[b]$ переводит разметку M в разметку M', такую что $M'(p) = M(p) - W(p,t)[b] + W(t,p)[b]$. Обозначение для такого срабатывания: $M[t[b]\rangle M'$. Если нас интересует только изменение разметки и не важно, какой именно переход сработал, пишем $M[\,\rangle M'$.

Замечание 2.2. Как и для PrT-сетей, в изначальном определении APrT-сетей ([42]) предполагается, что в каждой позиции может находиться не более одного экземпляра индивидуальной фишки. В качестве синтаксического варианта этого условия требуется, чтобы все τ_i, τ_j были попарно различны в следующем смысле:

$$Ax \not\vdash \theta_{i_1} = \theta_{j_1} \wedge \cdots \wedge \theta_{i_n} = \theta_{j_n}.$$

Замечание 2.3. Во многом аналогичны абстрактным предикатным сетям алгебраические сети Петри [86]. При определении алгебраических сетей используются понятия алгебраических спецификаций и абстрактных типов данных. Основные отличия от абстрактных предикатных сетей состоят в следующем: переходы в алгебраических сетях не имеют охран, множество аксиом либо пусто, либо состоит из равенств термов. Кроме того, начальная разметка входит в определение алгебраической сети.

2.2.4 Раскрашенные сети Петри

Раскрашенные сети Петри представляют собой еще один широко используемый формализм сетей Петри высокого уровня. Они были предложены К. Йенсеном [50, 51] почти одновременно с появлением предикатных сетей Петри. Раскрашенные сети Петри основаны на языке с типами, которые называются цветами. Таким образом, фишкам в раскрашенных сетях приписаны различные цвета. Позиции также имеют цвет, при этом позиция может содержать фишки только приписанного ей цвета.

Перейдем к формальным определениям. Предполагается, что задан некоторый язык $\mathcal{L}$ и некоторая модель $\mathcal{U}$ этого языка. При этом

- Каждому элементу a универсума приписан некоторый тип $\mathit{Type}(a)$. Множество всех элементов типа ω будем также обозначать через ω.
- Каждой переменной $v \in \mathit{Var}$ также приписан тип $\mathit{Type}(v)$.
- Тип выражения $\theta \in \mathcal{L}$ обозначается через $\mathit{Type}(\theta)$.
- Множество переменных, входящих в выражение θ, обозначается через $\mathit{Var}(\theta)$.
- При означивании некоторого множества V переменных учитывается их тип, т. е. означивание b есть функция, сопоставляющая переменной v элемент $b(v) \in \mathit{Type}(v)$.
- Значение выражения θ при означивании b обозначается $\theta[b]$. При этом требуется, чтобы $\mathit{Var}(\theta)$ было подмножеством области определения функции b. Значение $\theta[b]$ вычисляется путем подстановки вместо каждой переменной $v \in \mathit{Var}(\theta)$ ее значения $b(v)$. Напомним, что предполагается, что изначально задана модель с интерпретацией всех операций.

Определение 2.20. *Раскрашенной сетью Петри* (CP-сетью) называется набор $CPN = (\Omega, N, C, W, G, M_0)$, где

1) Ω — конечное непустое множество *цветов*.
2) $N = (P, T, F)$ — конечная сеть с множеством позиций P, множеством переходов T и отношением инцидентности F.
3) $C : P \to \Omega^*$ — *функция раскраски позиций*, сопоставляющая каждой позиции $p \in P$ ее цвет $C(p)$. В том случае, когда позиции p разметка сопоставляет кортежи элементов, $C(p)$ имеет вид $\omega_1 \times \cdots \times \omega_n$.
4) W — функция, приписывающая дугам сети N выражения языка $\mathcal{L}$ такая, что для $(p,t),(t',p') \in F$: $\mathit{Type}(W(p,t)) = \mathcal{M}(C(p))$, соответственно, $\mathit{Type}(W(t',p')) = \mathcal{M}(C(p'))$.

5) $G : T \to \mathcal{L}$ — *функция охраны*, приписывающая каждому переходу $t \in T$ некоторое выражение булевского типа.

6) M_0 — функция, сопоставляющая каждой позиции $p \in P$ мультимножество замкнутых выражений, такое, что $\forall p \in P : Type(M_0(p)) = \mathcal{M}(C(p))$. Эта функция задает *начальную разметку сети*.

Перейдем к определению поведения раскрашенной сети. Для раскрашенных сетей разрешается параллельное (синхронное) срабатывание нескольких переходов, причем допускается срабатывание перехода параллельно с самим собой.

Нам понадобятся следующие обозначения: для $t \in T$ полагаем $Var(t) = \{v \mid v \in Var(G(t)) \vee \exists p \in P : (p,t) \in F \wedge v \in Var(W(p,t)) \vee (t,p) \in F \wedge v \in Var(W(t,p))\}$

Определение 2.21. Пусть $CPN = (\Omega, N, C, W, G, M_0)$ — некоторая СР-сеть. *Означиванием перехода* $t \in T$ называется функция b с областью определения $Var(t)$ такая, что $\forall v \in Var(t) : b(v) \in Type(v)$, и $G(t)[b]$ — истинно.

Разметкой M называется функция, сопоставляющая каждой позиции некоторое мультимножество элементов, причем тип этих элементов должен совпадать с типом, приписанным позиции, т. е. $M(p) \in \mathcal{M}(Type(p))$.

Шагом сети называется непустое конечное мультимножество означенных переходов. Шаг Y является *активным* при разметке M (обозначение $M[Y\rangle$), если

$$\forall p \in P : \Sigma_{t[b] \in Y} W(p,t)[b] \leq M(p).$$

Результат срабатывания шага Y при разметке M есть разметка M' такая, что

$$\forall p \in P : M'(p) = M(p) - \Sigma_{t[b] \in Y} W(p,t)[b] + \Sigma_{t[b] \in Y} W(t,p)[b].$$

Для шага срабатывания используется обозначение $M[Y\rangle M'$.

Глава 3

Вложенные сети Петри: двухуровневые системы

3.1 Начальные примеры

Для того, чтобы дать читателю интуитивное представление о вложенных сетях Петри, приведем два небольших примера моделирования распределенных систем.

3.1.1 Пример 1: склад инструментов

Начнем с небольшого примера двухуровневой сети NPN_1, представленной на рис. 3.1. Эта сеть моделирует следующую задачу. Имеется множество исполнителей, каждый из которых может получать время от времени некоторое задание, для исполнения которого он использует специальный инструмент. Инструменты хранятся на складе (в буфере). Имеется некоторый конечный набор таких инструментов. Исполнители (клиенты) обращаются за инструментом на склад, получают его и некоторое время используют, после чего возвращают на склад. Если в момент обращения клиента за инструментом свободных инструментов нет, то он по-

лучает отказ, и через некоторое время может повторно обратиться за инструментом. Отсутствие инструмента не отменяет задание.

Количество клиентов, задействованных в системе, вообще говоря, не ограничено, хотя в каждый момент времени число их конечно. Множество A инструментов, напротив, заранее определено и конечно. В нашем примере A состоит из N элементов.

Поведение клиента описывается элементной сетью EN, которая представляет собой элементарную сеть Петри. Все клиенты описываются одной и той же сетью. Склад инструментов моделируется системной сетью SN. Сеть SN является сетью Петри высокого уровня с фишками трех типов: черные фишки (для позиций, представляющих пропозициональные условия), неразличимые между собой фишки некоторого другого цвета (представляющие инструменты) и фишки-сети (представляющие клиентов).

Содержательно позиции в элементной сети EN означают следующее:
T — наличие заданий для исполнителя;
W_0 — исполнитель свободен;
W_1 — исполнитель получил задание;
W_2 — исполнитель обращается на склад за инструментом;
W_3 — исполнитель выполняет задание;
W_3 — исполнитель закончил работу.

В системной сети SN позиции означают:
S_1 — склад открыт;
S_2 — клиенты, обратившиеся за инструментом;
S_3 — клиенты, получившие инструмент;
S_4 — количество выданных инструментов;
S_5 — свободные инструменты;
S_6 — клиенты, возвращающие инструмент.

Поскольку системная сеть SN является сетью высокого уровня, некоторые дуги в ней помечены выражениями. В нашем примере выражения представляют собой переменные или константы. Переменная x принимает значения из множества маркированных

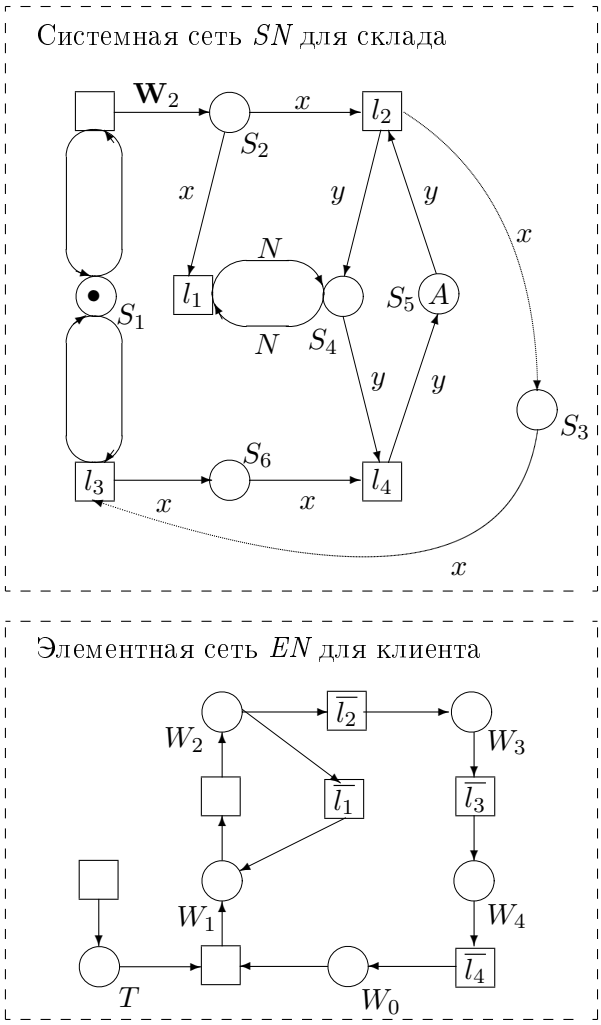


Рис. 3.1: Вложенная сеть NPN_1

элементных сетей, переменная y имеет значением цветную фишку, соответствующую инструменту, константа $\mathbf{W}_2$ означает элементную сеть EN с разметкой $\{W_2\}$, т. е. с разметкой, в которой имеется единственная фишка в позиции W_2. Если для некоторой дуги выражение не указано, то по умолчанию считается, что ей сопоставлено выражение 1, т. е. эта дуга "перемещает" одну черную фишку.

Начальная разметка сети SN (представляющей склад) показана на рис. 3.2. Здесь позиция S_1 содержит одну черную фишку (что означает истинность условия "склад открыт"), а позиция S_5 содержит мультимножество A из N одинаковых цветных фишек (что соответствует наличию на складе N свободных инструментов). Начальная разметка элементной сети EN (клиента) также показана на рисунке. В этой разметке имеется единственная черная фишка в позиции W_0 (исполнитель свободен). Позиция T первоначально фишек не содержит, что означает отсутствие заданий. Заметим, что в начальной разметке данной системной сети SN нет сетевых фишек. Таким образом, элементная сеть EN для клиента по отношению к системной сети играет роль описания типа переменных и констант, встречающихся в описании SN.

Если некоторый переход в системной или автономной сети не имеет специальных пометок, то он может сработать автономно. Автономное срабатывание перехода в системной сети выполняется по обычным правилам для сетей высокого уровня. При этом сетевые фишки рассматриваются как обычные фишки, не имеющие собственной структуры, и их разметка в результате срабатывания автономного перехода системной сети не меняется. Мы будем говорить, что сетевая фишка задействована в срабатывании некоторого перехода системной сети, если она убирается из некоторой позиции или добавляется в некоторую позицию в результате рассматриваемого срабатывания.

Элементные сети в рассматриваемом примере являются элементарными сетями Петри. В элементных сетях переходы, не име-

ющие специальных меток, также могут срабатывать автономно. Правила срабатывания для них те же, что и для элементарных сетей Петри, при этом после автономного срабатывания перехода элементная сеть (с новой разметкой) остается в прежней позиции системной сети.

Кроме того, во вложенных сетях предусмотрен механизм синхронизации срабатывания переходов в системной и элементных сетях (вертикальная синхронизация). Для этого в нашем примере некоторые переходы системной сети помечены специальными метками l_1, l_2, l_3, l_4. Переходы в элементной сети, которые должны срабатывать одновременно с переходом с меткой l в системной сети, помечены дополнительной к l меткой $\overline{l}$. Так, например, переход, помеченный меткой l_2, в сети SN может сработать только синхронно с переходом, помеченным $\overline{l_2}$, в элементной сети EN, задействованной в этом срабатывании.

Чтобы проиллюстрировать поведение вложенной сети, проследим несколько шагов исполнения сети NPN_1.

При заданной начальной разметке активным является непомеченный переход в левом верхнем углу сети SN. Его автономное срабатывание приводит к появлению сетевой фишки $\mathbf{W}_2$ в позиции S_2 системной сети (разметка M_1). Сработавший переход остается активным. Каждое его последующее срабатывание приводит к появлению в S_2 еще одного экземпляра элементной сети $\mathbf{W}_2$ (клиента).

Кроме того, после первого шага в системной сети при означивании $x := \mathbf{W}_2,\ y := a$ (где a есть атомарная фишка — элемент множества A) становится активным переход, помеченный меткой l_2. При этом задействованной в этом срабатывании оказывается сетевая фишка $\mathbf{W}_2$, которая имеет разметку $\{W_2\}$ с единственной черной фишкой в позиции W_2. В задействованной элементной сети активным является переход, помеченный $\overline{l_2}$. Оба указанных перехода могут сработать только одновременно (шаг вертикальной синхронизации), что приведет к новой разметке M_2. В разметке

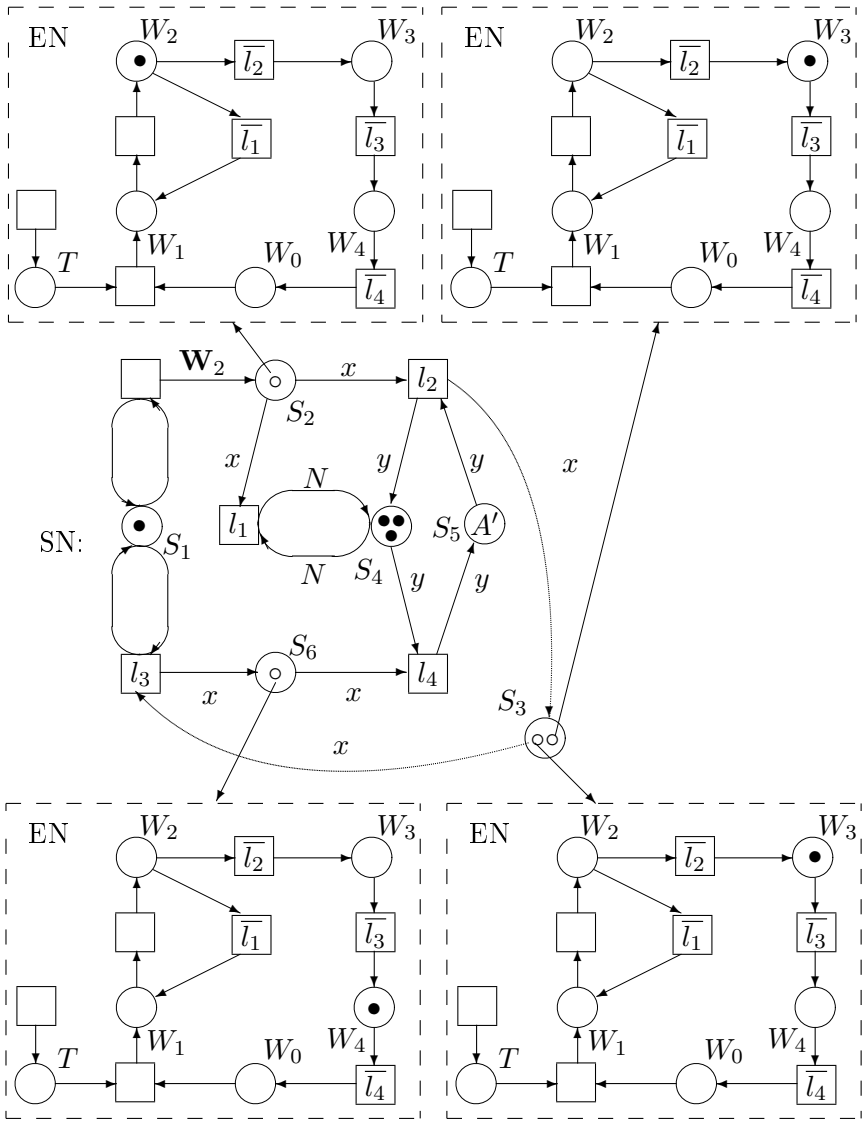


Рис. 3.2: Пример одного из достижимых состояний сети NPN_1

M_2 позиция S_5 содержит мультимножество $A - \{a\}$ атомарных фишек (т. е. на одну фишку меньше прежнего), позиция S_4 содержит одну черную атомарную фишку, позиция S_2 становится пустой, и позиция S_3 получает сетевую фишку с разметкой $\{W_3\}$. Содержимое других позиций остается прежним.

Далее в системной сети становится активным переход, помеченный l_3 (при означивании $x := \mathbf{W}_3$). Он должен сработать одновременно с переходом, помеченным $\overline{l_3}$, в задействованной в этом срабатывании элементной сети. Поскольку в сети $\mathbf{W}_3$ переход, помеченный $\overline{l_3}$, является активным, такое срабатывание возможно. Оно приводит к разметке M_3, в которой S_3 становится пустой, а S_6 получает сетевую фишку $\mathbf{W}_4$.

Продолжение этого исполнения может привести к разметке, показанной на рис. 3.2. Здесь A' означает множество A, уменьшенное на 3 атомарные фишки. Позиция S_2 содержит сетевую фишку $\mathbf{W}_2$ (клиент, обратившийся за инструментом), позиция S_3 содержит два экземпляра сетевой фишки $\mathbf{W}_3$ (клиенты, использующие инструмент), позиция S_6 содержит сетевую фишку $\mathbf{W}_4$ (клиент, желающий вернуть инструмент).

3.1.2 Пример 2: агентство по прокату автомобилей

Рассмотрим несколько более сложный пример. Будем моделировать работу агентства по прокату автомобилей. Основное отличие от предыдущего примера состоит в том, что автомобиль может находиться в разных состояниях и будет моделироваться сетью. Кроме того, клиент и автомобиль могут взаимодействовать между собой, не вовлекая в это агентство. Для моделирования таких взаимодействий вводится механизм горизонтальной синхронизации.

Пример с моделированием работы агентства по прокату автомобилей уже использовался в литературе, как модельный пример для сетей Петри. В параграфе 2.1 главы 1 мы приводили пример из [85] моделирования такой системы элементарной сетью

Петри. Этот же пример был рассмотрен Лакосом в [59] для иллюстрации формализма иерархических сетей Петри. Сеть Лакоса структурно не отличается от сети Райсига, все различия заключаются в способе моделирования самой сети. Понятие иерархии использовано только в процессе построения иерархической сети и не влияет на определение или поведение сети уже построенной. В процессе моделирования иерархическая сеть была разбита на три уровня: уровень клиента, уровень агентства и уровень взаимодействия клиента с агентством. После этого клиент и агентство представляются как позиции некоторой сети, а взаимодействие между ними — как переход. Далее эти позиции конкретизируются: каждая из позиций "клиент" и "агентство" рассматривается, в свою очередь, как сеть Петри со своими собственными позициями и переходами. Таким образом, иерархическая сеть строится путем последовательных уточнений. После этого понятие иерархии отбрасывается и поведение сети понимается как поведение элементарной сети Петри.

Вложенные сети Петри позволяют построить модель, в которой агентство взаимодействует с множеством клиентов и множеством автомобилей. Количество клиентов при этом потенциально не ограничено. Более того, модель учитывает, что клиент и автомобиль могут совершать автономные действия и могут взаимодействовать между собой без посредства агентства.

Вложенная сеть NPN_1, моделирующая работу агентства (см. рис. 3.3) состоит из системной сети SN и двух элементных сетей EN_1 и EN_2, представляющих соответственно клиентов и автомобили.

Для того, чтобы учитывать, какой именно автомобиль получил конкретный клиент, удобно использовать в качестве фишек пары (в общем случае — кортежи) элементов, как это принято в сетях Петри высокого уровня. В нашем примере мы будем рассматривать пары (клиент, автомобиль). Соответственно, сеть SN будет содержать фишки четырех типов: атомарные черные фиш-

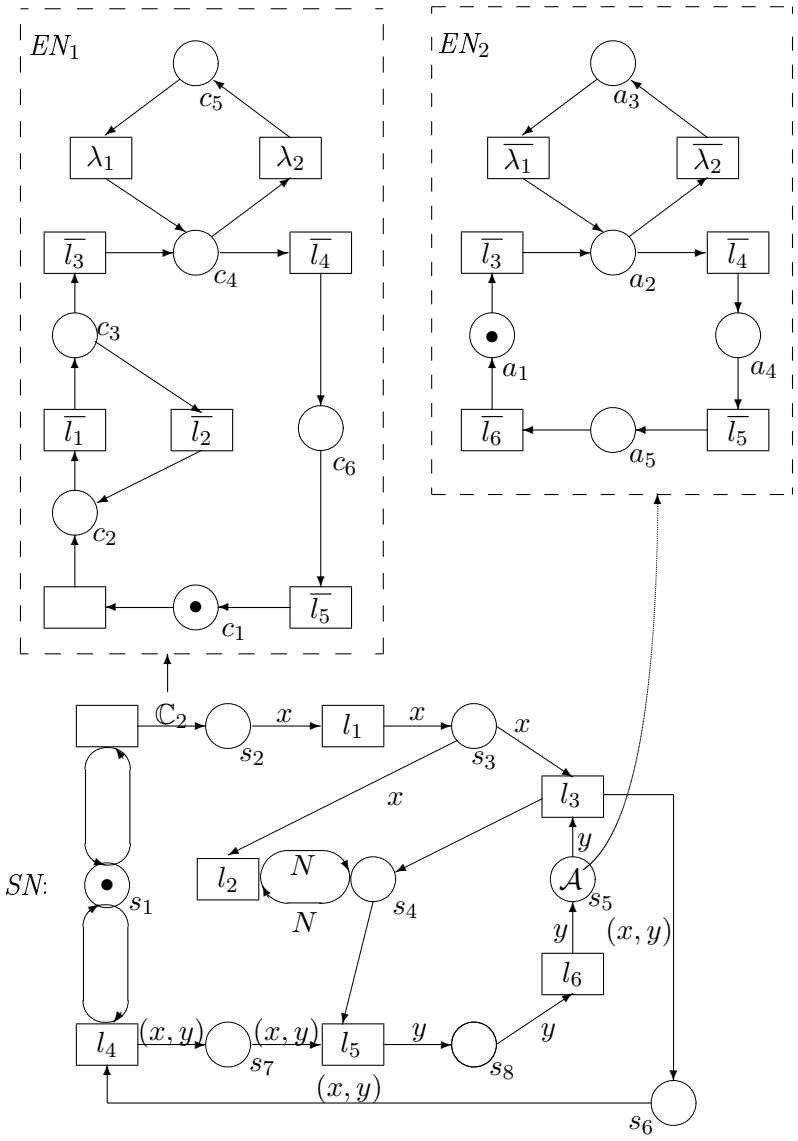


Рис. 3.3: Вложенная сеть Петри NPN_2

ки, сетевые фишки — автомобили, сетевые фишки — клиенты и пары маркированных сетей (клиент, автомобиль).

В сети NPN_1 в качестве выражений, приписанных дугам, используются переменные, константы и пары переменных. Как и в предыдущем примере, если выражение для дуги не указано, оно предполагается равным 1 (дуга переносит одну черную фишку). В рассматриваемом примере x — переменная для клиента (маркированная сеть EN_1), y — для автомобиля (маркированная сеть EN_2), (x, y) представляет пару двух маркированных сетей, N используется как обозначение для N экземпляров черных атомарных фишек (представляющих количество свободных автомобилей), $\mathbb{C}_2$ — константа для сети EN_1 с разметкой $\{c_2\}$, т. е. с единственной черной фишкой в позиции c_2.

В сетях высокого уровня каждой позиции приписывается арность. В нашем примере арность всех позиций, кроме s_6 и s_7, равна 1. Арность позиций s_6, s_7 равна двум. В системной сети SN фишками в позициях s_1, s_4 будут черные атомарные фишки, в позициях s_5, s_8 — сетевые фишки – автомобили, в позициях s_2, s_3 — сетевые фишки – клиенты, в позициях s_6, s_7 — пары сетевых фишек (клиент, автомобиль).

Приведем содержательное значение позиций и переходов сети NPN_2. Для переходов в скобках указаны сети, участвующие в срабатывании перехода.

Позиции:

s_2 — клиенты, желающие арендовать автомобиль;

s_3 — клиенты, обратившиеся в агентство за автомобилем;

s_4 — количество арендованных автомобилей;

s_5 — свободные автомобили;

s_6 — клиенты вместе с арендованными автомобилями;

s_7 — клиенты с автомобилями, которые они хотят вернуть в агентство;

s_8 — автомобили в мойке;

c_1 — клиент не нуждается в автомобиле;

c_2 — клиент хочет арендовать автомобиль;
c_3 — клиент спрашивает автомобиль в агентстве;
c_4 — клиент использует автомобиль;
c_5 — клиент желает вернуть автомобиль в агентство;
c_6 — клиент обратился в агентство, чтобы вернуть автомобиль;
a_1 — автомобиль в гараже агентства;
a_2 — автомобиль арендован;
a_3 — автомобиль используется;
a_4 — автомобиль находится в мойке.

Переходы:
l_1 — клиент спрашивает в агентстве автомобиль (агентство, клиент);
l_2 — отказ в аренде (агентство, клиент);
l_3 — передача автомобиля клиенту (агентство, клиент, автомобиль);
l_4 — обращение в агентство для возврата автомобиля (агентство, клиент, автомобиль);
l_5 — возвращение автомобиля в агентство (агентство, клиент, автомобиль);
l_6 — возвращение автомобиля в гараж агентства (агентство, автомобиль).

Начальная разметка M_0 сети NPN_2 представлена на рис. 3.3. В M_0 системная сеть содержит атомарную черную фишку в позиции s_1 (агентство открыто) и мультимножество $\mathcal{A}$, состоящее из N экземпляров сетевой фишки EN_2 (с начальной разметкой), в позиции s_5 (множество свободных автомобилей в гараже агентства). На рисунке стрелка из позиции s_5 к изображению элементной сети EN_2 представляет пример сетевой фишки EN_2 из начальной разметки сети SN. Элементные сети EN_1 в начальной разметке системной сети отсутствуют (ни один клиент еще не обратился в агентство) и играют роль описания типа (в частности, переменной x). Начальные разметки сетей EN_1 и EN_2 (для клиента и автомобиля) также указаны на рисунке. EN_1 содержит единственную

черную фишку в позиции c_1 (клиент не нуждается в автомобиле). EN_2 содержит единственную черную фишку в позиции a_1 (автомобиль находится в гараже агентства).

В данном примере используются два вида синхронизации: вертикальная, когда переход в системной сети срабатывает одновременно с некоторыми переходами в элементных сетях, и горизонтальная, когда одновременно срабатывают два перехода в элементных сетях (находящихся в одной и той же позиции системной сети). Для их различения используются два вида меток: метки $l, \bar{l}, \ldots$ — для вертикальной синхронизации и метки $\lambda, \overline{\lambda}, \ldots$ — для горизонтальной. Переходы, срабатывающие одновременно, помечаются взаимно дополнительными метками, например, l и $\bar{l}$.

Проследим несколько шагов исполнения сети NPN_2. При начальной разметке активным является автономный (непомеченный) переход t_1 в системной сети SN. Его автономное срабатывание приводит к появлению в позиции s_2 сетевой фишки $\mathbb{C}_2$ (элементной сети EN_1 с разметкой $\{c_2\}$). После этого переход t_1 остается активным, и каждое его очередное срабатывание приводит к появлению еще одного экземпляра элементной сети EN_1 в качестве сетевой фишки сети SN.

Кроме того, уже после первого срабатывания перехода t_1 в системной сети SN при означивании $x := \mathbb{C}_2$ переход, помеченный меткой l_1, становится активным. Задействованной в этом срабатывании оказывается элементная сеть EN_1, в которой, соответственно, активен переход, помеченный меткой $\overline{l_1}$. Синхронное срабатывание (вертикальная синхронизация) этих двух переходов приводит к перемещению сетевой фишки в позицию s_3 и изменению разметки этой фишки на $\{c_3\}$.

Следующий шаг состоит в срабатывании l_3 в SN, в результате чего "забираются" автомобиль из s_5 и клиент из s_3, а в позицию s_6 "помещается" пара из этих двух сетевых фишек. При этом происходит одновременное изменение собственных разметок в двух задействованных сетях. В каждой из них синхронно с l_3 сраба-

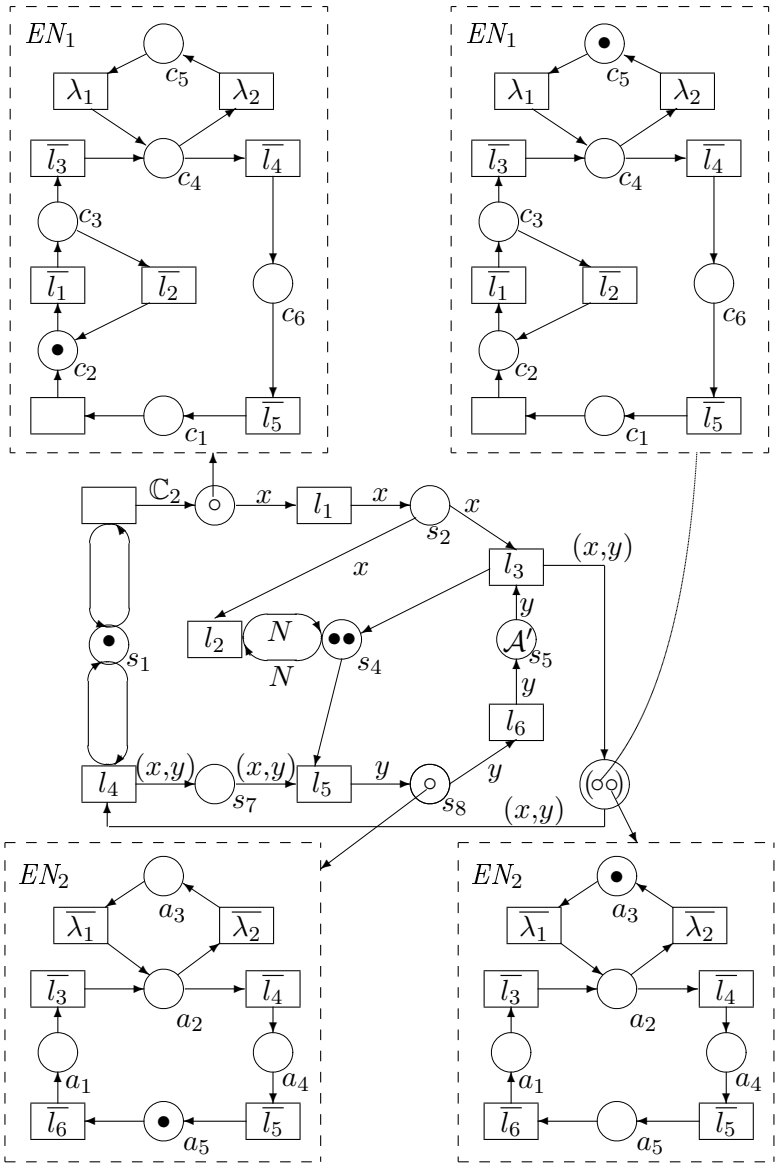


Рис. 3.4: Пример достижимого состояния для NPN_2

тывает по одному переходу, помеченному $\overline{l_3}$. Таким образом, в результате этого шага (вертикальной синхронизации) EN_1 содержит черную фишку в позиции c_4, EN_2 — в позиции a_2, в то время как SN содержит пару этих элементных сетей в s_5.

После этого шага активными в элементных сетях (находящихся в s_5) оказываются переходы, помеченные соответственно метками λ_1 в EN_1 и $\overline{\lambda_1}$ в EN_2. Эти переходы могут сработать только одновременно (горизонтальная синхронизация). После их срабатывания меняются разметки в элементных сетях EN_1 и EN_2, при этом обе сетевые фишки остаются в той же позиции s_5 системной сети. Мы предполагаем, что при горизонтальной синхронизации переходов выполняется следующее правило локализации:

Переход (помеченный меткой λ_1) в элементной сети EN срабатывает синхронно с переходом (помеченным $\overline{\lambda_1}$) в элементной сети EN', входящей в тот же кортеж (пару), что и EN. Если же арность позиции, в которой находится сетевая фишка, была бы равна 1, то соответствующий переход (помеченный дополнительной меткой) нужно было искать в другой сетевой фишке, находящейся в той же позиции системной сети. Горизонтальная синхронизация переходов в сетях, находящихся в разных позициях системной сети, запрещена.

На рисунке 3.4 показан пример достижимой разметки для вложенной сети NPN_1. Эта разметка соответствует состоянию системы, в которой один клиент использует арендованный автомобиль, другой обратился в агентство, чтобы взять автомобиль в аренду. Один автомобиль в это время находится в мойке, и $\mathcal{A}'$ есть множество $\mathcal{A}$ автомобилей агентства за вычетом двух автомобилей (один эксплуатируется клиентом, и второй — в мойке). Элементные сети, соответствующие автомобилям в гараже (позиция s_5), не показаны на этом рисунке из-за ограниченности места.

Отметим, что множество потенциальных клиентов в этой модели не ограничено, и, таким образом, не ограничено число сетевых фишек в системной сети. Напротив, множество автомоби-

лей агентства фиксировано и конечно. Таким образом, формализм вложенных сетей позволяет моделировать оба указанных случая.

Заметим также, что построенная модель может быть легко обобщена на случай, когда имеется система с несколькими агентствами и клиент может обращаться в любое из них. Для этого системная сеть должна состоять из нескольких экземпляров сети SN (нескольких одинаковых компонент связности графа), и все правила взаимодействия остаются прежними.

3.2 Структура сети

В этом разделе мы приведем формальное определение вложенных сетей Петри с двумя уровнями вложенности.

Пусть $Var = \{v, \ldots\}$ — множество имен *переменных*, $Con = Con_{atom} \cup Con_{net} = \{c, \ldots\}$ — множество имен *констант*, состоящее из множества Con_{atom} *атомарных констант* и множества Con_{net} *сетевых констант*. Через $Atom$ будем обозначать множество $Var \cup Con$ *атомов*.

Язык выражений $Expr(Atom)$ над множеством атомов $Atom$ определим следующим образом:

1) Атом $atom \in Atom$ является выражением в $Expr(Atom)$. Размерность a полагаем равной 1.

2) Если $atom_1, \ldots, atom_n \in Atom$, то кортеж $(atom_1, \ldots, atom_n)$ является выражением размерности n в $Expr(Atom)$.

3) Если $e_1, e_2 \in Expr(Atom)$ — выражения одинаковой размерности n, то $(e_1 + e_2)$ является выражением размерности n в $Expr(Atom)$.

Пусть $e \in Expr(Atom)$, через $Var(e)$ будем обозначать множество переменных, входящих в e.

Вместо $v+v$ будем писать $2v$, вместо $((v_1+v_2)+v_3) - v_1+v_2+v_3$. Соответственно, $k \cdot e$ означает $\underbrace{e+e+\cdots+e}_{k \text{ раз}}$.

Далее в определении двухуровневых NP-сетей константы из Con_{net} в выражениях языка $Expr(Atom)$ будут интерпретироваться как маркированные обыкновенные сети Петри (сетевые фишки), константы из Con_{atom} — как индивидуальные фишки, не имеющие собственной структуры (атомарные фишки). Множество сетевых фишек будет обозначаться через A_{net}, множество атомарных фишек — через A_{atom}. Мы предполагаем, что $\mathrm{A}_{atom} \neq \emptyset$. Фиксируем выделенный элемент в множестве A_{atom}, который на рисунке сети будет обозначаться черной точкой, в языке $Expr(Atom)$ ему будет соответствовать константа 1. Вместо выражения $n \cdot 1$ для обозначения мультимножества, состоящего из n экземпляров черной точки, будем писать просто n. Выражение $(a_1, a_2, \ldots, a_n)$ означает кортеж из элементов $a_1, a_2, \ldots, a_n$, а операция "+" в выражениях из $e \in Expr(A)$ интерпретируется как объединение мультимножеств. Таким образом, значениями выражений языка $Expr(Atom)$ будут мультимножества кортежей элементов, где элементы суть либо маркированные обыкновенные сети Петри, либо атомарные фишки.

Пусть далее $Lab_v = \{l_1, l_2, \ldots\}$ и $Lab_h = \{\lambda_1, \lambda_2, \ldots\}$ — два непересекающихся множества меток. Мы предполагаем, что для каждой метки $l \in Lab_v$ и каждой метки $\lambda \in Lab_h$ имеется *двойственная* метка $\overline{l} \in Lab_v$, соответственно, $\overline{\lambda} \in Lab_h$, такие что

1) $\overline{l_1} \neq \overline{l_2}$ для $l_1, l_2 \in Lab_v$, $l_1 \neq l_2$; $\overline{\lambda_1} \neq \overline{\lambda_2}$ для $\lambda_1, \lambda_2 \in Lab_h$, $l_1 \neq l_2$;

2) $\overline{\overline{l}} =_{\text{def}} l$; $\overline{\overline{\lambda}} =_{\text{def}} \lambda$.

Метки множества Lab_v будут использоваться для вертикальной синхронизации переходов, метки множества Lab_h — для горизонтальной синхронизации. Полагаем $Lab =_{\text{def}} Lab_v \cup Lab_h$.

Определение 3.1. Двухуровневой вложенной сетью Петри (NP-сетью) называется набор
$NPN = (Atom, Lab, SN, (EN_1, \ldots, EN_k), \Lambda)$, где

1) $Atom = Var \cup Con$ — множество атомов.

2) $Lab = Lab_v \cup Lab_h$ — множество меток, удовлетворяющих приведенным выше условиям.

3) $(EN_1, \ldots, EN_k)$ $(k \geq 1)$ — конечный набор обыкновенных сетей Петри. Сети $EN_1, \ldots, EN_k$ называются элементными сетями вложенной сети NPN.

4) $SN = (N, \mathcal{L}, \mathcal{U}, W, M_0)$ — сеть Петри высокого уровня, где

   - $N = (P, T, F)$ — сеть.
   - P — множество символов (позиций сети) с приписанной им арностью, $P \cap Atom = \emptyset$.
   - $\mathcal{L} = Expr(Atom)$ — язык выражений, определенный выше.
   - $\mathcal{U} = (\mathrm{A}, \mathcal{I})$ — модель языка $\mathcal{L}$, где $\mathrm{A} = \mathrm{A}_{net} \cup \mathrm{A}_{atom}$ и $\mathrm{A}_{net} = \{(EN, m) \mid \exists i = 1, \ldots, k : EN = EN_i, m \in \mathfrak{M}(EN_i)\}$, т. е. A_{net} есть множество маркированных элементных сетей. Множество A_{net} будем называть также множеством сетевых фишек сети SN. Множество A_{atom} есть конечное множество атомарных фишек. Как уже говорилось выше, A_{atom} содержит хотя бы один элемент – черную точку.

     Функция $\mathcal{I} : Con \to \mathrm{A}$ задает некоторую интерпретацию констант языка $\mathcal{L}$. Интерпретация операции “+” была определена выше.
   - W — функция, сопоставляющая каждой дуге $(x, y) \in F$ некоторое выражение $W(x, y) = (\theta_1, \ldots, \theta_n)$ размерности n, где $\theta_i \in \mathcal{L}$ $(1 \leq i \leq n)$ и n есть арность позиции,

инцидентной дуге (x, y). При этом на выражения, приписанные входным дугам, накладываются следующие ограничения:

- эти выражения не содержат констант из Con_{net} (т. е. только константы из Con_{atom});
- каждая переменная встречается в выражении $W(p,t)$ не более одного раза;
- для любых двух приписанных входным дугам одного и того же перехода t выражений $W(p_1,t)$ и $W(p_2,t)$ выполняется:
  $Var(W(p_1,t)) \cap Var(W(p_2,t)) = \emptyset$.

Ограничения на выражения на входных дугах показаны на рис.3.5.

На выражения, приписанные выходным дугам, накладывается следующее ограничение: выходное выражение для перехода t может содержать только переменные, которые входят в одно из входных выражений для этого же перехода.

Если для некоторой дуги выражение опущено, то по умолчанию предполагается, что это $1 \in \mathbb{N}$.

- M_0 — выделенная разметка, называемая начальной разметкой сети, где вообще разметка есть отображение M, которое сопоставляет каждой позиции $p \in P$ арности n некоторое мультимножество $M(p)$ над множеством A^n кортежей размерности n.

5) Λ — частичная функция пометки переходов, помечающая некоторые переходы системной сети метками из Lab_v и некоторые переходы в элементных сетях метками из $Lab_v \cup Lab_h$.

Таким образом, двухуровневая вложенная сеть Петри состоит из набора обыкновенных сетей Петри, задающих структуру сетевых фишек, и системной сети. Системная сеть представляет

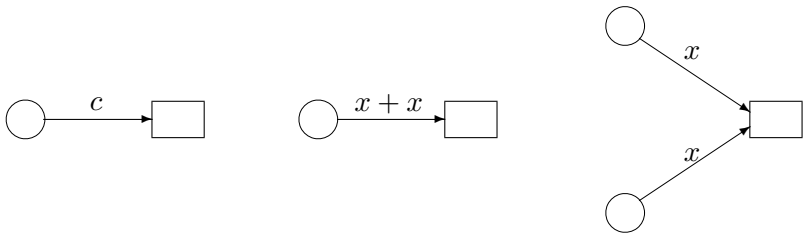


Рис. 3.5: Примеры запрещенных выражений на входных дугах. Здесь $x \in Var, c \in Con_{net}$.

собой предикатную сеть Петри с довольно ограниченным языком выражений, которые могут быть приписаны дугам. Охраны при переходах системной сети отсутствуют (т. е. истинны при любых означиваниях переменных). Модель языка $\mathcal{U}$ определяется элементными сетями. Носитель модели $\mathcal{U}$ наряду с обычными для предикатных сетей индивидуальными элементами содержит сетевые элементы — маркированные элементные сети. Кроме того, некоторые переходы в системной и элементных сетях помечены специальными метками для обеспечения механизма синхронизации этих переходов. Далее автономные срабатывания переходов в системной и элементных сетях будут определены по правилам срабатывания для предикатных и обыкновенных сетей соответственно. Синхронные срабатывания определяются специальными правилами.

3.3 Поведение сети

Пусть $NPN = (Atom, Lab, SN, (EN_1, \ldots, EN_k), \Lambda)$ — двухуровневая NP-сеть.

Как уже говорилось, разметка сети NPN есть отображение M, которое сопоставляет каждой позиции $p \in P$ системной сети SN некоторое (возможно пустое) мультимножество $M(p)$ над множе-

ством A^n кортежей фишек, где n — арность позиции p. Таким образом, разметка вложенной сети определяется как разметка ее системной сети. Будем писать $M(p) = 0$, если $M(p) = \emptyset$, т. е. позиция p не содержит фишек. Множество всех разметок сети NPN будем обозначать через $\mathfrak{M}(NPN)$.

Пусть t — некоторый переход системной сети SN, ${}^\bullet t = \{p_1, \ldots, p_i\}$, $t^\bullet = \{q_1, \ldots, q_j\}$ — множества его пред- и постусловий. Через $W(t) = \{W(p_1, t), \ldots, W(p_i, t), W(t, q_1), \ldots, W(t, q_j)\}$ обозначим множество всех выражений, приписанных инцидентным t дугам.

Означиванием перехода t называется функция b, приписывающая каждой переменной v, входящей в некоторое выражение из $W(t)$, значение $b(v)$ из множества $\mathrm{A}_{atom} \cup \mathrm{A}_{net}$. Очевидно, что при заданном означивании перехода t можно вычислить значение $\theta(b) = b(\theta)$ любого выражения $\theta \in W(t)$, поскольку интерпретация операции "+" известна.

Переход t сети SN является *активным* при некоторой разметке M и означивании b в том и только том случае, если $\forall p \in {}^\bullet t : W(p, t)(b) \subseteq M(p)$, т. е. всякая входная для t позиция p содержит мультимножество, являющееся значением приписанного соответствующей входной дуге выражения $W(p, t)$.

При этих условиях срабатывание перехода t порождает новую разметку M', (обозначается $M \xrightarrow{t[b]} M'$), такую что для всех позиций p, $M'(p) = (M(p) \setminus W(p, t)(b)) \cup W(t, p)(b)$.

Сетевые фишки из A_{net}, являющиеся значениями переменных во входных выражениях из $W(t)$ при означивании перехода t, будем называть элементными сетями, *задействованными* при срабатывании t. Эти сети удаляются из входных позиций при срабатывании перехода t (возможно, в составе некоторых кортежей).

Определение 3.2. Для NP-сети NPN определяются следующие четыре вида *шагов срабатывания*:

- **системно-автономный шаг:** Пусть t — непомеченный пе-

реход системной сети SN (т. е. значение $\Lambda(t)$ не определено). Если t является активным при разметке M и означивании b и $M \xrightarrow{t} M'$, то такое срабатывание перехода t в сети SN называется системно-автономным шагом вложенной сети NPN и обозначается $M[t[b]\rangle M'$ или просто $M[\ \rangle M'$. Легко видеть, что при таком срабатывании разметки сетевых фишек не меняются, но некоторые из них перемещаются из одной позиции в другую (копируются, исчезают), а также возможно появление новых сетевых фишек.

- **элементно-автономный шаг:** Пусть M — разметка сети NPN, $p \in P$ — некоторая позиция системной сети SN и $(\alpha_1, \ldots, \alpha_n) \in M(p)$ — кортеж фишек, находящийся в позиции p при разметке M. Пусть далее $\alpha_i = (EN, m)$ — сетевая фишка в этом кортеже. Пусть в обыкновенной сети Петри EN имеется переход t, такой что значение $\Lambda(t)$ не определено и $m[t\rangle m'$, т. е. t — непомеченный активный при разметке m переход, и его срабатывание по правилам для обыкновенных сетей Петри приводит к разметке m'. Пусть M' — разметка сети NPN, получающаяся из M заменой сетевой фишки $\alpha_i = (EN, m)$ на фишку $\alpha_i' = (EN, m')$, т. е. M' — разметка, получающаяся в результате локального срабатывания перехода t в сетевой фишке α_i, при этом сеть EN остается в той же позиции системной сети SN. Тогда такое срабатывание перехода t в сети EN называется элементно-автономным шагом вложенной сети NPN и обозначается $M[t\rangle M'$ или просто $M[\ \rangle M'$.

- **шаг горизонтальной синхронизации:** Пусть M — разметка сети NPN, $p \in P$ — позиция системной сети SN и $(\alpha_1, \ldots, \alpha_n) \in M(p)$ — кортеж фишек, находящийся в позиции p при разметке M. Пусть далее $\alpha_i = (EN_1, m_1), \alpha_j = (EN_2, m_2)$ — две сетевые фишки в этом кортеже. Пусть в обыкновенной сети Петри EN_1 имеется переход t_1, такой что

$\Lambda(t_1) = \lambda \in Lab_h$ и $m_1[t_1\rangle m_1'$, т. е. t_1 — активный при разметке m_1 переход и его срабатывание по правилам для обыкновенных сетей Петри приводит к разметке m_1'. Пусть при этом в сети EN_2 имеется переход t_2, такой что $\Lambda(t_2) = \overline{\lambda} \in Lab_h$ и $m_2[t_2\rangle m_2'$. Пусть далее M' есть разметка сети NPN, получающаяся из M заменой сетевой фишки $\alpha_i = (EN_1, m_1)$ на фишку $\alpha_i' = (EN_1, m_1')$ и заменой фишки $\alpha_j = (EN_2, m_2)$ на фишку $\alpha_j' = (EN_2, m_2')$, т. е. M' — разметка, получающаяся в результате одновременного локального срабатывания двух переходов t_1 и t_2 в сетевых фишках α_i и α_j соответственно, при этом сети EN_1 и EN_2 остаются в той же позиции системной сети SN. Такое синхронное срабатывание переходов t_1 и t_2 в сетях EN_1 и EN_2 называется шагом горизонтальной синхронизации для вложенной сети NPN и обозначается $M[t_1; t_2\rangle M'$ или просто $M[\,\rangle M'$.

Если арность позиции p равна 1 и p содержит не кортежи, а единичные фишки, то для сетевой фишки $\alpha = (EN_1, m_1)$ парная ей сетевая фишка с активным переходом, помеченным дополнительной меткой $\overline{\lambda}$, выбирается среди фишек, находящихся в той же позиции p при разметке M. Выбор пары фишек из двух разных позиций сети при этом запрещается.

- **шаг вертикальной синхронизации**: Пусть t — переход системной сети SN, $\Lambda(t) = l \in Lab_v$, t является активным при разметке M и означивании b и $M \xrightarrow{t} M'$. Пусть далее $\{\alpha_1, \ldots, \alpha_k\} \subset \mathrm{A}_{net}$ есть множество задействованных в этом срабатывании перехода t сетевых фишек, где $\alpha_1 = (EN_1, m_1), \ldots, \alpha_k = (EN_k, m_k)$, и пусть для каждого $1 \leq i \leq k$ в сети EN_i найдется переход t_i, такой что t_i активен в EN_i при разметке m_i, $m_i[t_i\rangle m_i'$ (в соответствии с правилами срабатывания для обычных сетей Петри) и $\Lambda(t_i) = \overline{l} \in Lab_v$.

  Пусть теперь W' есть мультимножество кортежей элемен-

тов, полученное из множества $W(t,p)(b)$ значений всех выражений на выходных дугах перехода t сети NPN при означивании b путем замены сетевой фишки $\alpha_i = (EN_i, m_i)$ на фишку $\alpha_i' = (EN_i, m_i')$ для всех $1 \le i \le k$.

Тогда результатом одновременного срабатывания перехода t в системной сети SN и дополнительных к нему переходов в задействованном в этом срабатывании элементных сетях является состояние $M'(p) = (M(p) \setminus W(p,t)(b)) + W'$.

Такое синхронное срабатывание перехода t (при означивании b) в системной сети и переходов $t_1, \ldots, t_k$ в элементных сетях $\alpha_1, \ldots, \alpha_k$ называется шагом вертикальной синхронизации для сети NPN и обозначается $M[t[b]; t_1, \ldots, t_k\rangle M'$ или просто $M[\,\rangle M'$.

Исполнением для вложенной сети NPN назовем конечную или бесконечную последовательность шагов $M_0[\,\rangle M_1[\,\rangle \ldots$, где M_0 есть начальная разметка сети NPN. Разметку M сети NPN назовем достижимой, если существует исполнение $M_0[\,\rangle M_1[\,\rangle \ldots [\,\rangle M_k$, где $M = M_k$.

Замечание 3.1 *(Вложенные сети с типами).* При использовании NP-сетей для моделирования прикладных систем может оказаться более удобным использование типизированного варианта вложенных сетей Петри. Практически такой вариант использовался при рассмотрении примера моделирования системы проката автомобилей. В этом случае сетевые фишки одной и той же сетевой структуры рассматриваются как объекты одного типа τ_i и с ними ассоциируется элементная сеть EN_i. Элементы типа τ_i — размеченные элементные сети вида (EN_i, m), где m — некоторая разметка сети EN_i.

Каждой переменной в выражениях на дугах тогда приписывается некоторый определенный тип. Так, в нашем примере x есть переменная типа τ_1, соответствующего сети EN_1 (клиент), а y — переменная типа τ_2, соответствующего сети EN_2 (автомо-

биль). Функция означивания сопоставляет переменным значения того же типа. Тип выражения определяется как обычно.

Позициям в системной сети также приписывается некоторый тип или типовое выражение. В нашем примере, в частности, позиции s_6 приписан тип (τ_1, τ_2). Фишки (кортежи фишек) в каждой позиции должны быть элементами соответствующего типа. Предполагается выполнение обычных правил согласования типов. Например, тип позиции должен совпадать с типом выражений на входных и выходных дугах, инцидентных этой позиции.

Введение типовой структуры во вложенных сетях не влияет на выразительность модели и возможности ее анализа. Однако при решении практических задач система типов улучшает ясность и наглядность модели, а также позволяет проводить предварительный анализ модели на синтаксическом уровне и сокращать перебор при семантическом анализе путем сокращения числа состояний системы за счет исключения синтаксически некорректных разметок.

3.4 Вложенные сети Петри как вполне структурированные системы переходов

Обыкновенные сети Петри являются одним из характерных примеров вполне структурированных систем переходов. Пусть $PN = (P, T, F)$ — обыкновенная сеть Петри (емкости позиций не ограничены). Упорядочим позиции сети некоторым фиксированным образом: $P = (p_1, \ldots, p_n)$. Тогда разметку M сети PN можно представить в виде n-мерного вектора $(m_1, \ldots, m_n) \in \mathbb{N}^n$, где $m_i = M(p_i)$ $(i = 1, \ldots, n)$.

Частичный порядок на разметках сети Петри определим как покоординатное сравнение $\leq^n$ (см. определение 1.3), где $\leq$ — стандартное упорядочение натуральных чисел. При сравнении разметок обыкновенных сетей Петри далее вместо $\leq^n$ будем писать

просто $\leq$.

Срабатывание переходов в обыкновенных сетях Петри обладает следующим важным *свойством монотонности*: если переход t является активным при разметке M_1 и разметка $M_2 \geq M_1$, то переход t также является активным при разметке M_2. При этом если $M_1[t\rangle M_1'$ и $M_2[t\rangle M_2'$, то $M_2' \geq M_1'$.

Свойство монотонности задает совместимость частичного порядка $\leq$ и отношения переходов $[\ \rangle$. Поскольку частичный порядок $\leq$ на множестве разметок сети является правильным квазиупорядочением (утверждение 1.5), обыкновенные сети Петри с частичным порядком $\leq$ на разметках являются вполне структурированными системами переходов.

На свойстве монотонности сетей Петри основано построение *покрывающего дерева сети* (см., например, [8]), которое используется для анализа некоторых свойств сетей Петри.

В этом разделе мы покажем, что двухуровневые вложенные сети Петри также являются вполне структурированными системами переходов относительно некоторого естественного упорядочения разметок сети.

3.4.1 Представление разметок вложенной сети

Пусть $NPN = (Atom, Lab, SN, (EN_1, \ldots, EN_k), \Lambda)$ — двухуровневая NP-сеть, $SN = (N, \mathcal{L}, \mathcal{U}, W, M_0)$ и $M : P_{SN} \to \mathcal{M}(\mathrm{A}^*)$ — разметка сети NPN.

Если считать, что множество P позиций сети строго упорядочено некоторым фиксированным способом, т. е. $P = (p_1, \ldots, p_k)$, то разметку M этой сети можно представить в виде вектора $M = (m_1, \ldots, m_k)$, где $m_i = M(p_i), i = 1, \ldots, k$. Разметка обыкновенной сети Петри записывается тогда в виде вектора из целых неотрицательных чисел, так что

$$m_i = \begin{cases} n, & \text{если } p_i \text{ содержит } n \text{ черных фишек,} \\ 0, & \text{если } p_i \text{ фишек не содержит.} \end{cases}$$

Для вложенной сети компоненты вектора разметки сами являются структурированными объектами. Поэтому разметку двухуровневой вложенной сети удобно представить в виде помеченного корневого дерева, которое мы будем называть *деревом разметки*. Дерево, соответствующее разметке M, будем обозначать через $\mathcal{T}(M)$.

Корень дерева разметки помечен именем системной сети SN и имеет k потомков (по числу позиций системной сети), помеченных именами позиций $p_1, \ldots, p_k$ сети SN. Поскольку имена позиций указываются явно, порядок расположения ветвей значения не имеет.

Каждая вершина p_i имеет столько потомков, сколько (кортежей) фишек содержится в p_i при разметке M. Порядок перечисления (кортежей) фишек и, следовательно, расположения соответствующих ветвей, также не существен.

Далее вершина, соответствующая кортежу фишек, помечена операцией образования кортежа $(, \ldots,)_n$ и в качестве потомков имеет последовательно пронумерованные вершины, соответствующие элементам кортежа. Каждая вершина, задающая элемент кортежа, имеет ровно одного потомка, который является корнем поддерева разметки соответствующей сетевой фишки. Потомки этой вершины помечены именами позиций элементной сети, а их потомки — именами атомарных фишек или кортежей атомарных фишек.

Мы видим, что в дереве разметки порядок расположения ветвей не существен, т. е. если два дерева изоморфны как помеченные графы, то они задают одну и ту же разметку. Таким образом, дерево разметки является коммутативным помеченным корневым деревом (в смысле определения 1.5) над множеством X, содержащим имена системной и элементных сетей (возможно, расширенные), имена атомарных фишек (также, возможно, расширенные), имена позиций системной и элементных сетей и символы операции образования кортежа. В качестве частичного порядка на множе-

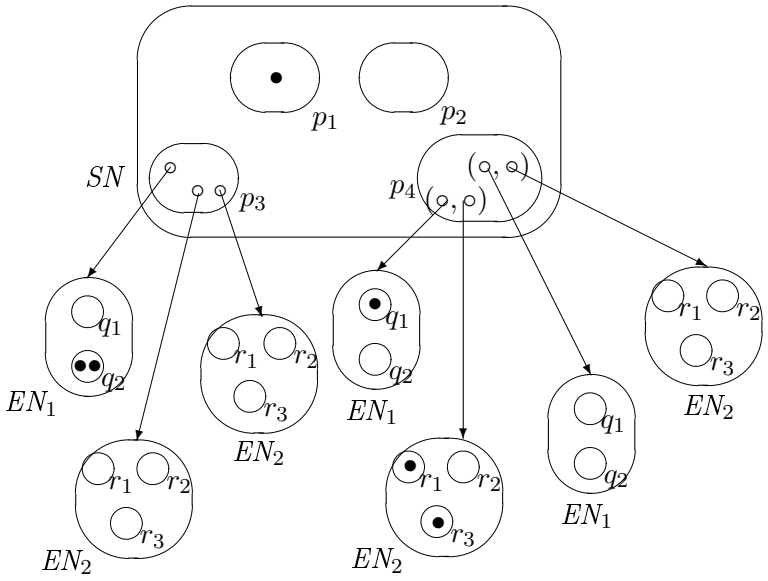


Рис. 3.6: Пример разметки M двухуровневой вложенной сети

стве X будем рассматривать отношение равенства.

На рисунке 3.6 в качестве примера приведено схематическое изображение разметки M некоторой вложенной сети NPN. Во вложенной двухуровневой сети NPN имеются две элементные сети EN_1 и EN_2, первая имеет две позиции q_1, q_2, вторая — три позиции r_1, r_2, r_3; системная сеть SN сети NPN имеет четыре позиции p_1, p_2, p_3, p_4. В разметке M первая позиция сети SN содержит одну атомарную черную фишку, вторая — пуста, третья имеет арность 1 и содержит сетевую фишку EN_1 с разметкой $(0, 2)$ и два экземпляра сетевой фишки EN_2 с пустой разметкой, четвертая позиция имеет арность 2 и содержит два кортежа длины 2. Первый кортеж состоит из сетевой фишки EN_1 с разметкой $(1, 0)$ и сетевой фишки EN_2 с разметкой $(1, 0, 1)$, второй кортеж состоит из сетевых фишек EN_1 и EN_2 с пустыми разметками. На рисунке 3.7 приведено

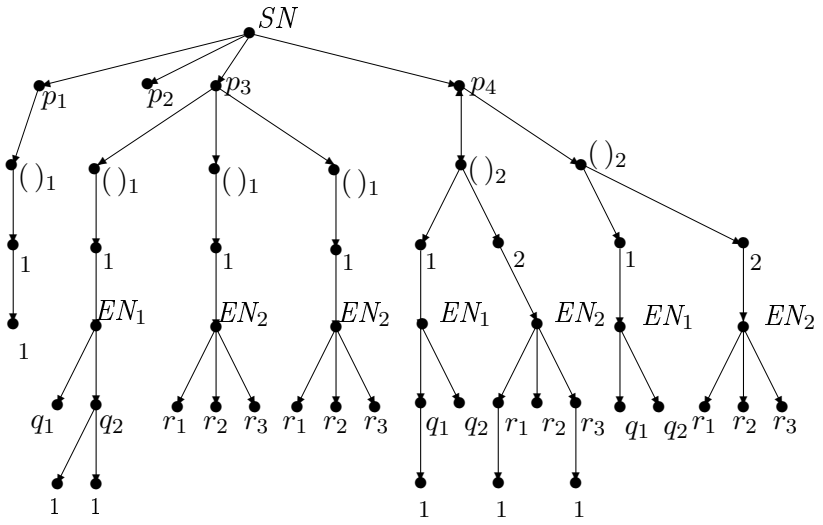


Рис. 3.7: Дерево разметки $\mathcal{T}(M)$

соответствующее дерево разметки $\mathcal{T}(M)$ для M.

Для того, чтобы иметь линейное представление разметки вложенной сети, помеченное дерево $\mathcal{T}(M)$ можно стандартным образом записать в виде (префиксного) выражения. При этом пометки на вершинах, не являющихся листьями, рассматриваются как символы операций, а поддеревья, корни которых являются потомками этих вершин, представляются выражениями — аргументами. Заметим, что для разных вхождений имен позиций в одно и то же выражение арность соответствующих символов операции может быть, вообще говоря, различной. В качестве примера приведем запись в виде выражения дерева разметки $\mathcal{T}(M)$ (см. рисунок 3.7):

$$SN(p_1(1), p_2, p_3(EN_1(q_1, q_2(1,1)), EN_2(r_1, r_2, r_3), EN_2(r_1, r_2, r_3)),$$
$$p_4((EN_1(q_1(1), q_2), EN_2(r_1(1), r_2, r_3(1))), (EN_1(q_1, q_2),$$
$$EN_2(r_1, r_2, r_3))))$$

Если каждое из множеств позиций системной и элементных се-

тей вложенной сети NPN линейно упорядочить некоторым фиксированным образом, то при записи выражения, задающего дерево разметки, можно не указывать явно имена позиций. Также, если некоторое подвыражение входит несколько раз в качестве аргумента, будем при записи просто указывать его кратность, т. е. писать, например, $2 \cdot e$ вместо e, e и 3 — вместо $1, 1, 1$. Тогда приведенное выше выражение для дерева разметки $\mathcal{T}(M)$ запишется так:

$$(1, 0, \{EN_1(0, 2), 2 \cdot EN_2(0, 0, 0)\}, \{(EN_1(1, 0), EN_2(1, 0, 1)), (EN_1(0, 0), EN_2(0, 0, 0))\}).$$

3.4.2 Частичный порядок на множестве разметок сети

Как уже говорилось выше, на множестве разметок обыкновенной сети Петри PN можно определить естественное отношение частичного порядка. Для разметок m_1, m_2 сети PN выполняется $m_1 \leq m_2$, если $\forall p \in P_{PN} : m_1(p) \leq m_2(p)$.

Определим бинарное отношение $\preceq$ на множестве разметок для двухуровневой вложенной сети Петри NPN. Для этого нам понадобятся некоторые предварительные определения.

Для двух мультимножеств $A, B \in \mathcal{M}(X)$ под *инъективным отображением* $\imath : A \to B$ будем понимать отображение, которое различным экземплярам элементов A сопоставляет различные экземпляры элементов B.

Более строго, пусть $A : X \to \mathbb{N}$ — некоторое мультимножество. Мультимножеству A сопоставим множество $\tilde{A} \subset (A \times (\mathbb{N} \setminus \{0\}))$ пар, состоящих из элементов A и натуральных чисел. Пусть $x \in A$ и $A(x) = k$, т. е. элемент x входит в A с кратностью k. Пусть $\tilde{x} =_{\text{def}} \{(x, 1), (x, 2), \ldots, (x, k)\}$ — множество, кодирующее k экземпляров элемента x. Тогда полагаем $\tilde{A} =_{\text{def}} \bigcup_{x \in X} \tilde{x}$. Таким образом, соответствие $A \leftrightarrow \tilde{A}$ задает кодирование мультимножеств, которое позволяет различать кратные вхождения одного и того

же элемента в мультимножество.

Пусть теперь $A : X \to \mathbb{N}, B : Y \to \mathbb{N}$ — два мультимножества. Будем говорить, что задано отображение $f : A \to B$, если для некоторого кодирования множеств A и B задано отображение $\tilde{f} : \tilde{A} \to \tilde{B}$. Будем писать $f(x) = y$, если для некоторых $n, m \in \mathbb{N}$ выполняется $\tilde{f}(x, n) = (y, m)$. Таким образом, отображение $f : A \to B$ может сопоставлять разным экземплярам одного и того же элемента $x \in A$ разные значения. Отображение $f : A \to B$ будем называть инъективным, если ему соответствует инъективное отображение $\tilde{f} : \tilde{A} \to \tilde{B}$ при некотором кодировании множеств A и B.

Определение 3.3. Пусть M_1, M_2 — состояния сети *NPN*. Полагаем $M_1 \preceq M_2$, если для любой позиции $p \in P_{SN}$, для которой $M_1(p) \neq \emptyset$, существует инъективное отображение $\jmath_p : M_1(p) \to M_2(p)$, такое что для $\overline{\alpha} = (\alpha_1, \ldots, \alpha_n) \in M_1(p)$ и $\jmath_p(\overline{\alpha}) = \overline{\alpha}\prime = (\alpha'_1, \ldots, \alpha'_n)$ для всех $i = 1, \ldots, n$ выполняется либо $\alpha_i = \alpha'_i$, либо $\alpha_i = (EN, m)$ есть сетевая фишка и $\alpha'_i = (EN, m')$, где $m \leq m'$.

Другими словами, для каждой фишки α разметки M_1, находящейся в позиции p, найдется соответствующая ей фишка α' разметки M_2 в той же позиции p, такая что если α является атомарной фишкой, то α' совпадает с α, если же α — сетевая фишка, то α' имеет такую же или большую, чем α, разметку.

На рисунке 3.8 приведена разметка M', связанная отношением $\preceq$ с разметкой M на рисунке 3.6. Имеем $M' \preceq M$. Действительно, $M'(p_1) = M'(p_2) = \emptyset$ и функции $\jmath_{p_1}, \jmath_{p_2}$ имеют пустую область определения. Функция $\jmath_{p_3}$ — тождественная. Функция $\jmath_{p_4}$ сопоставляет паре сетевых фишек пару сетевых фишек, в которой вторая имеет меньшую разметку (меньшее число фишек в позиции r_1).

Утверждение 3.1. *Отношение $\preceq$ на множестве состояний вложенной сети NPN является частичным порядком.*

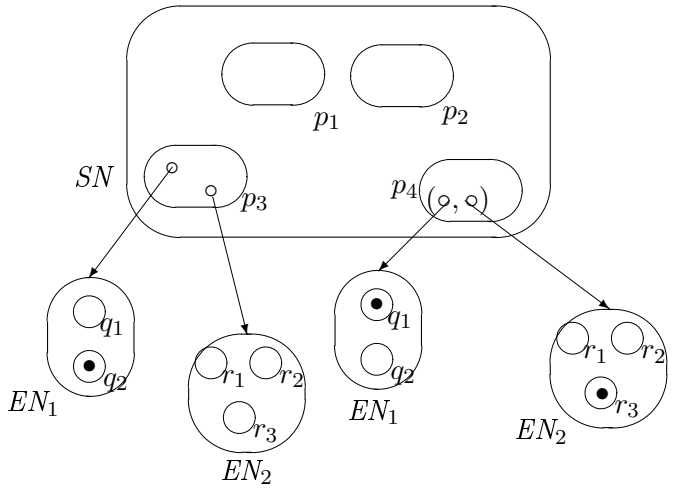


Рис. 3.8: Разметка M' такая, что $M' \preceq M$ (см. рис. 3.6)

Доказательство. Рефлексивность и транзитивность очевидны. Докажем антисимметричность.

Пусть для разметок M_1 и M_2 сети NPN выполняется $M_1 \preceq M_2$ и $M_2 \preceq M_1$. Пусть $p \in P_{SN}$ — некоторая позиция системной сети SN. Рассмотрим мультимножества $M_1(p)$ и $M_2(p)$ кортежей элементов (фишек). По определению порядка $\preceq$ либо $M_1(p)$ и $M_2(p)$ одновременно пусты, либо существуют два инъективных отображения $\imath_p : M_1(p) \to M_2(p)$ и $\jmath_p : M_2(p) \to M_1(p)$.

Из инъективности этих отображений следует, что мощности мультимножеств $M_1(p)$ и $M_2(p)$ равны.

Пусть $\overline{\alpha} = (\alpha_1, \ldots, \alpha_n) \in M_1(p)$. Рассмотрим i-ую компоненту кортежа. По определению если α_i $(1 \leq n)$ — атомарная фишка, то $\imath_p(\alpha_i) = \alpha_i$.

Для сетевой фишки $\alpha_i = (EN, m)$ имеем $\imath_p(\alpha_i) = \alpha_i' = (EN, m')$, где $m \leq m$. Тогда $\jmath_p(\alpha_i') = (EN, m'')$, где $m' \leq m''$. Продолжая это построение, получим последовательность разметок $m \leq m' \leq m'' \leq \ldots$. В силу конечности мультимножеств $M_1(p)$ и $M_2(p)$ эта

последовательность стабилизируется, начиная с некоторого номера. Аналогично, стабилизируются и последовательности, составленные из других компонент кортежей и, следовательно, найдется кортеж $\overline{\alpha}$ такой, что $\imath_p(\overline{\alpha}) = \overline{\alpha}$ и, соответственно, $\jmath_p(\overline{\alpha}) = \overline{\alpha}$.

Тогда рассмотрим разметки $M_1'(p) = M_1(p) - \overline{\alpha}$ и $M_2'(p) = M_2(p) - \overline{\alpha}$. Очевидно, выполняется $M_1' \preceq M_2'$ и $M_2' \preceq M_1'$. Повторяя приведенные выше рассуждения, найдем общий элемент в мультимножествах $M_1'(p)$ и $M_2'(p)$ и т.д. Поскольку мощности мультимножеств $M_1(p)$ и $M_2(p)$ равны, получаем, что и сами мультимножества $M_1(p)$ и $M_2(p)$ совпадают, что и требовалось. □

Нетрудно заметить, что разметка M_1 меньше разметки M_2 относительно порядка $\preceq$, если разметка M_1 может быть получена из разметки M_2 с помощью последовательных удалений фишек из позиций в системной сети или в ее сетевых фишках. Другими словами, M_1 может быть получена из M_2 путем удалений фишек и/или замен сетевых фишек на фишки с той же сетевой структурой, но с меньшими разметками. Это свойство отношения $\preceq$ связывает его с 1-упорядочением мультимножеств (см.раздел 1.2.2). В самом деле, определим отношение порядка$\leq_\circ$ на множестве всех возможных фишек сети *NPN*. Для атомарных фишек a_1 и a_2 полагаем $a_1 \leq_\circ a_2 \Leftrightarrow a_1 = a_2$. Для сетевых фишек $\alpha_1 = (EN_1, m_1)$ и $\alpha_2 = (EN_2, m_2)$ полагаем $\alpha_1 \leq_\circ \alpha_2 \Leftrightarrow (EN_1 = EN_2)\&(m_1 \leq m_2)$. Атомарные и сетевые фишки не сравнимы между собой. Это отношение естественно распространяется на кортежи фишек как покоординатное сравнение $\leq_\circ^n$. Пусть далее для позиции $p \in P_{SN}$ системной сети $\mathfrak{M}_p(NPN) =_{\text{def}} \{M(p) \mid M \in \mathfrak{M}(NPN)\}$ — множество проекций разметок сети *NPN* на позицию p. Тогда имеет место следующая

Лемма 3.1. *Для каждой позиции $p \in P_{SN}$ проекция отношения $\preceq$ на множество $\mathfrak{M}_p(NPN)$ является 1-упорядочением мультимножеств, основанным на отношении порядка $\leq_\circ$ (отношении $\leq_\circ^n$ для кортежей в случае, когда арность позиции p равна $n > 1$)*

на множестве фишек (кортежей фишек).

Доказательство. Пусть $M_1 \preceq M_2$ – две разметки сети NPN, p – позиция ее системной сети. Проверим условия (a), (b) и (c) из определения 1.4 1-упорядочения мультимножеств. Условие (a) соответствует случаям, когда либо $M_1(p)$ пусто, либо $M_1(p)$ и $M_2(p)$ содержат по одной фишке, и, очевидно, выполняется. Условие (b) устанавливается путем доопределения инъективного отображения $\jmath_p$ на дополнительных фишках как тождественного. И, наконец, транзитивность (c) следует из уже доказанного факта, что отношение $\preceq$ является частичным порядком. □

Теорема 3.1. *Отношение $\preceq$ на множестве состояний вложенной сети NPN является правильным квазиупорядочением.*

Доказательство. Пусть задана двухуровневая NP-сеть $NPN = (Atom, Lab, SN, (EN_1, \ldots, EN_k), \Lambda)$, где $SN = (N, \mathcal{L}, \mathcal{U}, W, M_0)$. Разметка сети NPN есть функция типа $M : P_{SN} \to \mathcal{M}(\mathrm{A}^*)$, и, следовательно, полагая, что множество позиций системной сети линейно упорядочено некоторым фиксированным образом, всякую разметку сети NPN можно представить в виде вектора размерности n, где $n =\mid P \mid$.

Поэтому, в силу леммы 1.5, доказательство теоремы достаточно провести для случая, когда системная сеть имеет одну позицию, т. е. разметка сети есть мультимножество маркированных элементных сетей и атомарных фишек.

В силу леммы 3.1 в случае, когда системная сеть имеет одну позицию, отношение $\preceq$ на разметках является 1-упорядочением мультимножеств, основанным на отношении порядка $\leq_\circ^n$, $n \geq 1$. Легко видеть, что отношение $\leq_\circ^n$ является правильным частичным порядком. Тогда по теореме 1.1 отношение частичного порядка $\preceq$ также является правильным квазиупорядочением. □

Следствие 3.1. *Пусть $M_0[\,\rangle M_1[\,\rangle \ldots$ — некоторое бесконечное исполнения для вложенной сети NPN. Тогда найдутся индексы*

$i < j$ такие, что $M_i \preceq M_j$.

Частичному порядку на разметках можно сопоставить порядок на множестве деревьев разметок.

Утверждение 3.2. *Пусть $M_1, M_2 \in \mathfrak{M}(NPN)$ — разметки вложенной сети NPN и $\mathcal{T}(M_1), \mathcal{T}(M_2)$ — соответствующие им деревья разметок. Тогда $M_1 \preceq M_2$ в том и только том случае, когда дерево $\mathcal{T}(M_1)$ изоморфно вкладывается в дерево $\mathcal{T}(M_2)$ (относительно равенства на множестве меток), т. е. существует отображение $\varphi : Nodes(\mathcal{T}(M_1)) \to Nodes(\mathcal{T}(M_2))$ такое, что:*

1) *φ — инъективна;*

2) *φ монотонна и переводит смежные вершины в смежные, т. е. для любых смежных вершин $\alpha, \beta \in Nodes(t)$ если $\alpha \leq \beta$, то $\varphi(\alpha) \leq \varphi(\beta)$ и вершины $\varphi(\alpha), \varphi(\beta)$ также смежны;*

3) *для любой вершины $\alpha \in Nodes(t)$: $\mathcal{L}(\alpha) = \mathcal{L}'(\varphi(\alpha))$, где $\mathcal{L}, \mathcal{L}'$ — функции пометки вершин для деревьев t и t' соответственно.*

Доказательство. Отображение φ сопоставляет корню дерева разметок $\mathcal{T}(M_1)$, помеченного именем системной сети, корень дерева разметок $\mathcal{T}(M_2)$. Вершинам в $\mathcal{T}(M_1)$, соответствующим позициям системной сети, сопоставляются такие же вершины в дереве $\mathcal{T}(M_2)$. Вершинам, соответствующим фишкам в разметке M_1, соответствующие вершины в $\mathcal{T}(M_2)$ определяются отображениями $\jmath_p$ из определения упорядочения $\preceq$. Поскольку разметки сетевых фишек в M_1 содержат черных точек не больше, чем разметки сопоставленных им отображениями $\jmath_p$ сетевых фишек в M_2, соответствующее отображение φ для вершин, помеченных единицами, также легко строится.

Обратное утверждение также легко устанавливается. Отображения $\jmath_p$ из определения упорядочения $\preceq$ сопоставляют фишкам

из разметки M_1 фишки из разметки M_2 так, что соответствующие им вершины в деревьях разметок $\mathcal{T}(M_1)$ и $\mathcal{T}(M_2)$ связаны соотношением φ. □

3.4.3 Свойство совместимости

Теорема 3.2. *Пусть NPN — двухуровневая вложенная сеть, $M_1 \preceq M_2$ — разметки сети NPN.*

1) *Если $M_1[Y\rangle M_1'$, где Y есть либо переход t (в случае элементно-автономного шага), либо пара переходов t_1, t_2 (в случае шага горизонтальной синхронизации), то выполняется $M_2[Y\rangle M_2'$ и $M_1' \preceq M_2'$.*

2) *Если $M_1[Y[b]\rangle M_1'$, где $Y[b]$ есть либо означенный переход $t[b]$ (в случае системно-автономного шага), либо набор переходов вместе с некоторым означиванием $t[b]; t_1, \ldots, t_k$ (в случае шага вертикальной синхронизации), то найдется такое означивание b', что $M_2[Y[b']\rangle M_2'$ и $M_1' \preceq M_2'$.*

Утверждение теоремы можно изобразить в виде следующей коммутативной диаграммы:

$$\begin{array}{ccc} M_1 & \preceq & M_2 \\ {\scriptstyle Y[b]}\downarrow & & {\scriptstyle (\exists b'\colon)}\downarrow{\scriptstyle Y[b']} \\ M_1' & \preceq & M_2' \end{array}$$

Доказательство. Пусть $M_1 \preceq M_2$ и $\{\jmath_p\}_{p \in P_{SN}}$ — соответствующее этому порядку семейство инъективных отображений. Доказательство проведем разбором случаев, в зависимости от вида шага срабатывания.

1. Пусть $M_1[t[b]\rangle M_1'$ — *системно-автономный шаг* сети *NPN*. Тогда переход t сети SN является активным при разметке M_1 и

означивании $b : Var \to \mathrm{A}$, т. е.

$$\forall p \in {}^\bullet t : W(p,t)(b) \subseteq M_1(p), \tag{3.1}$$

а результатом срабатывания перехода t является разметка M_1', для которой

$$M_1'(p) = (M_1(p) \setminus W(p,t)(b)) + W(t,p)(b). \tag{3.2}$$

Тогда в качестве функции означивания для разметки M_2 возьмем функцию $b' : Var \to \mathrm{A}$ такую, что для переменной x, входящей в выражение $W(p,t)$, $b'(x) =_{\mathrm{def}} \jmath_p(b(x))$. Данное определение корректно, поскольку для $p_1 \neq p_2 \in P_{SN}$ $Var(W(p_1,t)) \cap Var(W(p_2,t)) = \emptyset$ в силу ограничений на выражения, приписанные входным дугам переходов в системной сети.

Поскольку выражения на входных дугах не содержат констант и каждая переменная входит в такое выражение не более одного раза (в силу тех же ограничений), то из условия 3.1 и $\jmath_p(M_1(p)) \subseteq M_2(p)$ следует $\forall p \in {}^\bullet t : W(p,t)(b \circ \jmath_p) \subseteq \jmath_p(M_1(p)) \subseteq M_2(p)$, т. е. переход t является активным при разметке M_2 и означивании b'.

Тогда результат срабатывания t при означивании b' есть разметка $M_2'(p) = (M_2(p) \setminus W(p,t)(b \circ \jmath_p)) + W(t,p)(b \circ \jmath_p)$. Поскольку $W(t,p)(b \circ \jmath_p)$ отличается от $W(t,p)(b)$ только возможно большей разметкой в некоторых элементных сетях, нетрудно видеть, что $\jmath_p$, доопределенное на тех кортежах α, на которых оно не было определено, как $\jmath_p(\alpha) = \alpha$, является инъективным отображением, в силу существования которого $M_1' \preceq M_2'$.

2. Пусть $M_1 \preceq M_2$ и $M_1[t\rangle M_1'$ — *элементно-автономный шаг* сети *NPN*. Тогда имеются позиция $p \in P_{SN}$ в системной сети *SN* и кортеж элементов $\overline{\alpha} = (\alpha_1, \ldots, \alpha_n) \in M(p)$, такой что $\alpha_i = (EN, m_1)$, переход t активен в сети *EN* при разметке m_1 и выполняется $m_1[t\rangle m_1'$.

Пусть далее $\jmath_p(\alpha_i) = (EN, m_2)$, где $m_2 \geq m_1$. Тогда, очевидно, переход t активен в сети *EN* при разметке m_2 и, следовательно,

возможно элементное срабатывание этого перехода в сети NPN при разметке M_2. Пусть $m_2[t\rangle m_2'$ и, соответственно, $M_2[t\rangle M_2'$. Разметка M_2' отличается от M_2 только разметкой внутренней сетевой фишки α_i. Чтобы показать, что $M_1' \preceq M_2'$, построим инъективную функцию $\tilde{\jmath}_p : M_1'(p) \to M_2'(p)$ следующим образом:

$$\tilde{\jmath}_p(\alpha) = \begin{cases} (EN, m_2') & \text{при } \alpha = (EN, m_2), \\ \jmath_p(\alpha) & \text{при } \alpha \neq (EN, m_2). \end{cases}$$

Для позиции $p' \neq p$ полагаем $\tilde{\jmath}_p \equiv \jmath_p$. Нетрудно видеть, что так определенное семейство отображений $\{\tilde{\jmath}_p\}_{p \in P_{SN}}$ удовлетворяет условию инъективности, и, следовательно, $M_1' \preceq M_2'$.

3. Пусть $M_1 \preceq M_2$ и $M_1[t_1, t_2\rangle M_1'$ — *шаг горизонтальной синхронизации* сети NPN. Тогда в некоторой позиции $p \in P_{SN}$ в системной сети SN при разметке M имеется кортеж элементов $\overline{\alpha} = (\alpha_1, \ldots, \alpha_n) \in M(p)$, в котором $\alpha_i = (EN_1, m_1), \alpha_j = (EN_2, m_2)$, $m_1[t_1\rangle m_1'$ в сети EN_1 и $m_2[t_1\rangle m_2'$ в сети EN_2, где переходы t_1, t_2 помечены взаимно дополнительными метками из Lab_h.

Пусть теперь $\jmath_p(\alpha_i) = (EN, m_3)$ и $\jmath_p(\alpha_j) = (EN, m_4)$, где $m_3 \geq m_1$ и $m_4 \geq m_2$. Аналогично предыдущему случаю получаем, что найдутся состояния m_3' и m_4', такие что $m_3[t\rangle m_3'$ в сети EN_1 и $m_4[t\rangle m_4'$ в сети EN_2, и, соответственно, возможен шаг горизонтальной синхронизации $M_2[t\rangle M_2'$ в сети NPN. При этом разметка M_2' получается из M_2 заменой разметок m_3 и m_4 внутренних сетевых фишек α_i, α_j на разметки m_3' и m_4' соответственно. Инъективное отображение $\tilde{\jmath}_p : M_1'(p) \to M_2'(p)$, определяющее упорядоченность $M_1' \preceq M_2'$, строится аналогично предыдущему случаю. Полагаем

$$\tilde{\jmath}_p(\alpha) = \begin{cases} (EN, m_3') & \text{при } \alpha = (EN, m_3), \\ (EN, m_4') & \text{при } \alpha = (EN, m_4), \\ \jmath_p(\alpha) & \text{при } \alpha \neq (EN, m_3) \text{ и } \alpha \neq (EN, m_4). \end{cases}$$

Для позиции $p' \neq p$ полагаем $\tilde{\jmath}_p \equiv \jmath_p$.

Если арность позиции p равна 1 и p содержит не кортежи, а единичные фишки, то все приведенные рассуждения также остаются верными.

4. Пусть $M_1 \preceq M_2$ и $M_1[t[b]; t_1, \ldots, t_k\rangle M_1'$ — *шаг вертикальной синхронизации* сети *NPN*. Тогда в системной сети *SN* имеется переход t, активный при разметке M_1 и означивании b, и $M \xrightarrow{t} M'$. Как уже было доказано в п.1 этого доказательства, отсюда следует, что переход t является активным при разметке M_2 и некотором означивании b' и $M_2[t[b]\rangle M_2'$, где $M_2 \preceq M_2'$. Далее, для каждой системной фишки $\alpha_i = (EN_i, m_i)$ в позиции p, задействованной в b-срабатывании перехода t в системной сети *SN*, выполняется $m_i[t_i\rangle m_i'$. По уже доказанному случаю для элементно-автономных шагов из $M_1 \preceq M_2$ следует, что в сети $\jmath_p(\alpha_i) = (EN, m_{i'})$, где $m_{i'} \geq m_i$, переход t_i активен и, следовательно, возможно элементно-автономное срабатывание этого перехода в сети *NPN* при разметке M_2. Результатом срабатывания t_i является локальная разметка $m_{i'}'$ сетевой фишки, такая что $m_i' \preceq m_{i'}'$.

Таким образом, в сети *NPN* возможен шаг вертикальной синхронизации при означивании b'. Результатом его является разметка M_2'', полученная из разметки M_2' путем замены сетевой фишки $\alpha_i = (EN_i, m_i')$ на фишку $\alpha_i' = (EN_i, m_{i'}')$ (имеющую большую разметку) для всех $1 \leq i \leq k$. Нетрудно видеть, что в результате замены локальных разметок некоторых сетевых фишек на большие разметки мы получим разметку M_2'', удовлетворяющую $M_2'' \preceq M_2'$ и, следовательно, $M_2'' \preceq M_2'$, что и требовалось. □

Замечание 3.2. Заметим, что без ограничений на выражения на входных дугах в системной сети свойство совместимости для вложенных сетей не выполнялось бы. Примеры, приведенные на рисунках 3.9, 3.10, 3.11, показывают, что нарушение любого из трех ограничений приводит к нарушению свойства совместимости.

Замечание 3.3. Для обыкновенных сетей Петри имеет место более сильное утверждение, а именно:

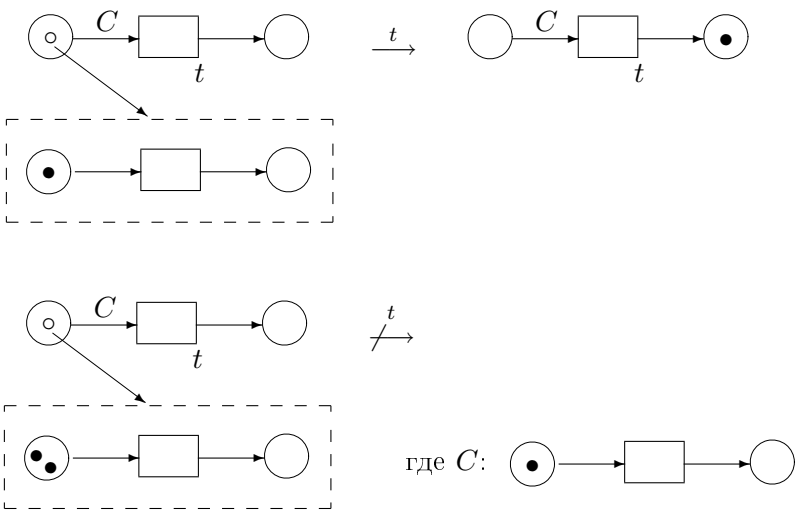


Рис. 3.9: Пример 1.

Если $M_1[t\rangle M_1'$ для обыкновенной сети PN и $M_1 < M_2$, то найдется состояние M_2' такое, что $M_2[t\rangle M_2'$ и $M_1' < M_2'$. Это свойство называется свойством *строгой совместимости* и может быть изображено в виде следующей коммутативной диаграммы:

$$\begin{array}{ccc} M_1 & < & M_2 \\ {\scriptstyle t}\downarrow & & \downarrow{\scriptstyle t} \\ M_1' & < & M_2' \end{array}$$

Выполнение свойства строгой совместимости обеспечивает разрешимость для обыкновенных сетей Петри таких свойств, как ограниченность позиции, ограниченность сети, потенциальная живость перехода.

Для вложенных сетей свойство строгой совместимости не выполняется, т. е. отношение порядка $\preceq$ не может быть заменено на строгое, что показывает пример, приведенный на рисунке 3.12.

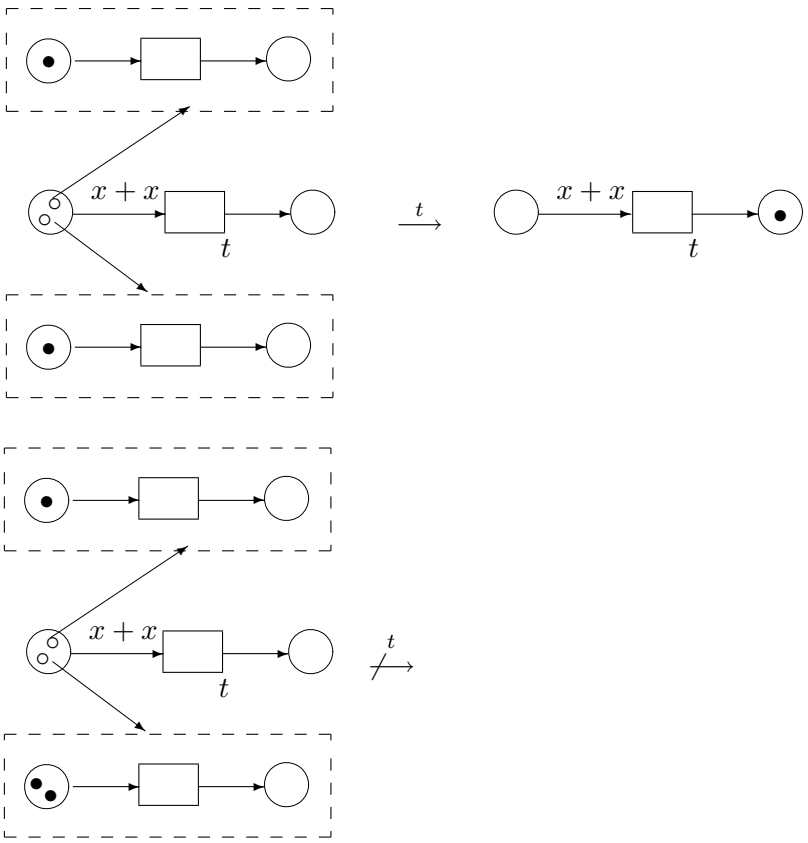


Рис. 3.10: Пример 2.

Итогом этого раздела является следующая

Теорема 3.3. *Пусть NPN — двухуровневая вложенная сеть Петри, $\preceq$ — частичный порядок на множестве $\mathfrak{M}(NPN)$ разметок сети NPN, определенный выше. Тогда $\langle NPN, \preceq \rangle$ является вполне структурированной системой переходов.*

Доказательство. Следует из теоремы 3.1 о правильности квази-

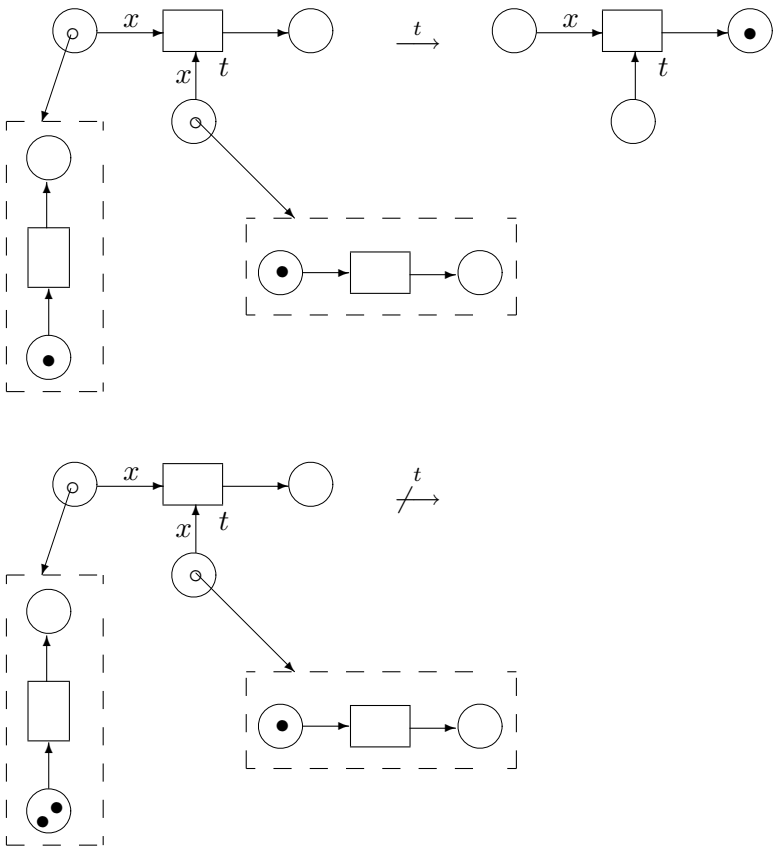


Рис. 3.11: Пример 3.

упорядочения $\preceq$ и теоремы 3.2 о выполнении свойства совместимости отношений перехода и $\preceq$ для двухуровневых вложенных сетей Петри. $\square$

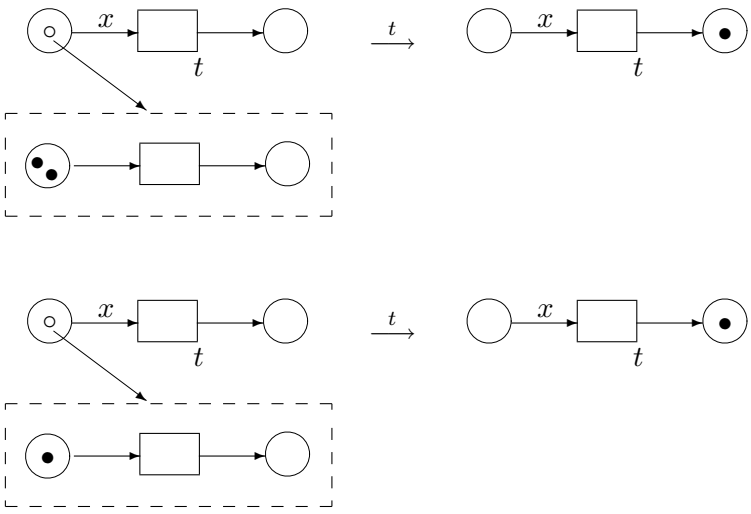


Рис. 3.12: Пример нарушения строгой совместимости для вложенных сетей.

3.5 Анализ семантических свойств

3.5.1 Покрывающее дерево сети. Проблема останова

В предыдущем разделе было показано, что двухуровневые вложенные сети Петри являются вполне структурированными системами переходов относительно частичного порядка $\preceq$ на множестве разметок сети.

Отношение $\preceq$, очевидно, разрешимо. Действительно, для сравнения двух разметок достаточно проверить для каждой позиции системной сети, сравнимы ли относительно 1-упорядочения мультимножеств мультимножества фишек в этой позиции при двух заданных разметках. Поскольку мультимножества фишек в позиции всегда конечны, это можно сделать, например, простым перебором.

Напомним, что в выражениях на выходных дугах перехода

встречаются только переменные, входящие в выражения на входных дугах этого перехода. Это ограничение обеспечивает конечность ветвлений диаграммы переходов вложенной сети Петри. Легко заметить также, что множество $Succ(M)$ (множество всех последующих разметок для разметки M) для заданной вложенной сети NPN вычислимо.

Таким образом, в силу утверждения 1.7 для двухуровневой вложенной сети Петри может быть эффективно построено покрывающее дерево сети.

В качестве примера построим дерево покрытия для NP-сети NPN_1, приведенной на рис. 3.1. Системная компонента сети NPN_1 имеет шесть позиций $S_1, S_2, \ldots, S_6$, и разметки сети NPN_1 будем изображать кортежами из шести элементов. Элементная компонента сети NPN_1 также содержит 6 позиций, будем располагать их в следующем порядке: $W_0, \ldots, W_4, T$. Если позиция S_i в некоторой разметке пуста, то на соответствующем месте кортежа будем писать 0. Натуральное число в кортеже будет обозначать число черных атомарных фишек в соответствующей позиции. Если же позиция S_i содержит, скажем, две сетевые фишки EN с разметками $m = (m_1, m_2, \ldots, m_6)$ и $m' = (m'_1, m'_2, \ldots, m'_6)$ соответственно, то на i-м месте кортежа разметки будем писать $\{EN(m_1, m_2, \ldots, m_6), EN(m'_1, m'_2, \ldots, m'_6)\}$. Таким образом, начальная разметка M_0 сети NPN_1 запишется как $(1, 0, 0, 0, A, 0)$.

Дерево покрытия для сети NPN_1 приведено на рис. 3.13. Здесь
$M_1 = (1, \{EN(0, 0, 1, 0, 0, 0)\}, 0, 0, A, 0)$,
$M_2 = (1, 0, \{EN(0, 0, 0, 1, 0, 0)\}, 0, A - a, 0)$,
$M'_2 = (1, \{EN(0, 0, 1, 0, 0, 0), EN(0, 0, 1, 0, 0, 0)\}, 0, 0, A, 0)$,
$M''_2 = (1, \{EN(0, 0, 1, 0, 0, 1)\}, 0, 0, A, 0)$,
$M_3 = (1, 0, 0, 0, A - a, \{EN(0, 0, 0, 0, 1, 0)\})$,
$M'_3 = (1, \{EN(0, 0, 1, 0, 0, 0)\}, \{EN(0, 0, 0, 1, 0, 0)\}, 0, A - a, 0)$,
$M''_3 = (1, 0, \{EN(0, 0, 0, 1, 0, 1)\}, 0, A - a, 0)$,
$M'_4 = (1, \{EN(0, 0, 1, 0, 0, 0)\}, 0, 0, A - a, \{EN(0, 0, 0, 0, 1, 0)\})$,
$M''_4 = (1, 0, 0, 0, A - a, \{EN(0, 0, 0, 0, 1, 1)\})$.

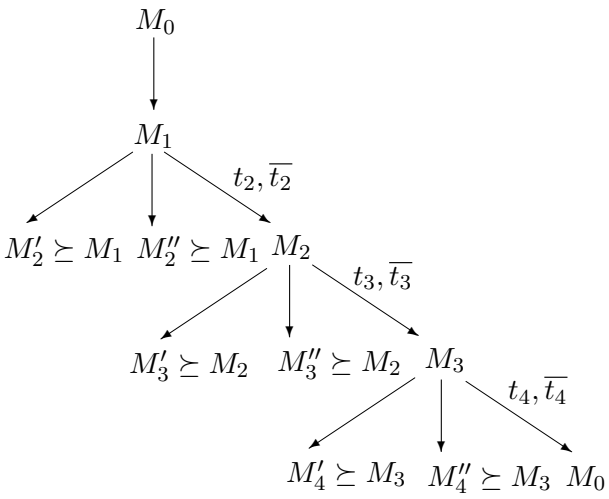


Рис. 3.13: Дерево покрытия для NP-сети NPN_1 (см. рис. 3.1).

Все листья в дереве покрытия для сети NPN_1 являются покрывающими вершинами. На этом примере хорошо видно, что дерево покрытия не дает полной информации о поведении сети. Так, путь от корня до листа с пометкой M_0 указывает на наличие циклического исполнения, которое соответствует циклу, который проходит в этой системе один исполнитель, начиная от его обращения за инструментом и заканчивая возвращением этого инструмента на склад. Пути, ведущие к разметкам с одним штрихом, соответствуют обращению на склад еще одного клиента. Это обращение не мешает продолжить свой обычный цикл клиенту, который уже получил инструмент. Поэтому все эти вершины являются покрывающими, и дальнейшее исполнение в этом случае не прослеживается. Листья, помеченные разметками с двумя штрихами, соответствуют автономному срабатыванию перехода в элементной компоненте, которое содержательно соответствует появлению у клиента еще одного задания. Это событие не влияет на возмож-

ность продолжения исполнения предыдущего задания, и соответствующая вершина также является покрывающей. Вместе с тем, дерево покрытия никак не отражает возможность или невозможность срабатывания перехода с пометкой t_1. Нетрудно заметить, что этот переход срабатывает, если позиция S_5 пуста, что соответствует тому, что все инструменты разобраны. Условие, связанное с ограниченностью некоторых ресурсов, очевидно, не удовлетворяет свойству монотонности, поэтому оно не может быть охарактеризовано с помощью дерева покрытия.

Как это было показано в параграфе 1.3, на основе построения покрывающего дерева сети можно решать целый ряд проблем анализа семантических свойств сетей.

Лемма 3.2. *Пусть NPN — двухуровневая вложенная сеть Петри, C — некоторый направленный вверх конус (относительно частичного порядка $\preceq$) в множестве разметок сети NPN, M_0 — начальная разметка этой сети. Утверждение темпоральной логики $M_0 \models (\exists)\Box C$ истинно для вложенной сети Петри NPN тогда и только тогда, когда покрывающее дерево сети NPN содержит путь от корня к листу, в котором все вершины помечены разметками, принадлежащими C.*

Доказательство. Следует из утверждения 1.8. □

Проблема поддержки управляющего состояния для вложенной сети *NPN* состоит в том, чтобы для данной разметки M (или конечного набора разметок $M^1, \ldots, M^k$) установить, существует ли исполнение $M_0[t_1\rangle M_1[t_2\rangle \ldots$ для *NPN*, в котором всякая разметка M_i удовлетворяет условию $M_i \succeq M$ (соответственно $M_i \succeq M^1 \vee \cdots \vee M_i \succeq M^k$). Содержательно это означает, что возможно такое исполнение, что “минимальный запас ресурсов”, определяемый разметкой M (или набор взаимозаменяемых ресурсов, определяемый разметками $M^1, \ldots, M^k$) будет поддерживаться в течение всего исполнения.

Теорема 3.4. *Проблема поддержки управляющего состояния разрешима для двухуровневых вложенных сетей Петри.*

Доказательство.
Легко видеть, что множество разметок $\{M' \mid M' \succeq M\}$ (соответственно, множество $\{M' \mid M' \succeq M^1 \vee \cdots \vee M' \succeq M^k\}$) является направленным вверх конусом относительно частичного порядка $\preceq$, а проблема поддержки управляющего состояния может быть записана в виде формулы $M_0 \models (\exists)\Box\{M' \mid M' \succeq M\}$ или, соответственно, $M_0 \models (\exists)\Box\{M' \mid M' \succeq M^1 \vee \cdots \vee M' \succeq M^k\}$, где M_0 — начальная разметка сети (здесь, как это часто делается, мы не различаем множество и предикат принадлежности этому множеству).

Как было показано выше, для двухуровневой вложенной сети Петри можно эффективно построить ее покрывающее дерево, а отношение $\preceq$ — разрешимо. Тогда в силу леммы 3.2 решение проблемы поддержки управляющего состояния сводится к проверке того, что покрывающее дерево сети *NPN* содержит путь от корня к листу, в котором все вершины помечены разметками, большими в смысле $\preceq$ заданной разметки M или, соответственно, одной из заданных разметок $M^1, \ldots, M^k$. □

Проблема неизбежности для вложенных сетей Петри является двойственной к проблеме поддержки управляющего состояния. Для сети *NPN* она состоит в том, чтобы для данной разметки M (или конечного набора разметок $M^1, \ldots, M^k$) установить, что всякое исполнение $M_1[t_2\rangle \ldots$ для *NPN* рано или поздно приводит к разметке M_i, для которой свойство $M_i \succeq M$ (соответственно, $M_i \succeq M^1 \vee \cdots \vee M_i \succeq M^k$) не выполняется. Содержательно это означает, что при любом исполнении "минимальный запас ресурсов", определяемый разметкой M (или набор взаимозаменяемых ресурсов, определяемый разметками $M^1, \ldots, M^k$) будет исчерпан.

Теорема 3.5. *Проблема неизбежности разрешима для двухуровневых вложенных сетей Петри.*

Доказательство. Проблема неизбежности разрешима в силу ее двойственности проблеме поддержки управляющего состояния (ср. с утверждением 1.10 из главы 1). □

Проблема останова: по данной начальной разметке M_0 определить, верно ли, что всякое исполнение сети приводит к тупиковой разметке, т. е. такой разметке, при которой ни один переход не может сработать.

Проблема останова сводится к проблеме неизбежности.

Теорема 3.6. *Проблема останова разрешима для двухуровневых вложенных сетей Петри.*

Доказательство. Для решения проблемы останова достаточно построить покрывающее дерево сети. Если все листья в этом дереве помечены тупиковыми разметками, то всякое исполнение сети завершается. Если же найдется лист, помеченный нетупиковой разметкой M (т. е. разметкой, большей или равной уже встречавшейся на этом пути разметки M', $M' \preceq M$), то возможно бесконечное продолжение этого исполнения. Действительно, в этом случае в силу свойства совместимости все шаги, приводящие от M к M', можно повторять бесконечное число раз. □

3.5.2 Метод насыщения. Свойство покрытия

Проблема покрытия для вложенной сети Петри состоит в следующем: для данной разметки M_0 и заданного своим конечным базисом направленного вверх конуса C проверить достижимость из M_0 некоторой разметки, принадлежащей C, т. е. проверить истинность формулы $M_0 \models (\exists)\Diamond C$. В частности, для $C = \uparrow M$, где M — некоторая разметка сети, проблема покрытия состоит в проверке достижимости из разметки M_0 разметки $M' \succeq M$.

Поскольку вложенные сети Петри являются вполне структурированными системами переходов, проблему покрытия будем решать с помощью *метода насыщения* (см. параграф 1.3), основан-

ного на свойстве стабилизации возрастающей последовательности направленных вверх конусов (см. утверждение 1.4). Для этого нужно показать, что любая двухуровневая вложенная сеть Петри NPN имеет эффективный предбазис, т. е. для любой разметки $M \in \mathfrak{M}(NPN)$ может быть эффективно вычислен конечный базис $pb(M)$ множества $Pred(\uparrow M)$.

Пусть NPN — двухуровневая вложенная сеть Петри, t — некоторый переход в NPN. Через ${}^\circ t$ обозначим множество минимальных относительно $\preceq$ разметок сети NPN, при которых переход t является активным. В силу вполне упорядочиваемости отношения $\preceq$ множество ${}^\circ t$ конечно.

Лемма 3.3. *Для любой двухуровневой сети NPN и перехода t этой сети множество ${}^\circ t$ может быть эффективно построено.*

Доказательство. В качестве доказательства приведем
Алгоритм построения множества ${}^\circ t$ минимальных разметок, при которых переход t может сработать.

Определим множество V_{NPN} как множество минимальных значений для переменных в выражениях, приписанных дугам сети NPN. Множество V_{NPN} состоит из конечного набора всех индивидуальных переменных сети NPN и множества всех ее элементных сетей с пустой разметкой. Множество V_{NPN}, очевидно, конечно.

1. Пусть сначала t — непомеченный переход в системной сети. Пусть $p \in {}^\bullet t$ — некоторая входная позиция для t в системной сети, $W(p,t)$ — выражение, приписанное дуге (p,t).

Если $W(p,t)$ не содержит переменных, то для каждой разметки $M \in {}^\circ t$ значение $M(p)$ полагаем равным значению выражения $W(p,t)$.

Если $W(p,t)$ содержит переменные $x_1, \ldots, x_k$, то $M(p)$ может принимать любое значение, принимаемое выражением $W(p,t)$ при означивании переменных $x_1, \ldots, x_k$ значениями из V_{NPN}. Легко видеть, что множество возможных значений для $M(p)$ также конечно.

Тогда множество $^\circ t$ строится как набор всех возможных разметок, для которых $M(p)$ принимает одно из конечного числа описанных выше значений.

2. Пусть t — помеченный меткой для вертикальной синхронизации переход в системной сети. Тогда множество $^\circ t$ строится так же, как и в предыдущем случае, с той только разницей, что элементные сети в множестве V_{NPN} получают не пустую разметку, а одну из минимальных разметок, при которой активен один из переходов, помеченных $\overline{\mathcal{L}(t)}$ в элементной сети. Если в элементной сети имеется ровно один переход, помеченный $\overline{\mathcal{L}(t)}$, то имеется ровно одна такая разметка. Если таких переходов несколько, то для каждого из них строится соответствующая разметка. И, наконец, если в элементной сети нет переходов, помеченных $\overline{\mathcal{L}(t)}$, то эта сеть исключается из множества возможных значений.

3. Пусть теперь t — непомеченный переход в некоторой элементной сети EN сети NPN. Пусть m — минимальная разметка сети EN, при которой переход t активен. Тогда множество минимальных разметок $^\circ t$ строится как множество всех разметок, при которых одна из позиций системной сети содержит сетевую фишку (EN, m), а все остальные позиции пусты.

4. И, наконец, если t — помеченный меткой для горизонтальной синхронизации переход в элементной сети EN сети NPN, то множество минимальных разметок $^\circ t$ строится как множество всех разметок, при которых одна из позиций системной сети содержит сетевую фишку (EN, m) и некоторую другую сетевую фишку (EN', m'), а все остальные позиции пусты. Здесь сеть EN' имеет переход, помеченный дополнительной меткой $\overline{\mathcal{L}(t)}$ и m' — минимальная разметка сети EN', при которой переход, помеченный $\overline{\mathcal{L}(t)}$, активен. □

Лемма 3.4. *Для любой двухуровневой вложенной сети Петри NPN и произвольной разметки $M \in \mathfrak{M}(NPN)$ существует эффективная процедура построения конечного базиса множества $Pred(\uparrow M)$.*

Доказательство. Пусть NPN — двухуровневая вложенная сеть Петри, $M \in \mathfrak{M}(NPN)$ — некоторая ее разметка. В предыдущей лемме было доказано, что множество ${}^\circ t$ минимальных относительно $\preceq$ разметок сети NPN, при которых переход t является активным, может быть эффективно построено.

Через t° обозначим множество разметок, получаемых в результате всех возможных срабатываний перехода t (возможно, совместно с другими переходами) на разметках из множества ${}^\circ t$. Очевидно, что множество t° конечно и может быть получено применением перехода t ко всем разметкам из ${}^\circ t$.

Рассмотрим множество $C = \uparrow M \cap \uparrow t^\circ$ разметок, состоящее из M' таких, что M' может быть получена в результате срабатывания перехода t и $M \preceq M'$. Это множество является направленным вверх конусом и в силу вполне упорядочиваемости отношения $\preceq$ имеет конечный базис, который мы будем обозначать через $lub(M, t^\circ)$.

Конечный базис $lub(M, t^\circ)$ может быть эффективно построен путем следующей процедуры. Пусть $M' \in t^\circ$. Построим разметку M_1. Пусть p — некоторая позиция сети NPN. Мультимножества фишек $M(p)$ и $M'(p)$ разобьем на суммы мультимножеств атомарных и сетевых фишек: $M(p) = [M(p)]_{atom} + [M(p)]_{net}$ и, соответственно, $M'(p) = [M'(p)]_{atom} + [M'(p)]_{net}$. Полагаем $M_1(p) = ([M(p)]_{atom} \cup [M'(p)]_{atom}) + ([M(p)]_{net} + [M'(p)]_{net}$. При этом сетевые фишки, которые "пришли" в $M_1(p)$ из $M(p)$, пометим нулем, а сетевые фишки из $M'(p)$ — единицей.

Если в $M_1(p)$ имеются две сетевые фишки с одинаковой сетевой структурой и разными пометками (ноль и единица), то построим разметку $M_2(p)$, заменив эти две фишки одной фишкой с той же сетевой структурой и разметкой — максимумом двух исходных разметок, т. е. для каждой позиции q сетевой фишки $m(q) = \max(m_1(q), m_2(q))$, где m_1, m_2 — исходные, m — новая разметки сетевых фишек. Затем, если в $M_1(p)$ имеется другая пара сетевых фишек, удовлетворяющая тому же условию, стро-

им разметку M_3, заменяя эту пару на "максимальную" фишку. Построение продолжаем, пока все такие пары во всех позициях сети не будут исчерпаны. Затем применяем ту же процедуру ко всем построенным разметкам, начиная с M_2, затем — к вновь построенным разметкам. Поскольку число фишек в каждой позиции сети NPN конечно, в некоторый момент этот процесс прервется. Выберем в построенном множестве разметок подмножество минимальных относительно $\preceq$ элементов и получим конечный базис $lub(M, t^\circ)$.

Пусть теперь некоторая разметка $M' \in Pred(\uparrow M)$. Тогда возможно срабатывание некоторого перехода t (автономное или совместно с другими переходами), которое переводит M' в разметку $M'' \in \uparrow M$. Отсюда следует, что $M'' \in \uparrow t^\circ$.

С другой стороны, легко проверить, что для любой разметки M'' такой, что $M'' \in \uparrow M$ и $M'' \in \uparrow t^\circ$, выполняется $Pred_t(M'')$. Получаем $Pred(\uparrow M) = \cup_t Pred_t(\uparrow lub(M, t^\circ)) = \uparrow(\cup_t Pred_t(lub(M, t^\circ)))$.

Эта формула задает эффективное построение конечного базиса множества $Pred(\uparrow M)$ по конечному базису $lub(M, t^\circ)$. □

Теорема 3.7. *Проблема покрытия разрешима для двухуровневых вложенных сетей Петри.*

Доказательство. В силу теоремы 1.4 разрешимость проблемы покрытия следует из эффективности предбазиса (лемма 3.4) и разрешимости отношения порядка $\preceq$ для двухуровневых вложенных сетей Петри. □

К проблеме покрытия можно свести некоторые другие проблемы анализа сетей Петри.

Переход t в сети Петри PN называется *потенциально живым*, если существует достижимая в PN разметка M, при которой t может сработать.

Проблема потенциальной живости перехода: определить, является ли переход t в некоторой сети Петри потенциально живым.

Разметка M сети Петри PN называется *t-тупиковой* для некоторого перехода t, если для любой достижимой из M разметки M' переход t не может сработать в M'.

Теорема 3.8. *Пусть NPN — двухуровневая вложенная сеть Петри, t — некоторый переход в NPN. Тогда*

1) *проблема потенциальной живости перехода t для сети NPN разрешима;*

2) *для произвольной разметки M можно эффективно установить, является ли она t-тупиковой.*

Доказательство. Пусть NPN — двухуровневая вложенная сеть Петри, t — некоторый переход в NPN. Ранее было доказано, что множество ${}^\circ t$ минимальных относительно $\preceq$ разметок сети NPN, при которых переход t является активным, может быть эффективно построено.

Переход t является потенциально живым в том и только том случае, когда достижима некоторая разметка, принадлежащая направленному вверх конусу с базисом ${}^\circ t$. Тогда по теореме 3.7 о разрешимости проблемы покрытия проблема потенциальной живости перехода для двухуровневых вложенных сетей Петри также разрешима.

Разметка M является t-тупиковой тогда и только тогда, когда $M \notin \uparrow {}^\circ t$, т. е. M не принадлежит направленному вверх конусу с базисом ${}^\circ t$. Таким образом, проблема проверки того, является ли данная разметка t-тупиковой — двойственная к проблеме потенциальной живости перехода и, следовательно, разрешима. □

Замечание 3.4. Для обыкновенных сетей Петри проблема потенциальной живости перехода может решаться с помощью *полного покрывающего дерева* сети ([8]). А именно, переход t потенциально живой в том и только том случае, когда он является пометкой для некоторой дуги в полном покрывающем дереве сети.

Полное покрывающее дерево для обыкновенных сетей Петри можно рассматривать как уточнение полного покрывающего дерева для соответствующей сети Петри системы переходов. В процессе построения полного покрывающего дерева каждая его вершина объявляется либо листом, либо внутренней вершиной. Вершины-листья не имеют потомков. Вершины дерева помечаются векторами из множества $\mathbb{N}_\omega^n = \mathbb{N}_\omega \times \cdots \times \mathbb{N}_\omega$, где n — число позиций сети. Здесь $\mathbb{N}_\omega = \mathbb{N} \cup \{\omega\}$ — множество натуральных чисел, дополненное специальным символом ω для обозначения потенциально неограниченного числа фишек в некоторой позиции. Арифметические операции и отношения распространяются на $\mathbb{N}_\omega$ естественным образом. А именно, полагаем, что для любого $n \in \mathbb{N}$ выполняется
1) $\omega > n$,
2) $n + \omega = \omega + n = \omega + \omega = \omega - n = \omega - \omega = \omega$,
3) $\omega \cdot n = n \cdot \omega = \omega$,
4) $0 \cdot \omega = \omega \cdot n = \omega$.

Полное покрывающее дерево для обыкновенных сетей Петри строится следующим образом.

1) Первоначально предполагается, что дерево содержит единственную вершину M_0, соответствующую начальной разметке сети, и не имеет дуг.

2) Пусть M — вершина дерева, которая еще не объявлена листом, но в дереве нет исходящих из нее дуг. Тогда возможны следующие случаи:

   а) Ни один из переходов не может сработать при разметке M, т. е. $\forall t \in T : M \not\geq F(t)$. В этом случае вершина M объявляется листом.

   б) На пути из корня дерева в вершину M существует вершина M', помеченная той же самой разметкой, то есть $M = M'$. В этом случае вершина M объявляется листом.

в) На пути из корня дерева в вершину M существует вершина M' такая, что $M' < M$. Тогда для всякой позиции $p \in P$, для которой $M'(p) < M(p)$, соответствующая этой позиции координата в векторе разметки заменяется на ω.

Процесс продолжается до тех пор, пока все листья, не имеющие потомков, не будут объявлены листьями.

В силу того, что обыкновенные сети Петри являются вполне структурированными системами переходов со строгой совместимостью, полное покрывающее дерево для любой обыкновенной сети Петри конечно и приведенная процедура его построения обязательно завершается.

Переход t в обыкновенной сети Петри является потенциально живым в том и только том случае, когда он является пометкой для некоторой дуги в полном покрывающем дереве системы переходов. Поскольку для обыкновенных сетей Петри полное покрывающее дерево конечно и может быть эффективно построено, проблема потенциальной живости перехода разрешима для таких систем.

Аналогично, для обыкновенных сетей Петри t-тупиковость некоторой разметки может быть установлена с помощью полного дерева покрытия. Заменим в сети PN начальную разметку на разметку M и построим полное покрывающее дерево для полученной системы. Разметка M является t-тупиковой тогда и только тогда, когда построенное покрывающее дерево не содержит дуги, помеченной t.

3.5.3 Свойства ограниченности, достижимости и живости

В замечании 3.3 было показано, что вложенные сети Петри, рассматриваемые как вполне структурированные системы переходов,

не обладают свойством строгой совместимости, которая выполняется для обыкновенных сетей Петри. Строгая совместимость позволяет извлекать из покрывающего дерева обыкновенной сети Петри дополнительно такое важное семантическое свойство сети, как ограниченность.

Для обыкновенных сетей Петри позиция p в сети PN называется *ограниченной*, если существует число $n \in \mathbb{N}$ такое, что для любой достижимой в сети PN разметки M выполняется $M(p) \leq n$. Обыкновенная сеть Петри называется *ограниченной*, если все ее позиции ограничены. Легко заметить, что если разрешима проблема ограниченности позиции в сети, то разрешима и проблема ограниченности сети.

Известно ([8]), что проблема ограниченности позиции в обыкновенной сети Петри разрешима. Ограниченность позиции p в сети PN проверяется путем построения полного покрывающего дерева сети. Позиция p неограничена в сети PN в том и только том случае, когда среди вершин полного покрывающего дерева сети PN имеется вершина, помеченная (обобщенной) разметкой M, в которой позиции p соответствует символ ω.

Для вложенных сетей Петри понятие ограниченности определяется следующим образом:

Позиция p во вложенной сети Петри NPN *$\preceq$-ограничена (или просто ограничена)* , если существует такое мультимножетсво фишек S, что при любой достижимой в сети PN разметке M выполняется $M(p) \preceq S$, т. е. существует инъективное отображение $\jmath_p : M_1(p) \to M_2(p)$, такое что для $\overline{\alpha} = (\alpha_1, \ldots, \alpha_n) \in M_1(p)$ если $\jmath_p(\overline{\alpha}) = \overline{\alpha}\prime = (\alpha'_1, \ldots, \alpha'_n)$, то для всех $i = 1, \ldots, n$ либо $\alpha_i = \alpha'_i$, либо $\alpha_i = (EN, m)$ есть сетевая фишка и $\alpha'_i = (EN, m')$, где $m \leq m'$.

Будем называть вложенную сеть Петри NPN *ограниченной*, если все ее позиции ограничены.

Таким образом, вложенная сеть Петри NPN является $\preceq$-ограниченной, если существует такая разметка S сети NPN, что для любой достижимой в NPN разметки M выполняется $M \preceq S$.

Поскольку сети Петри не обладают строгой совместимостью, проблему ограниченности для них не удается решить методом построения полного покрывающего дерева, как это делается для обыкновенных сетей Петри. Далее будет доказано, что проблема ограниченности для вложенных сетей Петри неразрешима.

Напомним, что разметка M называется *достижимой* для вложенной сети NPN, если существует последовательность переходов в NPN, переводящая начальную разметку M_0 в разметку M. *Проблема достижимости* состоит в построении алгоритма, который по любой вложенной сети NPN и любой разметке M этой сети определяет, является ли M достижимой для NPN.

Переход t вложенной сети Петри NPN называется *живым*, если он является потенциально живым при любой достижимой в NPN разметке, т. е. для любой достижимой в NPN разметки M найдется достижимая из M разметка M' такая, что t может сработать в M'. Вложенная сеть NPN называется *живой*, если все ее переходы живы.

Проблема живости сети состоит в построении алгоритма, который по произвольной вложенной сети NPN определяет, является ли она живой.

Для обыкновенных сетей Петри проблемы живости сети и достижимости в ней произвольной разметки эквивалентны (взаимно сводимы друг к другу) [8] и разрешимы [55, 70]. Для вложенных сетей Петри обе эти проблемы неразрешимы.

Неразрешимость указанных проблем будет доказана путем сведения их к аналогичным проблемам для сетей Петри с обнуляющими дугами.

Определение 3.4. Сети Петри с обнуляющими дугами (reset arcs Petri nets) [29] есть расширение модели обыкновенных сетей Петри за счет добавления специальных обнуляющих дуг. Обнуляющая дуга всегда направлена от позиции к переходу. При срабатывании соответствующего перехода позиция, являющаяся началом инцидентной переходу дуги, обнуляется, т. е. из нее убираются все

находившиеся в ней фишки (если таковые имелись).

Теорема 3.9. *Любая сеть Петри с обнуляющими дугами может быть промоделирована посредством двухуровневой вложенной сети.*

Доказательство. Моделирование основано на идее представления n фишек в некоторой позиции сети с обнуляющими дугами посредством одной элементной сети, содержащей (во всех своих позициях) n фишек. Тогда обнуление некоторой позиции выполняется как удаление соответствующей элементной сети и замена ее на константную элементную сеть $E0$, не содержащую фишек.

При этом добавление или удаление фишки в результате срабатывания некоторого перехода моделируется как добавление или удаление фишки в соответствующей элементной сети, что делается с помощью механизма вертикальной синхронизации.

Рис. 3.14 иллюстрирует эту идею. В левой части (a) показан фрагмент сети Петри с обнуляющими дугами, содержащей n фишек в позиции p. Дуга (t_+, p) при срабатывании соответствующего перехода добавляет фишку в позицию p, дуга (t_-, p), соответственно, забирает фишку из этой позиции. В правой части (b) этого рисунка показан фрагмент NP-сети, моделирующий левый фрагмент. Здесь n фишек в позиции p представлены одной сетевой фишкой *EN*, которая имеет одну позицию с n атомарными черными фишками и два перехода, помеченные, соответственно, $\overline{l_+}$ и $\overline{l_-}$, которые добавляют или забирают фишку в/из p. Эти переходы срабатывают синхронно с переходами t_+ или, соответственно, t_- в системной сети. Переход t_r в системной сети забирает фишку из p и таким образом обнуляет ее. □

Для сетей Петри с обнуляющими дугами некоторые проблемы анализа, разрешимые для обыкновенных сетей Петри, оказываются неразрешимыми. В [19] доказано, что для них неразрешима проблема достижимости. Неразрешимость проблемы ограниченности для сетей Петри с обнуляющими дугами установлена в [32].

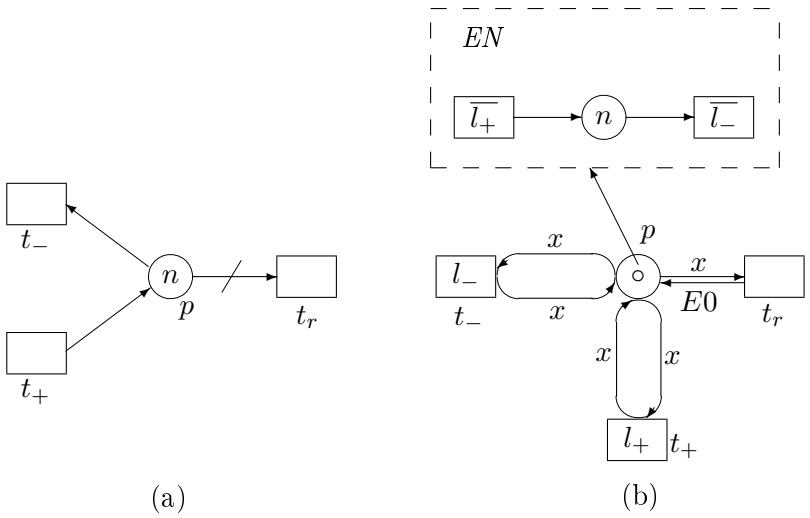


Рис. 3.14: Моделирование обнуляющей дуги

Поскольку сети Петри с обнуляющими дугами моделируются вложенными сетями Петри, из неразрешимости указанных проблем для сетей Петри с обнуляющими дугами следует их неразрешимость для вложенных сетей Петри.

Теорема 3.10. *Проблемы достижимости и $\preceq$-ограниченности неразрешимы для двухуровневых вложенных сетей Петри.*

Доказательство. Неразрешимость этих проблем для двухуровневых вложенных сетей Петри следует из неразрешимости их для сетей Петри с обнуляющими дугами и теоремы 3.9 о моделировании вложенных сетей сетями с обнуляющими дугами. □

Для доказательства неразрешимости проблемы живости для вложенных сетей Петри покажем, что для сетей Петри с обнуляющими дугами (так же, как и для обыкновенных сетей Петри, см. [8]) проблема достижимости может быть сведена к проблеме живости. Для этого сначала докажем следующее

Утверждение 3.3. *Для сетей Петри с обнуляющими дугами проблема достижимости произвольной разметки сводится к проблеме достижимости нулевой разметки, т. е. такой разметки, при которой все позиции сети пусты.*

Доказательство. Сведение выполняется аналогично тому, как это делается для обыкновенных сетей Петри. □

Теорема 3.11. *Для сетей Петри с обнуляющими дугами проблема достижимости нулевой разметки сводится к проблеме живости сети.*

Доказательство. Доказательство полностью аналогично доказательству соответствующего утверждения для обыкновенных сетей Петри (см. [8], стр.32). □

Отсюда получаем

Теорема 3.12. *Проблема живости сети неразрешима для двухуровневых вложенных сетей Петри.*

Доказательство. Следует непосредственно из теорем 3.11, 3.9 и неразрешимости проблемы достижимости для сетей Петри с обнуляющими дугами. □

Глава 4

Многоуровневые вложенные сети Петри

В многоуровневых вложенных сетях Петри элементные сети сами могут быть вложенными сетями Петри.

4.1 Определение структуры и поведения

Определение многоуровневой сети Петри является обобщением соответствующего определения для двухуровневых сетей.

Пусть $Var = \{v, \ldots\}$ — множество имен *переменных*, $Con = Con_{atom} \cup Con_{net} = \{c, \ldots\}$ — множество имен *констант*, состоящее из множества имен Con_{atom} *атомарных констант* и множества имен Con_{net} *сетевых констант*. Через $Atom$ будем обозначать множество $Var \cup Con$ *атомов*.

Язык выражений $Expr(Atom)$ над множеством атомов $Atom$ определяется как и раньше (см. раздел 3.2).

Множество меток $Lab =_{\text{def}} Lab_v \cup Lab_h$, где Lab_v — множество меток для вертикальной, Lab_h — для горизонтальной синхронизации, определяются так же, как и в случае двухуровневых сетей Петри.

Пусть A_{atom} есть множество атомарных фишек. Мы предполагаем, что A_{atom} содержит хотя бы один элемент – черную точку.

Определение 4.1. *Сетевой компонентой* будем называть сеть Петри высокого уровня $\mathfrak{N} = (N, \mathcal{L}, W, \Lambda)$, где

1) $N = (P, T, F)$ — сеть.

2) P — множество символов (позиций сети) с приписанной им арностью, $P \cap Atom = \emptyset$.

3) $\mathcal{L} = Expr(Atom)$ — язык выражений, определенный выше.

4) W — функция, сопоставляющая каждой дуге $(x, y) \in F$ некоторое выражение $W(x, y) = \overline{\theta} = (\theta_1, \ldots, \theta_n)$ размерности n, где $\theta_i \in \mathcal{L}$ $(1 \leq i \leq n)$ и n есть арность позиции, инцидентной дуге (x, y). При этом на выражения, приписанные входным дугам, накладываются те же ограничения, что и в случае системной сети двухуровневой сети Петри, а именно:

   - эти выражения не содержат констант из Con_{net} (т. е. только константы из Con_{atom});
   - всякая переменная встречается в выражении $W(p, t)$ не более одного раза;
   - для любых двух выражений $W(p_1, t)$ и $W(p_2, t)$, приписанных входным дугам одного и того же перехода t, $Var\,(W(p_1, t)) \cap Var\,(W(p_2, t)) = \emptyset$.

   На выражения, приписанные выходным дугам, как и раньше, накладывается ограничение на вхождение переменных, не встречающихся в соответствующих им входных выражениях. Если для некоторой дуги выражение опущено, то по умолчанию предполагается, что это $1 \in \mathbb{N}$.

5) Λ — частичная функция пометки переходов, помечающая некоторые переходы из T метками из $Lab_v \cup Lab_h$.

В частности, все выражения на дугах компоненты вложенной сети могут быть просто натуральными числами. Такая компонента является обыкновенной сетью Петри.

Разметку сетевой компоненты определим индуктивно.

Определение 4.2. Пусть $\overline{\mathfrak{N}} = (\mathfrak{N}_0, \ldots, \mathfrak{N}_k)$ — конечный набор сетевых компонент, A_{atom} — конечный набор атомарных констант.

1) Функция, сопоставляющая каждой позиции p сети $\mathfrak{N}_i$ ($1 \leq i \leq k$) некоторое конечное мультимножество $M(p)$ кортежей атомарных фишек из $a \in \mathrm{A}_{atom}$, есть разметка сетевой компоненты $\mathfrak{N}_i$ (относительно $\overline{\mathfrak{N}}$ и набора атомарных фишек A_{atom}). Размерность кортежей при этом должна совпадать с арностью позиции p. Пару $(\mathfrak{N}_i, M)$ будем называть *маркированной сетевой компонентой* или *сетевой фишкой*.

2) Пусть $\alpha_1, \ldots, \alpha_n$ — набор маркированных сетевых компонент. Тогда функция, сопоставляющая каждой позиции p сети $\mathfrak{N}_i$ некоторое конечное мультимножество $M(p)$ кортежей (соответствующей размерности) элементов множества $\{\alpha_1, \ldots, \alpha_n\} \cup \mathrm{A}_{atom}$ сетевых и атомарных фишек, есть разметка сетевой компоненты $\mathfrak{N}_i$ (относительно $\overline{\mathfrak{N}}$ и набора атомарных фишек A_{atom}), при этом пара $(\mathfrak{N}_i, M)$ образует *маркированную сетевую компоненту*.

Разметку сетевой компоненты удобно представлять в виде дерева.

Определение 4.3. Пусть $\overline{\mathfrak{N}} = (\mathfrak{N}_0, \ldots, \mathfrak{N}_k)$ — конечный набор сетевых компонент вложенной сети, A_{atom} — конечный набор атомарных констант, M — разметка для компоненты $\mathfrak{N}_i$ (относительно $\overline{\mathfrak{N}}$ и набора атомарных фишек A_{atom}).

Деревом $\mathcal{T}(M)$ разметки M назовем конечное помеченное корневое дерево, такое что:

- Корень дерева разметки помечен именем компоненты $\mathfrak{N}_i$.

- Корень $\mathfrak{N}_i$ имеет k_i потомков (по числу позиций компоненты $\mathfrak{N}_i$), помеченных именами позиций $p^i_1, \ldots, p^i_{k_i}$ сети $\mathfrak{N}_i$.

- Каждая вершина, помеченная именем позиции p^i_j, не имеет потомков, если $M(p^i_j) = \emptyset$, либо (в противном случае) имеет одного потомка, помеченного символом "$(\,, \ldots, \,)_n$" образования кортежа из n элементов, где n — арность позиции p^i_j.

- Вершина, помеченная символом "$(\,, \ldots, \,)_n$" образования кортежа из n элементов, имеет ровно n потомков, помеченных, соответственно, числами $1, 2, \ldots, n$.

- Вершина, помеченная натуральным числом (индексом в кортеже), имеет ровно одного потомка, который помечен либо именем атомарной фишки, либо именем одной из компонент $\mathfrak{N}_0, \ldots, \mathfrak{N}_k$. При этом вершина, помеченная именем атомарной фишки, является листом, а вершина с пометкой $\mathfrak{N}_j$ является корнем поддерева, являющегося деревом разметки для сетевой компоненты $\mathfrak{N}_j$.

- Существует взаимно-однозначное соответствие между кортежами фишек из мультимножества $M(p)$ и поддеревьями, порожденными потомками вершины p, такое, что если на i-й позиции в кортеже находится фишка α, то потомок вершины с пометкой i есть либо лист, помеченный α (в случае атомарной фишки), либо вершина, порождающая поддерево, задающее разметку сетевой фишки α.

Дерево разметки является коммутативным деревом, т. е. порядок расположения ветвей дерева не существен (именно с этой целью для компонент кортежа вводятся дополнительные вершины с указанием позиции). Легко заметить, что каждый лист дерева помечен либо именем позиции, либо именем атомарной фишки.

Легко видеть, что всякое поддерево дерева разметки с корнем, помеченным именем некоторой компоненты $\mathfrak{N}_j$, является деревом

разметки для $\mathfrak{N}_j$, т. е. задает маркированную сетевую компоненту — сетевую фишку. Соответственно, поддерево с корнем, помеченным символом "$(\,,\ldots,\,)_n$" образования кортежа из n элементов, задает кортеж маркированных сетевых компонент.

Дерево $\mathcal{T}(M)$ задает разметку M для компоненты $\mathfrak{N}$. Разметка M есть функция, сопоставляющая каждой позиции p сети $\mathfrak{N}$ мультимножество кортежей размерности n, где n — арность позиции p. Элементами кортежей являются маркированные сетевые компоненты. При этом элементами мультимножества $M(p)$ являются кортежи, соответствующие поддеревьям, корни которых являются непосредственными потомками вершины p (которая, в свою очередь, является непосредственным потомком корня $\mathfrak{N}$). Будем говорить, что разметка M содержит сетевую фишку $(\mathfrak{N}, M_{\mathfrak{N}})$, если дерево $\mathcal{T}(M)$ содержит поддерево с корнем $\mathfrak{N}$, являющееся деревом $\mathcal{T}(M_{\mathfrak{N}})$ разметки $M_{\mathfrak{N}}$.

Глубиной разметки M будем называть максимальное число помеченных именами сетевых компонент вершин на пути в $\mathcal{T}(M)$ от корня к некоторому листу.

Определение 4.4. Разметка M сетевой компоненты $\mathfrak{N}$ называется *допустимой*, если в дереве разметки $\mathcal{T}(M)$ на пути от корня к любому листу имя каждой сетевой компоненты $\mathfrak{N}_i$ встречается не более одного раза.

Легко видеть, что для заданной многоуровневой сети NPN глубина всех ее допустимых разметок ограничена.

Пусть $\overline{\mathfrak{N}} = (\mathfrak{N}_0, \ldots, \mathfrak{N}_k)$ — конечный набор сетевых компонент вложенной сети, A_{atom} — конечный набор атомарных констант (атомарных фишек), $Con = Con_{atom} \cup Con_{net}$ — конечное множество имен констант, встречающихся в выражениях, приписанных дугам в сетевых компонентах. *Интерпретация* $\mathcal{I}$ констант (над набором сетевых компонент $\overline{\mathfrak{N}}$ и множеством атомарных фишек A_{atom}) сопоставляет каждому имени константы из Con_{atom} атомарную фишку из A_{atom}, а имени константы из Con_{net} — мар-

кированную сетевую компоненту (одну из компонент $\mathfrak{N}_1, \ldots, \mathfrak{N}_k$ вместе с некоторой допустимой маркировкой).

Определение 4.5. Вложенная сеть Петри (NP-сеть) есть пятерка $NPN = (\overline{\mathfrak{N}}, \mathrm{A}_{atom}, \mathcal{I}, \mathfrak{N}_0, M_0)$, где

1) $\overline{\mathfrak{N}} = (\mathfrak{N}_0, \ldots, \mathfrak{N}_k)$ есть конечный набор сетевых компонент, причем имена всех позиций и переходов в $\overline{\mathfrak{N}}$ попарно различны;

2) A_{atom} — конечный набор атомарных фишек;

3) $\mathcal{I}$ — интерпретация констант, встречающихся в выражениях на дугах в $\overline{\mathfrak{N}}$; константе c функция $\mathcal{I}$ сопоставляет некоторую сетевую или атомарную фишку из $\mathrm{A}_{net}(\mathfrak{N}) \cup \mathrm{A}_{atom}$, при этом все разметки в сетевых фишках являются допустимыми;

4) $\mathfrak{N}_0$ — выделенная компонента, называемая *системной сетью*;

5) M_0 — некоторая допустимая разметка системной сети $\mathfrak{N}_0$ над $\overline{\mathfrak{N}}$ и A_{atom}. Разметка M_0 называется *начальной разметкой* вложенной сети NPN.

Для NP-сети NPN определим *граф вложенности* $G_{\lhd}(NPN)$.

Определение 4.6. Пусть $\overline{\mathfrak{N}} = (\mathfrak{N}_0, \ldots, \mathfrak{N}_k)$ — конечный набор сетевых компонент, A_{atom} — конечный набор атомарных фишек и $\mathcal{I}$ — интерпретация констант NP-сети NPN над множеством $\mathrm{A}_{net}(\mathfrak{N}) \cup \mathrm{A}_{atom}$. Граф вложенности $G_{\lhd}(NPN)$ есть ориентированный граф, вершинами которого являются сетевые компоненты $\mathfrak{N}_0, \ldots, \mathfrak{N}_k$. Дуга, направленная от вершины $\mathfrak{N}_i$ к вершине $\mathfrak{N}_j$, имеется в графе $G_{\lhd}(NPN)$ в том и только том случае, если выполняется одно из следующих условий:

1) В компоненте $\mathfrak{N}_i$ встречается приписанное дуге выражение, содержащее такую константу $c \in Con_{net}$, что $\mathcal{I}(c) = (\mathfrak{N}_j, M)$, где M — некоторая разметка сети $\mathfrak{N}_j$.

2) Для некоторой константы $c \in Con_{net}$ выполняется $\mathcal{I}(c) = (\mathfrak{N}, M)$, и в дереве $\mathcal{T}(M)$ имеется ориентированный путь от вершины $\mathfrak{N}_i$ к вершине $\mathfrak{N}_j$.

3) В дереве $\mathcal{T}(M_0)$ начальной разметки M_0 имеется ориентированный путь от вершины $\mathfrak{N}_i$ к вершине $\mathfrak{N}_j$.

Если в графе вложенности $G_{\lhd}(NPN)$ имеется ориентированный путь от вершины $\mathfrak{N}_i$ к вершине $\mathfrak{N}_j$, то будем говорить, что компонента $\mathfrak{N}_j$ является *элементом* компоненты $\mathfrak{N}_i$ (обозначение: $\mathfrak{N}_j \lhd \mathfrak{N}_i$).

Определение 4.7. NP-сеть *NPN*, граф вложенности $G_{\lhd}(NPN)$ которой не содержит ориентированных циклов, будем называть *многоуровневой* вложенной сетью Петри. В противном случае сеть *NPN* будем называть *рекурсивной* вложенной сетью Петри.

В рекурсивной NP-сети сетевая компонента может быть своим собственным элементом. Такие сети будут рассмотрены в следующей главе. В этой главе мы рассматриваем сети, в которых такая ситуация не допускается — многоуровневые NP-сети. Легко видеть, что двухуровневые NP-сети являются частным случаем многоуровневых, в которых все компоненты, кроме системной сети, являются обыкновенными сетями Петри.

Определим поведение NP-сети.

Для сетевой компоненты, являющейся обыкновенной сетью Петри, правила срабатывания переходов определяются как обычно. Мы пишем $M \xrightarrow{t} M'$, если переход t переводит разметку M в разметку M'.

Для сетевой компоненты, являющейся сетью Петри высокого уровня, понятия *означивания перехода*, *срабатывания перехода*

($M \xrightarrow{t[b]} M'$ обозначает срабатывание перехода t при означивании b, которое разметку M переводит в разметку M', при этом все фишки рассматриваются как атомарные объекты) и сетевых фишек, *задействованных в срабатывании*, определяются так же, как и для системной сети двухуровневой сети Петри.

В случае многоуровневой сети одна и та же сетевая компонента может играть роль системной сети по отношению к своим элементам и элементной сети по отношению к сетевым компонентам, элементом которых она является. Поэтому мы не будем различать системно- и элементно-автономные шаги сети и будем использовать для них общее название "автономный шаг срабатывания".

Определение 4.8. Для NP-сети *NPN* определяются следующие три вида *шагов срабатывания*:

- **автономный шаг:** Пусть M — разметка сети *NPN*, $(\mathfrak{N}, M_{\mathfrak{N}})$ — сетевая фишка в этой разметке. Пусть t — непомеченный переход сетевой компоненты $\mathfrak{N}$ (т. е. значение $\Lambda(t)$ не определено). Если t является активным при разметке $M_{\mathfrak{N}}$ и означивании b и $M_{\mathfrak{N}} \xrightarrow{t[b]} M'_{\mathfrak{N}}$, то такое срабатывание перехода t в сетевой компоненте $\mathfrak{N}$ называется автономным шагом вложенной сети *NPN*. Разметка M сети *NPN* при этом переходит в разметку M', которая получается из M заменой сетевой фишки $(\mathfrak{N}, M_{\mathfrak{N}})$ на фишку $(\mathfrak{N}, M'_{\mathfrak{N}})$ (обозначается $M[t[b]\rangle M'$ или просто $M[\,\rangle M'$). В том случае, когда $\mathfrak{N}$ есть обыкновенная сеть Петри, означивание переменных не нужно и оно, естественно, не указывается. Легко видеть, что при таком срабатывании разметки сетевых фишек в $(\mathfrak{N}, M_{\mathfrak{N}})$ (если таковые имеются) не меняются, но некоторые из них перемещаются из одной позиции в другую (копируются, исчезают), а также возможно появление новых сетевых фишек.

- **шаг горизонтальной синхронизации:** Пусть M — разметка сети *NPN*, $p \in P$ — некоторая позиция в некото-

рой сетевой компоненте $\mathfrak{N}$, входящей в разметку M и $\overline{\alpha} = (\alpha_1, \ldots, \alpha_n)$ — кортеж фишек, находящийся в позиции p при разметке M. Пусть далее $\alpha_i = (\mathfrak{N}_1, M_1), \alpha_j = (\mathfrak{N}_2, M_2)$ — две сетевые фишки в этом кортеже. Пусть в сетевой компоненте $\mathfrak{N}_1$ имеется переход t_1, такой что $\Lambda(t_1) = \lambda \in Lab_h$ и $M_1 \xrightarrow{t_1[b_1]} M_1'$ при некотором означивании b_1, т. е. t_1 — активный при разметке M_1 переход, и его срабатывание по правилам для сетей Петри высокого уровня приводит к разметке M_1'. Пусть при этом в сети $\mathfrak{N}_2$ имеется переход t_2 такой, что $\Lambda(t_1) = \overline{\lambda} \in Lab_h$ и $M_2 \xrightarrow{t_2[b_2]} M_2'$ при некотором означивании b_2. Пусть далее M' есть разметка сети NPN, получающаяся из M заменой сетевой фишки $\alpha_i = (\mathfrak{N}_1, M_1)$ на фишку $\alpha_i' = (\mathfrak{N}_1, M_1')$ и заменой фишки $\alpha_j = (\mathfrak{N}_2, M_2)$ на фишку $\alpha_j' = (\mathfrak{N}_2, M_2')$, т. е. M' — разметка, получающаяся в результате одновременного локального срабатывания двух переходов t_1 и t_2 в сетевых фишках α_i и α_j соответственно, при этом сетевые компоненты $\mathfrak{N}_1$ и $\mathfrak{N}_2$ остаются в той же позиции системной сети SN и на тех же местах в кортеже, в которых они находились. Такое синхронное срабатывание переходов t_1 и t_2 в сетевых компонентах $\mathfrak{N}_1$ и $\mathfrak{N}_2$ называется шагом горизонтальной синхронизации для вложенной сети NPN и обозначается $M[t_1[b_1]; t_2[b_2]\rangle M'$ или просто $M[\,\rangle M'$.

Если арность позиции p равна 1 и p содержит не кортежи, а единичные фишки, то для сетевой фишки $\alpha = (\mathfrak{N}_1, M_1)$ парная ей сетевая фишка с активным переходом, помеченным дополнительной меткой $\overline{\lambda}$, выбирается среди фишек, находящихся в той же позиции p при разметке M. Выбор парной фишки среди фишек, находящихся в других позициях сети, запрещается.

- **шаг вертикальной синхронизации**: Пусть M — разметка сети NPN, $(\mathfrak{N}, M_{\mathfrak{N}})$ — сетевая фишка, входящая в разметку M. Пусть t — переход сетевой компоненты $\mathfrak{N}$ такой,

что $\Lambda(t) = l \in Lab_v$, переход t является активным при разметке $M_{\mathfrak{N}}$ и означивании b и $M_{\mathfrak{N}} \xrightarrow{t[b]} M'_{\mathfrak{N}}$. Пусть далее $\{\alpha_1, \ldots, \alpha_k\} \subset \mathrm{A}_{net}$ есть множество задействованных в этом срабатывании перехода t сетевых фишек, причем $\alpha_1 = (\mathfrak{N}_1, M_1), \ldots, \alpha_k = (\mathfrak{N}_k, M_k)$. Пусть также для каждого $1 \leq i \leq k$ в сети $\mathfrak{N}_i$ найдется переход t_i, такой что t_i активен в $\mathfrak{N}_i$ при разметке M_i и означивании b_i, $M_i \xrightarrow{t_i[b_i]} M'_i$ (в соответствии с правилами срабатывания для сетей Петри высокого уровня) и $\Lambda(t_i) = \bar{l} \in Lab_v$.

Поскольку условия срабатывания перехода t в $\mathfrak{N}$ не зависят от внутренней разметки сетевых фишек, переход t является активным также и при разметке $M^{\star}_{\mathfrak{N}}$, полученной из M заменой в каждой задействованной в срабатывании перехода t сетевой фишке $\mathfrak{N}_i$ разметки M_i на разметку M'_i для всех $1 \leq i \leq k$. При этом в качестве означивания берется функция b' такая, что для всякой переменной x из некоторого выражения на входной дуге $b'(x) = (\mathfrak{N}_i, M'_i)$, если $b(x) = (\mathfrak{N}_i, M_i)$.

Пусть $M^{\star}_{\mathfrak{N}} \xrightarrow{t[b]} M''_{\mathfrak{N}}$. Тогда результатом одновременного срабатывания перехода t в сетевой компоненте $\mathfrak{N}$ и дополнительных к нему переходов в задействованном в этом срабатывании сетевых компонентах является разметка M' сети *NPN*, получающаяся из M заменой сетевой фишки $(\mathfrak{N}, M_{\mathfrak{N}})$ на фишку $(\mathfrak{N}, M''_{\mathfrak{N}})$. Такое синхронное срабатывание перехода t (при означивании b) в системной сети и переходов $t_1, \ldots, t_k$ (при означиваниях $b_1, \ldots, b_k$ соответственно) в сетях $\alpha_1, \ldots, \alpha_k$ называется шагом вертикальной синхронизации для NP-сети *NPN* и обозначается $M[t[b]; t_1[b_1]; \ldots; t_k[b_k]\rangle M'$ или просто $M[\,\rangle M'$.

Замечание 4.1. Заметим, что вертикальная синхронизация переходов производится только для сетевых компонент двух смежных уровней дерева разметки. Это соответствует принципу распреде-

ленности системы.

Исполнением для NP-сети NPN назовем конечную или бесконечную последовательность шагов $M_0[\,\rangle M_1[\,\rangle \ldots$, где M_0 есть начальная разметка сети NPN.

Как обычно, разметку M сети NPN назовем достижимой, если существует исполнение $M_0[\,\rangle M_1[\,\rangle \ldots [\,\rangle M_k$, где $M = M_k$.

Утверждение 4.1. *Пусть NPN — многоуровневая NP-сеть. Тогда всякая достижимая в NPN разметка является допустимой.*

Доказательство. Пусть M — достижимая разметка сети NPN. Тогда существует исполнение $M_0[\,\rangle M_1[\,\rangle \ldots [\,\rangle M_k$, где $M = M_k$.

Доказательство проведем индукцией по k. Начальная разметка M_0 является допустимой по определению. Пусть M_i — допустимая разметка и $M[\,\rangle M_{i+1}$. Покажем, что разметка M_{i+1} также допустима.

Рассмотрим три случая, соответствующих трем типам шагов срабатывания. При автономном шаге срабатывания новые пути в дереве разметки могут появиться только в результате появления новых сетевых фишек в некоторых позициях сетевой компоненты. В дереве разметки этому будет соответствовать "навешивание" новых поддеревьев на вершины, помеченные именами позиций и/или операцией образования кортежа. При этом эти новые сетевые фишки либо уже встречались в других позициях (кортежах других позиций) этой же сетевой компоненты, либо являются значениями сетевых констант. Если после добавления новых сетевых фишек в дереве разметки появился бы ориентированный путь от вершины с пометкой $\mathfrak{N}_i$ к вершине с такой же пометкой, то в первом случае это бы противоречило допустимости предыдущей разметки M_i. Во втором случае, когда добавляется сетевая фишка, являющаяся значением сетевой константы, наличие такого пути противоречило бы условию отсутствия ориентированных циклов в графе вложенности $G_{\lhd}(NPN)$. Действительно, в силу условия 2 определения 4.6 такой путь не может проходить внутри

новой сетевой фишки и в силу условия 2 он не может соединять вершину поддерева сетевой фишки и другую вершину в дереве разметки.

Случаи горизонтальной и вертикальной синхронизации рассматриваются аналогично. □

Нетрудно заметить, что для заданной многоуровневой сети NPN глубина всех ее допустимых (а, значит, и всех достижимых) разметок ограничена и не превышает длину максимального ориентированного пути с началом в вершине, соответствующей системной сети $\mathfrak{N}_0$, в графе вложенности $G_{\lhd}(NPN)$. Для рекурсивной вложенной сети глубина достижимых разметок, вообще говоря, не ограничена.

Далее разметками многоуровневой NP-сети будем называть допустимые разметки.

4.2 Многоуровневые сети как вполне структурированные системы переходов

Частичный порядок на множестве разметок многоуровневой NP-сети является обобщением частичного порядка $\preceq$ для двухуровневых сетей. Поэтому мы используем для него то же самое обозначение $\preceq$.

Пусть NPN — многоуровневая NP-сеть. Определим бинарное отношение $\preceq$ на множестве ее разметок $\mathfrak{M}(NPN)$.

Определение 4.9. Пусть $M_1, M_2 \in \mathfrak{M}(NPN)$ — разметки многоуровневой NP-сети NPN. Отношение $M_1 \preceq M_2$ определим индуктивно:

1) Для обыкновенной сети Петри PN и двух ее разметок M_1, $M_2 \in \mathfrak{M}(PN)$, $M_1 \preceq M_2$ в том и только том случае, когда для любой позиции $p \in P$ сети PN выполняется $M_1(p) \leq M_2(p)$.

2) Для сети Петри PN высокого уровня (одноуровневой вложенной сети Петри, в которой все фишки являются атомарными) и двух ее разметок $M_1, M_2 \in \mathfrak{M}(PN)$, $M_1 \preceq M_2$ в том и только том случае, когда для любой позиции $p \in P$ сети PN выполняется $M_1(p) \leq_{\mathcal{M}^1} M_2(p)$, где $\leq_{\mathcal{M}^1}$ — 1-упорядочение мультимножеств (см. определение 1.4).

3) Для многоуровневой NP-сети NPN и двух ее разметок M_1, $M_2 \in \mathfrak{M}(NPN)$, $M_1 \preceq M_2$ в том и только том случае, когда для любой позиции $p \in P_{SN}$ системной сети SN сети NPN существует инъективное отображение $\jmath_p : M_1(p) \to M_2(p)$ такое, что для $\overline{\alpha} = (\alpha_1, \ldots, \alpha_n) \in M_1(p)$ если $\jmath_p(\overline{\alpha}) = \overline{\alpha}\prime = (\alpha'_1, \ldots, \alpha'_n)$, то для всех $i = 1, \ldots, n$ выполняется $\alpha \preceq \alpha'$.

Нетрудно заметить, что для двухуровневых сетей определение отношения $\preceq$, приведенное выше, эквивалентно определению 3.3, данному в предыдущей главе.

Утверждение 4.2. *Отношение $\preceq$ на множестве разметок многоуровневой NP-сети NPN является частичным порядком.*

Доказательство. Рефлексивность и транзитивность отношения $\preceq$ очевидны. Докажем антисимметричность.

Доказательство проведем индукцией по глубине разметки сети. Предположим, что для разметок глубины не больше n утверждение доказано.

Пусть глубина разметок M_1 и M_2 сети NPN не превышает $n+1$ и выполняется $M_1 \preceq M_2$, $M_2 \preceq M_1$. Пусть $p \in P_{SN}$ — некоторая позиция системной сети SN. Рассмотрим мультимножества $M_1(p)$ и $M_2(p)$ кортежей элементов (фишек). По определению порядка $\preceq$ либо $M_1(p)$ и $M_2(p)$ одновременно пусты, либо существуют два инъективных отображения $\imath_p : M_1(p) \to M_2(p)$ и $\jmath_p : M_2(p) \to M_1(p)$.

Из инъективности этих отображений следует, что мощности мультимножеств $M_1(p)$ и $M_2(p)$ равны. Пусть $\overline{\alpha} = (\alpha_1, \ldots, \alpha_n) \in$

$M_1(p)$. Рассмотрим i-ую компоненту кортежа. По определению если α_i $(1 \leq n)$ — атомарная фишка, то $\imath_p(\alpha_i) = \alpha_i$.

Для сетевой фишки $\alpha_i = (EN, m)$ имеем $\imath_p(\alpha_i) = \alpha_i' = (EN, m')$, где $m \leq m'$. Тогда $\jmath_p(\alpha_i') = (EN, m'')$, где $m' \leq m''$. Продолжая это построение, получим последовательность разметок $m \leq m' \leq m'' \leq \ldots$. В силу конечности мультимножеств $M_1(p)$ и $M_2(p)$ эта последовательность стабилизируется, начиная с некоторого номера. Следовательно, для двух разметок m_1 и m_2 сети EN выполняется $m_1 \preceq m_2$ и $m_2 \preceq m_1$. Поскольку глубина разметок m_1, m_2 не превышает n, в силу индукционного предположения, получаем $m_1 = m_2$. Аналогично, равными оказываются и другие компоненты кортежей, а, следовательно, и сами кортежи.

Таким образом, для каждого кортежа из $M_1(p)$ найдется такой же кортеж в $M_2(p)$. Отсюда, в силу равномощности мультимножеств $M_1(p)$ и $M_2(p)$, получаем, что мультимножества $M_1(p)$ и $M_2(p)$ равны, что и требовалось доказать. □

Ранее было доказано (см. утверждение 3.2), что отношение $\preceq$ на множестве разметок двухуровневой NP-сети соответствует изоморфному вложению деревьев разметок (относительно равенства на множестве меток, которое в случае конечности домена является правильным частичным порядком). Аналогичное утверждение справедливо и для многоуровневых NP-сетей.

Утверждение 4.3. *Пусть $M_1, M_2 \in \mathfrak{M}(NPN)$ — разметки многоуровневой NP-сети NPN и $\mathcal{T}(M_1), \mathcal{T}(M_2)$ — соответствующие им деревья разметок. Тогда $M_1 \preceq M_2$ в том и только том случае, когда дерево $\mathcal{T}(M_1)$ изоморфно вкладывается в дерево $\mathcal{T}(M_2)$ (относительно равенства на множестве меток), т. е. существует отображение $\varphi : Nodes(\mathcal{T}(M_1)) \to Nodes(\mathcal{T}(M_2))$ такое, что*

1) φ — инъективна;

2) φ монотонна и переводит смежные вершины в смежные,

т. е. для любых смежных вершин $\alpha, \beta \in Nodes\,(t)$ если $\alpha \leq \beta$, то $\varphi(\alpha) \leq \varphi(\beta)$ и вершины $\varphi(\alpha), \varphi(\beta)$ также смежны;

3) *для любой вершины $\alpha \in Nodes\,(t)$: $\mathcal{L}(\alpha) = \mathcal{L}'(\varphi(\alpha))$, где $\mathcal{L}, \mathcal{L}'$ — функции пометки вершин для деревьев t и t' соответственно.*

Доказательство. Доказательство проведем индукцией по глубине разметки (высоте дерева разметки). Для двухуровневых NP-сетей утверждение было доказано раньше (см. Утв. 3.2). Пусть утверждение верно для разметок глубины не больше n. Пусть теперь $M_1 \preceq M_2$ — две разметки глубины не больше $n+1$.

Отображение φ вершин дерева $\mathcal{T}(M_1)$ в вершины дерева $\mathcal{T}(M_2)$ строится следующим образом. Корню дерева $\mathcal{T}(M_1)$ сопоставляется корень дерева $\mathcal{T}(M_2)$. Вершинам в $\mathcal{T}(M_1)$, соответствующим позициям системной сети, сопоставляются такие же вершины в дереве $\mathcal{T}(M_2)$. Вершинам, соответствующим фишкам в разметке M_1, соответствующие вершины в $\mathcal{T}(M_2)$ определяются отображениями $\jmath_p$ из определения упорядочения $\preceq$. Если некоторая вершина a в $\mathcal{T}(M_1)$ соответствует атомарной фишке, то она, как и вершина $\varphi(a)$ в $\mathcal{T}(M_1)$, является листом. Если же a соответствует сетевой фишке, то она является корнем поддерева некоторой разметки m, и для $m' = \jmath_p(m)$ выполняется $m \preceq m'$. Тогда в силу индукционного предположения поддерево с корнем m в $\mathcal{T}(M_1)$ изоморфно вкладывается в поддерево с корнем m' в $\mathcal{T}(M_1)$. Отображение φ на этих поддеревьях определяется в соответствии с этим вложением.

Обратное утверждение доказывается аналогично. □

Теорема 4.1. *Отношение $\preceq$ на множестве разметок многоуровневой NP-сети NPN является правильным квазиупорядочением.*

Доказательство. В силу утверждения 4.3 отношение $\preceq$ на разметках сети соответствует отношению изоморфного вложения де-

ревьев разметок (относительно равенства на множестве меток). Поскольку для конкретной вложенной сети множество меток в ее деревьях разметок конечно, отношение равенства меток является правильным частичным порядком. Кроме того, высота деревьев разметок для многоуровневой NP-сети ограничена. Тогда по теореме 1.2 об изоморфном вложении деревьев отношение $\preceq$ на множестве разметок многоуровневой NP-сети также является правильным квазиупорядочением. □

Теорема 4.2. *Пусть NPN — NP-сеть, $M_1 \preceq M_2$ — разметки сети NPN. Если $M_1[Y[b]\rangle M_1'$, где $Y[b]$ есть либо означенный переход $t[b]$ (в случае системно-автономного шага), либо набор переходов вместе с некоторым означиванием $t[b]; t_1, \ldots, t_k$ (в случае шага вертикальной синхронизации), то найдется такое означивание b', что $M_2[Y[b']\rangle M_2'$ и $M_1' \preceq M_2'$.*

Доказательство. Пусть $M_1 \preceq M_2$ и $\{\jmath_p\}_{p \in P_{SN}}$ — соответствующее этому порядку семейство инъективных отображений. Доказательство проведем индукцией по максимальной глубине разметок M_1 и M_2. Для разметок глубины 1 утверждение теоремы очевидно.

Пусть максимальная глубина M_1 и M_2 равна k. Поскольку все сетевые фишки в разметках M_1 и M_2 имеют меньшую глубину, предполагаем, что для них утверждение уже доказано.

Индукционный шаг доказывается разбором случаев в зависимости от вида шага срабатывания.

1. Пусть $M_1[t[b]\rangle M_1'$ — автономное срабатывание перехода t при означивании b в системной сети SN NP-сети NPN, т. е. $\forall p \in {}^\bullet t : W(p,t)(b) \subseteq M_1(p)$ и $M_1'(p) = (M_1(p) \setminus W(p,t)(b)) \cup W(t,p)(b)$.

Тогда в качестве функции означивания для разметки M_2 возьмем функцию $b' : Var \to \mathrm{A}$ такую, что $b'(x) =_{\mathrm{def}} \jmath_p(b(x))$. Данное определение корректно, поскольку для $p_1 \neq p_2 \in P_{SN}$, в силу ограничений на выражения, приписанные входным дугам переходов в системной сети, выполняется $Var(W(p_1,t)) \cap Var(W(p_2,t)) = \emptyset$.

Поскольку выражения на входных дугах не содержат констант, и каждая переменная входит в такое выражение не более одного раза (в силу тех же ограничений), то из $\jmath_p(M_1(p)) \subseteq M_2(p)$ следует $\forall p \in {}^\bullet t : W(p,t)(b \circ \jmath_p) \subseteq \jmath_p(M_1(p)) \subseteq M_2(p)$, т. е. переход t является активным при разметке M_2 и означивании b'.

Тогда результат срабатывания t при означивании b' есть разметка $M_2'(p) = (M_2(p) \setminus W(p,t)(b \circ \jmath_p)) \cup W(t,p)(b \circ \jmath_p)$. Поскольку $W(t,p)(b \circ \jmath_p)$ отличается от $W(t,p)(b)$ только возможно большей разметкой в некоторых элементных сетях, нетрудно видеть, что $\jmath_p$, доопределенное на тех кортежах α, на которых оно не было определено, как $\jmath_p(\alpha) = \alpha$, является инъективным отображением, в силу существования которого $M_1' \preceq M_2'$.

2. Пусть $M_1 \preceq M_2$ и $M_1[\rangle M_1'$ — автономное срабатывание перехода t в сетевой фишке $\alpha_i = (EN, m_1)$, находящейся в некотором кортеже в позиции p системной сети, либо срабатывание перехода во внутренней для α_i сетевой фишке, и $m_1[t\rangle m_1'$.

Пусть далее $\jmath_p(\alpha_i) = (EN, m_2)$, где $m_2 \succeq m_1$. Тогда по индукционному предположению переход t активен в сети EN при разметке m_2, и, следовательно, возможно элементное срабатывание этого перехода в сети NPN при разметке M_2. Пусть $m_2[t\rangle m_2'$ и, соответственно, $M_2[t\rangle M_2'$. Разметка M_2' отличается от M_2 только разметкой внутренней сетевой фишки α_i. Чтобы показать, что $M_1' \preceq M_2'$, построим инъективную функцию $\tilde{\jmath_p} : M_1'(p) \to M_2'(p)$ следующим образом:

$$\tilde{\jmath_p}(\alpha) = \begin{cases} (EN, m_2') & \text{при } \alpha = (EN, m_2), \\ \jmath_p(\alpha) & \text{при } \alpha \neq (EN, m_2). \end{cases}$$

Для позиции $p' \neq p$ полагаем $\tilde{\jmath_p} \equiv \jmath_p$. Нетрудно видеть, что так определенное семейство отображений $\{\tilde{\jmath_p}\}_{p \in P_{SN}}$ удовлетворяет условию инъективности, и, следовательно, $M_1' \preceq M_2'$.

3. Пусть $M_1 \preceq M_2$ и $M_1[t_1, t_2\rangle M_1'$ — шаг горизонтальной синхронизации сети NPN, т. е. одновременное срабатывание переходов t_1 и t_2 в сетевых фишках $\alpha_i = (EN_1, m_1), \alpha_j = (EN_2, m_2)$,

находящихся при разметке M в некотором кортеже в позиции p системной сети.

Пусть теперь $\jmath_p(\alpha_i) = (EN, m_3)$ и $\jmath_p(\alpha_j) = (EN, m_4)$, где $m_3 \succeq m_1$ и $m_4 \succeq m_2$. В силу индукционного предположения, найдутся состояния m'_3 и m'_4, такие что $m_3[t\rangle m'_3$ в сети EN_1 и $m_4[t\rangle m'_4$ в сети EN_2, и, соответственно, возможен шаг горизонтальной синхронизации $M_2[t\rangle M'_2$ в сети NPN. При этом разметка M'_2 получается из M_2 заменой разметок m_3 и m_4 внутренних сетевых фишек α_i, α_j на разметки m'_3 и m'_4 соответственно. Инъективное отображение $\tilde{\jmath_p} : M'_1(p) \to M'_2(p)$, определяющее упорядоченность $M'_1 \preceq M'_2$, строится аналогично предыдущему случаю. Полагаем

$$\tilde{\jmath_p}(\alpha) = \begin{cases} (EN, m'_3) & \text{при } \alpha = (EN, m_3), \\ (EN, m'_4) & \text{при } \alpha = (EN, m_4), \\ \jmath_p(\alpha) & \text{при } \alpha \neq (EN, m_3) \text{ и } \alpha \neq (EN, m_4). \end{cases}$$

Для позиции $p' \neq p$ полагаем $\tilde{\jmath_p} \equiv \jmath_p$.

Если арность позиции p равна 1 и p содержит не кортежи, а единичные фишки, то все приведенные рассуждения также остаются верными.

4. Пусть $M_1 \preceq M_2$ и $M_1[\,\rangle M'_1$ — *шаг вертикальной синхронизации* в системной сети SN сети NPN, т. е. в SN имеется переход t, активный при разметке M_1 и некотором означивании b. Пусть его автономное срабатывание приводит к разметке M'. Как уже было доказано в п.1 этого доказательства, отсюда следует, что переход t является активным при разметке M_2 и некотором означивании b' и $M_2[t[b]\rangle M'_2$, где $M_2 \preceq M'_2$. Далее, для каждой системной фишки $\alpha_i = (EN_i, m_i)$ в позиции p, задействованной в b-срабатывании перехода t в системной сети SN, выполняется $m_i[t_i\rangle m'_i$. По индукционному предположению для автономных шагов из $M_1 \preceq M_2$ следует, что в сети $\jmath_p(\alpha_i) = (EN, m_{i'})$, где $m_{i'} \succeq m_i$, переход t_i активен, и, следовательно, возможно автономное срабатывание этого перехода в сети NPN при разметке M_2. Результатом срабаты-

вания t_i является локальная разметка $m'_{i'}$ сетевой фишки такая, что $m'_i \preceq m'_{i'}$.

Таким образом, в сети NPN возможен шаг вертикальной синхронизации при означивании b'. Результатом его является разметка M''_2, полученная из разметки M'_2 путем замены сетевой фишки $\alpha_i = (EN_i, m'_i)$ на фишку $\alpha'_i = (EN_i, m'_{i'})$ (имеющую большую разметку) для всех $1 \leq i \leq k$. Нетрудно видеть, что в результате замены локальных разметок некоторых сетевых фишек на большие разметки мы получим разметку M''_2, удовлетворяющую $M''_2 \preceq M'_2$, и, следовательно, $M''_2 \preceq M'_2$, что и требовалось доказать. □

Теорема 4.3. *Пусть NPN — многоуровневая NP-сеть, $\preceq$ — частичный порядок на множестве $\mathfrak{M}(NPN)$ разметок сети NPN, определенный выше. Тогда $\langle NPN, \preceq \rangle$ является вполне структурированной системой переходов.*

Доказательство. Следует из теоремы 4.1 о правильности квазиупорядочения $\preceq$ и теоремы 4.2 о выполнении свойства совместимости отношений перехода $\preceq$ для многоуровневых вложенных сетей Петри. □

4.3 Алгоритмы анализа для многоуровневых сетей

В предыдущем разделе было показано, что многоуровневые NP-сети, так же как и двухуровневые, являются вполне структурированными системами переходов. В этом разделе мы покажем, как результаты по анализу семантических свойств, изложенные в разделе 3.5, можно распространить на многоуровневые NP-сети.

4.3.1 Анализ дерева покрытия

Покажем, что для любой многоуровневой NP-сети NPN дерево покрытия может быть эффективно построено. Для этого необхо-

димо проверить, что множество $Succ(M)$ всех последующих разметок для разметки M во вложенной сети NPN вычислимо. Действительно, для данной разметки M множество возможных означиваний переходов в сети NPN конечно и может быть построено прямым перебором, следовательно, конечно и может быть эффективно построено множество активных означенных переходов сети NPN. Далее, в силу того, что выражения на выходных дугах всякого перехода t не содержат переменных, не встречавшихся в выражениях на входных дугах этого перехода, означивание перехода t однозначно определяет результат срабатывания перехода t, который вычисляется по известным правилам срабатывания перехода. Таким образом, множество $Succ(M)$ эффективно вычислимо для любой разметки M сети NPN.

Конечность множества $Succ(M)$ означает конечность ветвления в дереве покрытия сети NPN. Тогда в силу свойства вполне структурированности дерево покрытия для многоуровневой сети Петри всегда конечно и, следовательно, может быть эффективно построено.

Отношение $\preceq$ для многоуровневых NP-сетей, очевидно, разрешимо. Действительно, для сравнения двух разметок M_1 и M_2 сети NPN достаточно построить соответствующие им конечные деревья разметки $\mathcal{T}(M_1)$ и $\mathcal{T}(M_2)$ и проверить, что одно из них изоморфно вкладывается в другое. Для этого достаточно убедиться, что каждой корневой ветви “меньшего” дерева соответствует точно такая же корневая ветвь в “большем” дереве.

По утверждению 1.7 из вычислимости $Succ(M)$ и разрешимости $\preceq$ следует, что дерево покрытия для многоуровневой NP-сети может быть эффективно построено. Это позволяет распространить на случай многоуровневой NP-сети методы анализа ряда семантических свойств, изложенные в параграфе 3.5.1.

Теорема 4.4. *Пусть NPN — многоуровневая NP-сеть, C — некоторый направленный вверх конус (относительно частичного порядка $\preceq$) в множестве разметок сети NPN. Утверждение тем-*

поральной логики $s \models (\exists)\Box C$ *истинно для сети NPN тогда и только тогда, когда покрывающее дерево NP-сети NPN содержит путь от корня к листу, в котором все вершины имеют пометки, принадлежащие* C.

Доказательство. Следует из утверждения 1.8. □

Теорема 4.5. *Проблемы поддержки управляющего состояния и неизбежности разрешимы для многоуровневых NP-сетей.*

Доказательство. Следует из теоремы 4.3, утверждения 1.9, разрешимости отношения $\preceq$ и вычислимости *Succ* для многоуровневых NP-сетей. □

Теорема 4.6. *Проблема останова разрешима для многоуровневых NP-сетей.*

Доказательство. Как и в случае двухуровневых NP-сетей, для решения проблемы останова достаточно построить покрывающее дерево сети и проверить, являются ли пометки на его листьях тупиковыми разметками. Если все пометки на листьях покрывающего дерева — тупиковые, то всякое исполнение сети завершается. Если же найдется нетупиковая пометка, то для данной сети существует бесконечное исполнение. □

4.3.2 Метод насыщения

Для решения проблемы покрытия для многоуровневых NP-сетей, как и ранее, будем применять метод насыщения. Для этого нужно показать, что всякая многоуровневая NP-сеть имеет эффективный предбазис, т. е. для любой разметки $M \in \mathfrak{M}(NPN)$ может быть эффективно вычислен конечный базис множества $Pred(\uparrow M)$.

Лемма 4.1. *Для любой многоуровневой NP-сети NPN и перехода* t *этой сети множество* ${}^\circ t$ *может быть эффективно построено.*

Доказательство. В качестве доказательства приведем
Алгоритм построения множества $^\circ t$ минимальных разметок, при которых переход t может сработать.

1. Если t — переход в системной сети, то множество $^\circ t$ строится так же, как в случае двухуровневой сети (см. Лемму 3.4).

2. Пусть t — непомеченный переход в сетевой компоненте $\mathfrak{N}$. Минимальную разметку $M_{\mathfrak{N}}$ сетевой компоненты $\mathfrak{N}$, при которой переход t может сработать в $\mathfrak{N}$, построим так же, как при построении множества $^\circ t$ для двухуровневой сети. Тогда множество минимальных разметок $^\circ t$ строится путем объединения следующих множеств:

- множество всех разметок, при которых одна из позиций системной сети содержит сетевую фишку (EN, m), а все остальные позиции пусты;
- множество всех разметок, при которых все позиции системной сети пусты, кроме одной, содержащей некоторую сетевую фишку EN_1 с разметкой, в которой, в свою очередь, все позиции пусты, кроме одной, содержащей сетевую фишку (EN, m);
- разметки, при которых все позиции системной сети пусты, а одна содержит некоторую сетевую фишку EN_1, которая содержит некоторую другую сетевую фишку EN_2, содержащую, в свою очередь, сетевую фишку (EN, m);
- и т.д.

Поскольку в разметках многоуровневых сетей элементные сети в цепочках вложенных друг в друга фишек могут встречаться не более одного раза, это построение конечно. С помощью графа вложенности сети перебор можно уменьшить, рассматривая только возможные для данной сети вложения одних сетевых компонент в другие.

Множество ${}^\circ t$ строится как набор всех возможных разметок, для которых $M(p)$ принимает одно из конечного числа описанных выше значений.

3. Пусть t — помеченный меткой для вертикальной синхронизации переход в системной сети. Тогда множество ${}^\circ t$ строится так же, как и в предыдущем случае, с той только разницей, что "последней" в рассмотренной цепочке вложений оказывается не одна сетевая фишка, а двухуровневая сеть с элементами, помеченными дополнительными метками. Сама эта сеть строится так же, как и в случае двухуровневых сетей.

4. И, наконец, если t — помеченный меткой для горизонтальной синхронизации переход в элементной сети EN сети NPN, то множество минимальных разметок ${}^\circ t$ строится так же, как и в случае **2**, с той только разницей, что вместо сети EN везде берется пара сетей, содержащих помеченные взаимно дополнительными метками переходы. □

Лемма 4.2. *Для любой многоуровневой NP-сети NPN и произвольной разметки $M \in \mathfrak{M}(NPN)$ существует эффективная процедура построения конечного базиса множества $Pred(\uparrow M)$.*

Доказательство. Пусть NPN — некоторая двухуровневая сеть, $M \in \mathfrak{M}(NPN)$ — ее разметка. В предыдущей лемме было доказано, что множество ${}^\circ t$ минимальных относительно $\preceq$ разметок сети NPN, при которых переход t является активным, может быть эффективно построено.

Через t° обозначим множество разметок, получаемых в результате всех возможных срабатываний перехода t (возможно, совместно с другими переходами) на разметках из множества ${}^\circ t$. Очевидно, что множество t° конечно и может быть получено применением перехода t ко всем разметкам из ${}^\circ t$.

Конечный базис $lub(M, t^\circ)$ верхнего конуса $C = (\uparrow M) \cap (\uparrow t^\circ)$ разметок строится аналогично случаю двухуровневых сетей (см. Лемму 3.4). При этом при замене двух сетевых фишек (EN, m_1)

и (EN, m_2) с одинаковой сетевой структурой и разными пометками (ноль и единица) одной фишкой (EN, m), новая разметка m строится как $lub(m_1, m_2)$, т. е. итеративно. Процесс построения заканчивается, потому что глубина разметок при переходе от сетевых фишек к их разметкам уменьшается, и, в конце концов, мы дойдем до двухуровневых сетей.

Как и в случае двухуровневых сетей, эффективное построение конечного базиса множества $Pred(\uparrow M)$ по конечному базису $lub(M, t^\circ)$ задается формулой

$Pred(\uparrow M) = \cup_t Pred_t(\uparrow lub(M, t^\circ)) = \uparrow(\cup_t Pred_t(lub(M, t^\circ)))$. □

Утверждение 4.4. *Существует эффективная процедура построения конечного базиса множества $Pred^*(C)$ по произвольному заданному своим конечным базисом верхнему конусу C в множестве разметок $\mathfrak{M}(NPN)$ многоуровневой NP-сети NPN.*

Доказательство. Следует из утверждения 1.13 и того, что всякая многоуровневая NP-сеть имеет эффективный предбазис. □

Теорема 4.7. *Проблема покрытия разрешима для многоуровневых NP-сетей.*

Доказательство. Следует из теоремы 1.4, эффективности предбазиса и разрешимости отношения порядка $\preceq$ для многоуровневых NP-сетей. □

Теорема 4.8. *Проблема потенциальной живости перехода разрешима для многоуровневых NP-сетей.*

Доказательство. Пусть NPN — многоуровневая NP-сеть, t — некоторый переход в NPN. Через $^\circ t$ обозначим множество минимальных относительно $\preceq$ разметок сети NPN, при которых переход t является активным. В силу правильности упорядочения $\preceq$, множество $^\circ t$ конечно. Оно может быть эффективно построено путем процедуры, приведенной выше.

Переход t потенциально живой в том и только том случае, когда достижима некоторая разметка, принадлежащая верхнему конусу с базисом ${}^\circ t$. Поэтому по теореме 4.7 о разрешимости проблемы покрытия проблема потенциальной живости перехода для многоуровневых NP-сетей также разрешима. □

Разметка M сети Петри PN называется *t-тупиковой* для некоторого перехода t, если для любой достижимой из M разметки M' переход t не может сработать в M'.

Теорема 4.9. *Пусть NPN — многоуровневая NP-сеть, t — некоторый переход в NPN. Тогда для произвольной разметки M можно эффективно установить, является ли она t-тупиковой.*

Доказательство. Разметка M является t-тупиковой тогда и только тогда, когда $M \in \uparrow {}^\circ t$, т. е. M принадлежит верхнему конусу с базисом ${}^\circ t$. В доказательстве предыдущего утверждения приведен алгоритм построения конечного базиса конуса $\uparrow {}^\circ t$. Таким образом, проблема проверки того, является ли данная разметка t-тупиковой, сводится к проблеме покрытия, разрешимость которой показана в теореме 4.7. □

Глава 5

Рекурсивные вложенные сети Петри

5.1 Пример рекурсивной вложенной сети

Рекурсивная вложенная сеть Петри может порождать свою собственную копию в качестве элемента (непосредственного или в качестве потомка). Рассмотрение рекурсивных вложенных сетей мы начнем с небольшого примера рекурсивной вложенной сети Петри RN, симулирующей последовательность внутренних вызовов рекурсивной процедуры (как, например, при вычислении функции факториала $fact(0) = 1$; $fact(n) = n \cdot fact(n-1)$ при $n \geq 1$).

Рекурсивная NP-сеть RN, изображенная на рисунке 5.1, состоит из системной сети F_0 и элементной сети F. Константа F, приписанная дуге, интерпретируется как элементная сеть F с начальной разметкой, изображенной на рисунке. Значения переменной x — сеть F с различными (достижимыми) маркировками.

Исполнение рекурсивной NP-сети RN происходит следующим образом. Первоначально системная сеть имеет единственную фишку (черную точку) в позиции p_1. Первый шаг выбирается недетерминированно и содержательно зависит от начального зна-

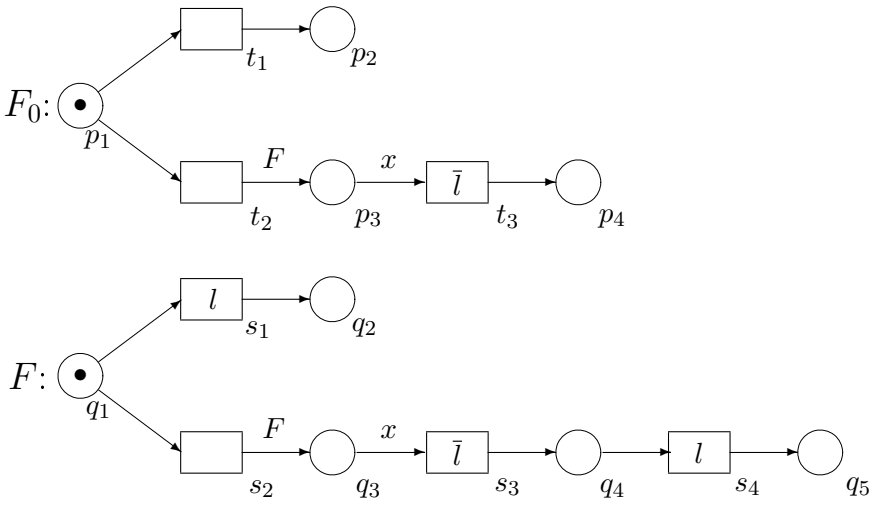


Рис. 5.1: Рекурсивная вложенная сеть RN

чения аргумента n вычисляемой функции. Если $n = 0$, то срабатывает переход t_1 (что соответствует вычислению $fact(0) = 1$) и вычисление завершается.

Если $n \neq 0$, то срабатывает переход t_2, порождая элементную сеть F с начальной разметкой, что соответствует уменьшению n на 1 и внутреннему вызову процедуры вычисления факториала. После этого системная сеть F_0 содержит сетевую фишку (сеть F с начальной разметкой) в позиции p_3. Далее в этой сетевой фишке может сработать переход s_2, что приводит к порождению новой сетевой фишки и нового уровня в сети RN. Несколько таких шагов приведут к разметке, изображенной схематично на рисунке 5.2.

В некоторый момент в сети нижнего уровня может сработать переход s_1. Он должен сработать синхронно с переходом s_3, помеченным меткой $\bar{l}$ в сети предыдущего уровня. Это приводит к исчезновению сетевой фишки самого нижнего уровня и уменьшению числа уровней в NP-сети на 1. После этого может сработать

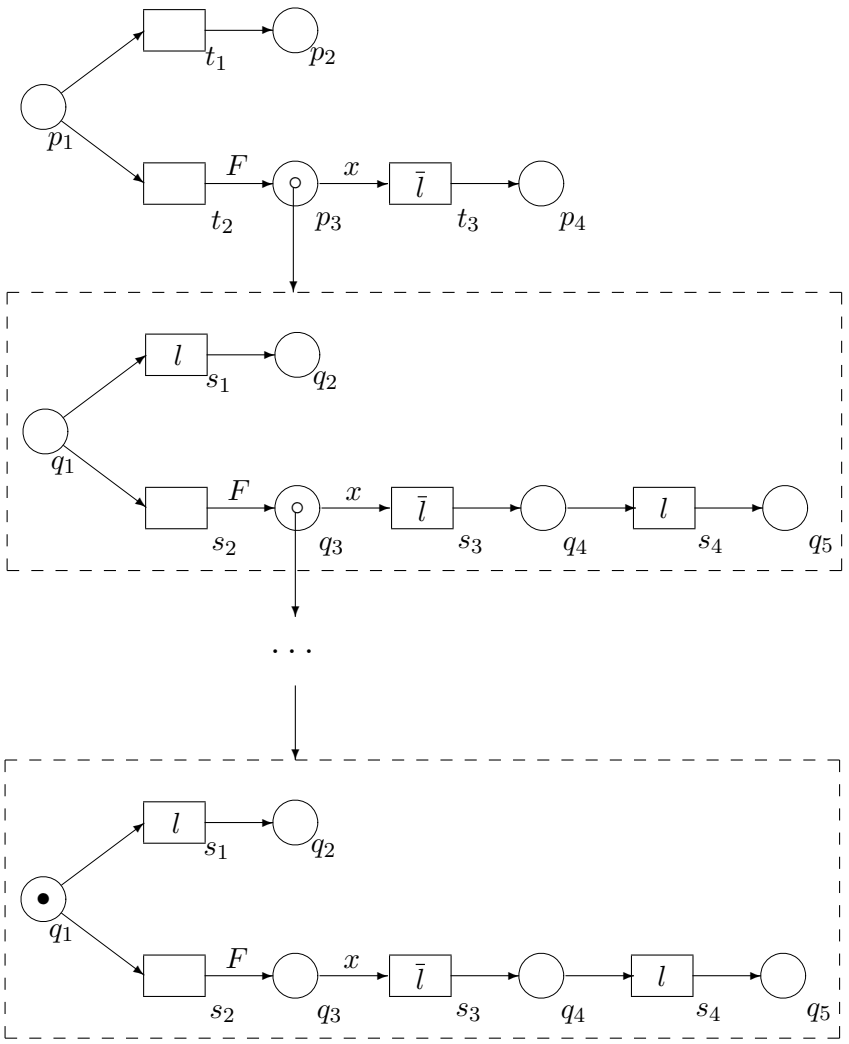


Рис. 5.2: Пример достижимой разметки для RN

только переход s_4 в сетевой фишке (текущего) самого нижнего уровня. Он срабатывает синхронно с переходом s_3 в "родительской" сети. Такой шаг уменьшит число уровней еще на 1. Этот процесс будет продолжаться до тех пор, пока в разметке не останется только один уровень — системная сеть с черной точкой в позиции p_3, после чего исполнение завершится.

Для NP-сети RN возможно также одно бесконечное исполнение, при котором процесс порождения нового уровня продолжается бесконечно. Однако после любого числа шагов разметка сети остается конечной. Сеть RN симулирует вычисление факториала в том смысле, что проверка на равенство нулю моделируется с помощью недетерминированного выбора (возможность проверки на ноль, как известно, расширила бы выразительность сетей Петри до универсальной вычислимости), и имеется взаимно-однозначное соответствие между множеством вычислений функции $fact(n)$ для $n = 0, 1, 2, \ldots$ и множеством конечных исполнений сети RN.

5.2 Вложенные сети с автономными и локализованными элементами

Благодаря вертикальной синхронизации срабатывание того или иного перехода в сетевой фишке может зависеть от разметки родительской по отношению к этой фишке сети. Один из способов сделать поведение сетевой фишки полностью автономным — разрешить только такие шаги вертикальной синхронизации, после выполнения которых задействованные в этом срабатывании фишки исчезают из родительской сети (*поглощающие шаги вертикальной синхронизации*). В этом случае от внешнего окружения зависит только порождение или уничтожение сетевой фишки. Это свойство выполняется, в частности, в том случае, когда выражения на выходных дугах перехода не содержат переменных, соответствующих задействованным в срабатывании этого перехода сетевым фишкам, что может быть проверено на синтаксическом

уровне. С другой стороны, нетрудно заметить, что по начальной разметке вложенной сети и выражениям на ее дугах можно определить, какого типа (сетевые или атомарные) фишки могут содержаться в той или иной позиции сети. Таким образом, проверку того, является ли переход, помеченный меткой для вертикальной синхронизации, поглощающим, можно выполнить на синтаксическом уровне.

Определение 5.1. Переход t NP-сети, помеченный меткой для вертикальной синхронизации, назовем *поглощающим*, если для любой входной для t позиции p выполняется следующее условие: либо для любого выражения $W(t,q)$, приписанного выходной дуге перехода t, имеет место $Var(W(p,t)) \cap Var(W(t,q)) = \emptyset$, либо p не содержит сетевых фишек во всех достижимых состояниях.

NP-сеть, в которой все переходы, помеченные метками для вертикальной синхронизации, являются поглощающими, будем называть *сетью с автономными элементами.*

NP-сеть RN, приведенная на рисунке 5.1, является сетью с автономными элементами. Действительно, у переходов, помеченных меткой l, предусловия могут содержать только атомарные фишки, а у перехода, помеченного $\bar{l}$, выражения на выходных дугах не содержат переменных.

Определим также более слабое ограничение на зависимость поведения элемента от разметок внешних по отношению к нему сетей. Пусть α — некоторая сетевая фишка в разметке M сети NPN. Через $\mathcal{T}^k(M,\alpha)$ обозначим максимальное поддерево в дереве разметки M такое, что высота поддерева $\mathcal{T}^k(M,\alpha)$ равна k и вершина α является листом в этом поддереве. Схематично поддерево $T^k(\alpha)$ представлено на рисунке 5.2.

Определение 5.2. Будем говорить, что сетевая фишка α в разметке M сети NPN является *локализованной*, если существует число $k \geq 0$ такое, что поведение этой сетевой фишки полностью

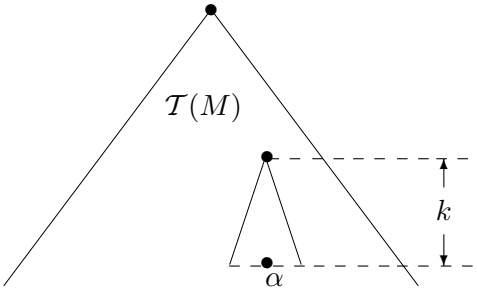


Рис. 5.3: Поддерево $\mathcal{T}^k(M,\alpha)$ в дереве разметки $\mathcal{T}(M)$

определяется разметкой поддерева $\mathcal{T}^k(M,\alpha)$. Число k будем называть *порядком локализации сетевой фишки* α.

NP-сеть, в которой все возникающие в ходе ее исполнения сетевые фишки являются локализованными с некоторым порядком локализации k, будем называть *сетью с локализованными элементами*. Число k будем называть в этом случае *порядком локализации сети*.

Итак, в локализованной сетевой фишке изменение ее внутренней разметки зависит от ограниченного числа верхних по отношению к ней уровней. Автономные фишки являются локализованными с порядком локализации ноль.

Отметим, что определение сетей с локализованными элементами, в отличие от определения сетей с автономными элементами, является семантическим, а не синтаксическим. Между тем во многих случаях локализованность элементов в сети легко устанавливается непосредственно по ее структуре. Например, если для любой сетевой фишки либо она сама, либо ее "родитель" является автономным элементом, то сеть является локализованной с порядком локализации 1. Заметим также, что в силу ограниченности глубины достижимых разметок все многоуровневые NP-сети являются сетями с локализованными элементами.

5.3 Проблемы останова и поддержки активности переходов для рекурсивных сетей

В предыдущей главе было показано, что из вполне структурированности многоуровневых NP-сетей следует разрешимость для них проблем поддержки управляющего состояния, неизбежности и останова.

В случае рекурсивных сетей глубина деревьев достижимых разметок может быть не ограничена. В этом случае изоморфное вложение деревьев $\preceq$ не может быть использовано в качестве подходящего упорядочения разметок сети, так как для деревьев неограниченной глубины оно, вообще говоря, не является правильным квазиупорядочением.

При использовании других упорядочений (например, гомеоморфного вложения деревьев в смысле Краскала [56]) удается получить правильное квазиупорядочение, но для шагов вертикальной синхронизации оказывается нарушенным условие совместимости. Это связано с тем, что шаг вертикальной синхронизации делает поведение отдельной фишки зависимым от разметки объемлющих эту фишку сетей, причем эту зависимость нельзя локализовать. Поэтому алгоритм решения, в частности, проблемы останова для многоуровневых сетей не удается обобщить на общий случай рекурсивных сетей.

Проблема останова состоит в проверке того, что всякое исполнение вложенной сети приводит к некоторой тупиковой разметке (при которой ни один переход не может сработать).

Проблема поддержки активности переходов для вложенной сети NPN состоит в том, чтобы для данного конечного набора переходов $t^1, \ldots, t^k$ установить, существует ли такое исполнение $M_0[\,\rangle M_1[\,\rangle \ldots$ для NPN, что при любой из встречающихся в этом исполнении разметке M_i хотя бы один из переходов $t^1, \ldots, t^k$ может сработать.

Проблема поддержки активности переходов может рассматриваться как специальный случай *проблемы поддержки управляющего состояния*, которая состоит в том, чтобы для данного конечного набора разметок $M^1, \ldots, M^k$ установить, существует ли такое исполнение $M_0[\,\rangle M_1[\,\rangle \ldots$ для этой сети, что всякая встречающаяся в этом исполнении разметка M_i удовлетворяют условию $(M_i \geq M^1) \vee \cdots \vee (M_i \geq M^k)$ для некоторого частичного порядка $\geq$ на множестве разметок сети. Содержательно это означает, что возможно такое исполнение, при котором "минимальный запас ресурсов", определяемый набором взаимозаменяемых ресурсов, заданных разметками $M^1, \ldots, M^k$, будет поддерживаться в течение всего исполнения. В частности, при наличии условия монотонности для обыкновенных сетей Петри (или условия совместимости для других моделей) распознавание поддержки активности переходов является проверкой обеспечения запаса ресурсов, достаточного для активности одного из заданных переходов.

Для многоуровневых NP-сетей выполняются условия совместимости для отношения $\preceq$ на множестве разметок, и поэтому для них проблема поддержки управляющего состояния разрешима. Для рекурсивных сетей Петри, в силу нарушения условия совместимости для отношения $\preceq$, проверка поддержки управляющего состояния относительно $\preceq$ не является обобщением проблемы поддержки активности переходов.

Далее мы покажем, что проблемы останова и поддержки активности переходов разрешимы для рекурсивных NP-сетей с автономными и локализованными элементами. Для этого будет использоваться специальная конструкция, аналогичная покрывающему дереву сети.

Прежде всего заметим, что глубина разметки в рекурсивной сети может увеличиваться только при срабатывании перехода, одна из выходных дуг которого содержит сетевую константу в приписанном ей выражении. При построении покрывающего дерева для рекурсивной сети мы будем дополнительно отслеживать те

случаи, когда порожденная в некоторый момент сетевая фишка порождает внутри себя такую же сетевую фишку с той же начальной разметкой.

Пусть RN — рекурсивная NP-сеть, M — некоторая ее разметка. Через $Succ(M) = \{M' \mid M[\,\rangle M'\}$ обозначим множество всех последующих разметок для M.

Итеративное покрывающее дерево (с порядком локализации k) для NP-сети RN с начальной разметкой M_0 представляет собой конечный ориентированный граф (дерево) с вершинами, помеченными разметками сети RN, и строится следующим образом:

- корню дерева приписывается пометка M_0, и эта вершина объявляется внутренней вершиной дерева;
- внутренняя вершина с пометкой M имеет по одному потомку, помеченному M', для каждого состояния $M' \in Succ(M)$;
- если вершина не имеет потомков, то она объявляется *финальной* вершиной;
- если на пути от корня до некоторой вершины с пометкой M' встречается вершина с пометкой M такой, что $M \preceq M'$, то говорят, что M' *покрывает* M, и вершина M' объявляется *покрывающей*;
- если на пути от корня до некоторой вершины с пометкой M' встречается вершина с пометкой M такой, что

  1) обе разметки M и M' получены при срабатывании некоторого перехода, порождающего сетевые фишки α и β с одинаковыми сетевыми структурами и разметками;
  2) последняя фишка β оказывается (непосредственно или опосредованно) вложенной в первую фишку α;
  3) $\mathcal{T}^k(M, \alpha) \preceq \mathcal{T}^k(M, \beta)$,

  то вершина M' объявляется *итеративной*;

- финальные, покрывающие и итеративные вершины не имеют потомков.

Таким образом, листья в итеративном покрывающем дереве NP-сети являются финальными, покрывающими или итеративными. Легко видеть, что для многоуровневой NP-сети итеративное покрывающее дерево совпадает с обычным покрывающим деревом.

Лемма 5.1. *Для любой рекурсивной NP-сети RN итеративное покрывающее дерево при заданном порядке локализации конечно и может быть эффективно построено.*

Доказательство. Прежде всего напомним, что в выражениях на выходных дугах любого перехода встречаются только переменные, входящие в выражения на входных дугах этого перехода. Это ограничение обеспечивает конечность ветвлений итеративного покрывающего дерева рекурсивной NP-сети.

Покажем, что все пути в покрывающем дереве конечны. Если глубина разметок вдоль некоторого пути в покрывающем дереве ограничена, то в силу правильности упорядочения $\preceq$ этот путь должен содержать покрывающую вершину, и, следовательно, он конечен. Если вдоль некоторого пути встречаются разметки сколь угодно большой глубины, то на этом пути обязательно встретится итеративная вершина.

Действительно, число сетевых констант, встречающихся в сети RN, конечно. Следовательно, вдоль пути, в котором бесконечное число раз внутри некоторой сетевой фишки порождается новая сетевая фишка, обязательно будет бесконечное число раз порождаться одна и та же сетевая фишка α (с одной и той же разметкой). Рассмотрим бесконечную последовательность разметок $M_1, M_2, \ldots$, получаемых в результате очередного порождения внутренней сетевой фишки α, и построим последовательность $\mathcal{T}^k(M_1, \alpha), \mathcal{T}^k(M_2, \alpha), \ldots$ поддеревьев высоты k. В силу ограниченности высоты этих поддеревьев по теореме 1.2 о правильности

изоморфного вложения деревьев найдутся такие индексы $i \leq j$, что $\mathcal{T}^k(M_i, \alpha) \preceq \mathcal{T}^k(M_j, \alpha)$.

Итак, все пути в итеративном покрывающем дереве сети конечны. Тогда из леммы Кенига следует конечность итеративного покрывающего дерева при заданном уровне локализации.

Далее, поскольку отношение порядка $\preceq$ разрешимо (т. е. существует эффективная процедура проверки истинности $M \preceq M'$ для любой пары разметок M, M' и проверки $\mathcal{T} \preceq \mathcal{T}'$ для двух деревьев разметок ограниченной высоты) и, кроме того, отображение *Succ* вычислимо (т. е. существует эффективная процедура вычисления множества $Succ(M)$ для любой разметки M), то итеративное покрывающее дерево для рекурсивной сети RN может быть эффективно построено. □

Для построения покрывающего дерева не требуется совместимость между отношениями упорядоченности $\preceq$ и перехода $[\,\rangle$. Однако именно при выполнении условия совместимости покрывающее дерево содержит полезную информацию о свойствах поведения системы.

Теорема 5.1. *Проблемы останова и поддержки активности переходов разрешимы для рекурсивных NP-сетей с локализованными элементами.*

Доказательство. Для решения этих проблем строится конечное итеративное покрывающее дерево для рекурсивной NP-сети.

В покрывающем дереве рекурсивной NP-сети RN путь, оканчивающийся финальной разметкой, соответствует конечному исполнению для RN.

Если некоторый путь заканчивается покрывающей вершиной с пометкой M' такой, что $M \preceq M'$, то в силу теоремы о совместимости любая последовательность срабатываний, возможная при разметке M, возможна также при разметке M'. Тогда последовательность срабатываний, переводящая разметку M в разметку M', может повторяться бесконечное число раз.

Пусть некоторый путь заканчивается итеративной вершиной с пометкой M' и на этом пути встречается вершина с пометкой M так, что обе разметки M и M' получены при срабатывании некоторого перехода, порождающего одну и ту же сетевую фишку (одну и ту же маркированную сеть $(\mathfrak{N}, m)$), и при этом фишка α', порожденная последней, оказывается (непосредственно или опосредованно) вложенной в первую фишку α. При этом выполняется также $\mathcal{T}^k(M, \alpha) \preceq \mathcal{T}^k(M', \alpha')$.

Пусть теперь $t_1, t_2, \ldots$ — последовательность срабатываний переходов, возможная при разметке M, и пусть $t_{i_1}, t_{i_2}, \ldots$ — ее подпоследовательность, все переходы которой срабатывают внутри сетевой фишки α. Тогда в силу локализованности элементов подпоследовательность $t_{i_1}, t_{i_2}, \ldots$ срабатываний возможна как при разметке M, так и при разметке M'. В частности, последовательность срабатываний переходов внутри сетевой фишки α, переводящая разметку M в разметку M', может повторяться неограниченное число раз.

Таким образом, для проверки того, что всякое вычисление в системе переходов завершается, достаточно перебрать все листья в конечном итеративном покрывающем дереве сети и убедиться, что все они помечены финальными состояниями.

Исполнение, в котором при всякой встречающейся разметке, возможно срабатывание одного из переходов $t_1, t_2, \ldots, t_k$, возможно для рекурсивной NP-сети RN тогда и только тогда, когда итеративное покрывающее дерево рекурсивной сети RN содержит путь от корня к листу, в котором все вершины имеют пометки, удовлетворяющие этому (эффективно проверяемому) условию.

Действительно, если на каждом пути от корня к листу в покрывающем дереве встречается разметка, не удовлетворяющая требуемому условию, то, поскольку любое исполнение имеет в качестве префикса некоторый путь в дереве покрытия, разметка, нарушающая требуемое условие, встретится в любом исполнении.

С другой стороны, если покрывающее дерево содержит путь, в

котором все разметки удовлетворяют заданному условию, то либо этот путь заканчивается финальной разметкой, либо может быть продолжен так, что все следующие разметки также удовлетворяют требуемому условию. В самом деле, для пути в покрывающем дереве, заканчивающегося покрывающей вершиной, это обеспечивается условием совместимости.

Для пути, заканчивающегося итеративной вершиной, возможно его продолжение, повторяющее бесконечное число раз уже "пройденную" часть пути. Таким образом, если некоторый переход был активным во всех разметках на пути до итеративной вершины, то этим свойством будет обладать и его бесконечное повторение. □

Глава 6

Сравнение вложенных сетей с другими формальными моделями

6.1 Языковая выразительность вложенных сетей Петри

В этом разделе исследуются языковые свойства вложенных рекурсивных сетей Петри. Пример 5.1 в главе 4 показывает, что рекурсивные NP-сети удобны для моделирования систем, имеющих рекурсивную природу. Этим свойством не обладают обыкновенные сети Петри. Так, известно [49], что такой фундаментальный класс формальных систем, как класс контекстно-свободных языков, не сравним с классом языков, порождаемых обыкновенными сетями Петри. Ниже будет изложен довольно простой способ моделирования рекурсивными вложенными сетями произвольного КС-языка из [4].

Во вложенных сетях метки переходов из Lab используются для синхронизации переходов в системной и в элементных сетях, за

счет чего достигается взаимодействие различных слоев рекурсивной сети. При определении языка, порождаемого сетью, используется еще один вид меток — имена переходов, как это принято в традиционных помеченных сетях Петри.

Считается, что каждый шаг помеченной сети порождает символ из множества имен переходов A, которым помечен сработавший переход, или пустой символ, если данный переход не помечен. При срабатывании шага синхронизации все составляющие его переходы помечены одним и тем же символом из A (или все не помечены). Таким образом, любое исполнение сети порождает некоторое слово из A^*.

Определение 6.1. Пусть $A = \{a_1, a_2, \ldots, a_n\}$ — конечный алфавит. NP-сеть *NPN* назовем *помеченной*, если задана частичная функция $f : T \to A$, где $T = T_{\mathfrak{N}_1} \cup T_{\mathfrak{N}_2} \cup \cdots \cup T_{\mathfrak{N}_n}$ — множество всех переходов в сетевых компонентах $\mathfrak{N}_1, \ldots, \mathfrak{N}_k$ сети *NPN*. При этом предполагается, что переходы, помеченные взаимно дополнительными метками из *Lab*, либо помечены одним и тем же символом из A, либо все непомечены.

Будем говорить, что слово $\alpha = (a_1 a_2 \ldots a_k) \in A^*$ порождается NP-сетью *NPN* в качестве терминального, если существует последовательность $M_0 \xrightarrow{t_1} M_1 \xrightarrow{t_2} \cdots \xrightarrow{t_n} M_n$ срабатываний переходов в *NPN*, где M_0 — начальная разметка, M_n — тупиковая разметка и $\alpha = (f(t_0) f(t_1) \ldots f(t_n))$.

Множество $L(NPN)$ терминальных слов, порождаемых NP-сетью *NPN*, будем называть терминальным языком, порождаемым сетью *NPN*.

Напомним определение контекстно-свободных языков.

Определение 6.2. *Порождающей формальной грамматикой* называется четверка $G = (V, W, S, R)$, где V — алфавит *терминальных* (основных) символов; W — алфавит *нетерминальных* (вспомогательных) символов; $V \cap W = \emptyset$; R — конечное множество *правил вывода* вида $\xi \to \eta$, где ξ и η — цепочки (слова) в алфавите

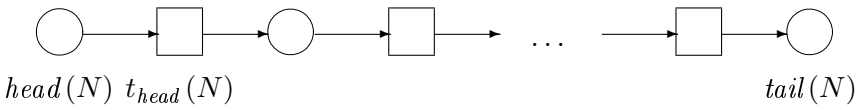


Рис. 6.1: Сеть-цепочка

$V \cup W$; S — *начальный символ* (аксиома грамматики). Применение правила $R = (\xi \to \eta)$ к слову ψ в алфавите $V \cup W$ состоит в замене некоторого вхождения подслова ξ в ψ на подслово η.

Языком $L(G)$, *порождаемым грамматикой* G, называется множество всех цепочек в терминальном алфавите V, выводимых из S последовательным применением правил из R.

Грамматика называется *контекстно-свободной (КС-грамматикой)*, если все ее правила вывода имеют вид $A \to \alpha$, где $A \in W$, $\alpha \in (V \cup W)^*$.

Определим несколько преобразований над простейшими рекурсивными сетями. *Цепочкой* назовем сеть $N = (P, T, F)$, имеющую вид, представленный на Рис. 6.1. В сети N выделяются *головная позиция* $head(N)$, *хвостовая позиция* $tail(N)$ и один *головной переход* $t_{head}(N)$.

Определим следующие две операции над цепочками:

- *Конкатенация.* Пусть $N_1 = (P_1, T_1, F_1)$ и $N_2 = (P_2, T_2, F_2)$ — цепочки. Тогда $N_1.N_2 =_{\text{def}} (P_1 \cup P_2 \setminus \{head(N_2)\}, T_1 \cup T_2, F_1 \cup (F_2 \setminus \{(head(N_2), t_{head}(N_2))\}) \cup \{(tail(N_1), t_{head}(N_2))\})$.

- *Склеивание головных позиций.* Пусть $N_1 = (P_1, T_1, F_1), \ldots, N_k = (P_k, T_k, F_k)$ — цепочки. Тогда
  $Link_{i=1}^{k}(N_i) =_{\text{def}} (\cup_{i=1}^{k}(P_i \setminus \{head(N_i)\}) \cup \{p\},$
  $\cup_{i=1}^{k} T_i, \cup_{i=1}^{k}(F_i \setminus \{(head(N_i), t_{head}(N_i))\} \cup \{(p, t_{head}(N_i))\}))$.

Результат применения данных операций к цепочкам показан на рисунке 6.2. Очевидно, что в результате конкатенации опять

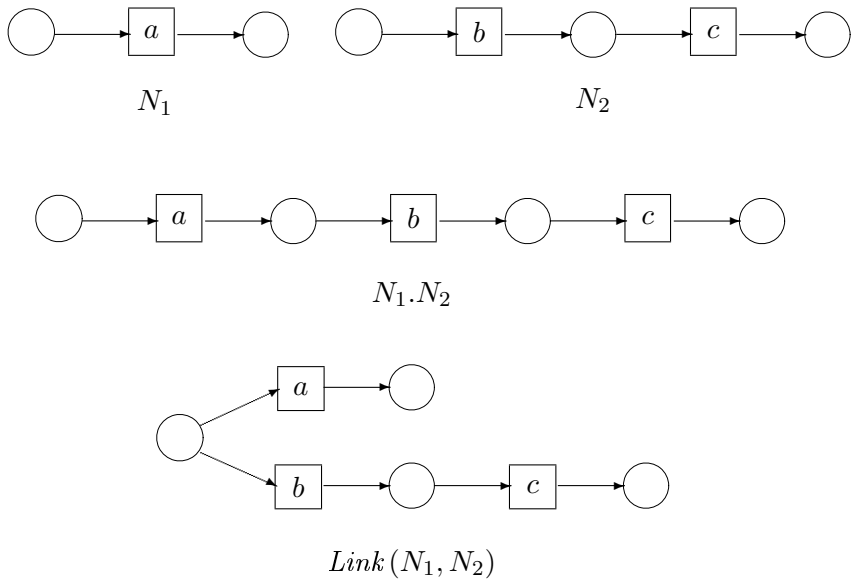


Рис. 6.2: Операции над сетями

получается цепочка, тогда как в результате склеивания n цепочек получается сеть более сложной структуры, у которой одна головная и n хвостовых позиций.

Введем также три шаблона элементарных рекурсивных сетей, которые будут играть роль атомарных структур, из которых будет строиться итоговая сеть (Рис. 6.3). На рисунке метки из Lab проставлены внутри переходов, метки из A — над переходами. Эти шаблоны имеют структуру цепочки, поэтому к ним применимы операции конкатенации и склеивания.

Теорема 6.1. *Для любой КС-грамматики G существует помеченная рекурсивная NP-сеть с автономными элементами, которая порождает $L(G)$ в качестве терминального языка.*

Доказательство. В качестве доказательства приведем алгоритм

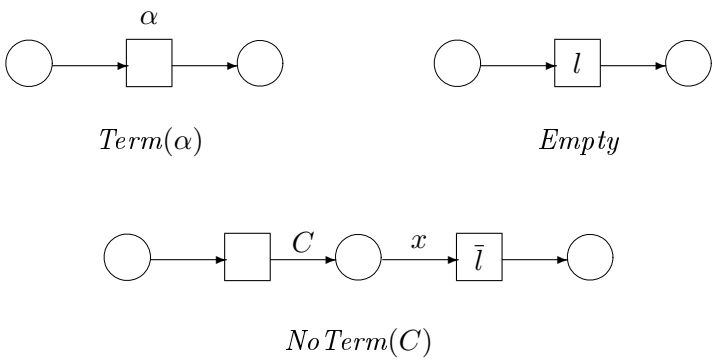


Рис. 6.3: Шаблоны

построения рекурсивной NP-сети с автономными элементами по данной произвольной КС-грамматике $G = (V, W, S, R)$:

1) Для каждого правила вывода $r_i \in R$ построим соответствующую ему сеть-цепочку $N(r_i)$:
шаг 1. Пусть $r_i : T \to \xi$. Полагаем $\eta := \xi$. $N(r_i) := (\{p\}, \emptyset, \emptyset)$.
шаг 2. Если длина η не равна 0, то возможны два случая:

   - $\eta = T'\eta'$, где $T' \in W$, $\eta' \in (V \cup W)^*$. Тогда $N(r_i) := Term(N_{T'}).N(r_i)$, $\eta := \eta'$, и возвращаемся на шаг 2.
   - $\eta = a\eta'$, где $a \in V$, $\eta' \in (V \cup W)^*$. Тогда $N(r_i) := NoTerm(a).N(r_i)$, $\eta := \eta'$, и возвращаемся на шаг 2.

   шаг 3. $N(r_i) := N(r_i).Empty$.

2) Обозначим $I_T = \{i \mid (r_i \in R) \,\&\, (r_i = T \to \xi)\}$. Построим для каждого $T \in W$ сеть $\mathfrak{N}_T = Link_{i \in I_T}(N(r_i))$.

3) Искомая помеченная NP-сеть состоит из системной сети $\mathfrak{N}_S$ и множества элементных сетей $\{\mathfrak{N}_T\}_{T \in W}$. Каждой константе на дугах сопоставлена сеть с соответствующим именем и

одной черной фишкой в головной позиции в качестве разметки. Начальная разметка состоит из одной черной фишки в головной позиции сети $\mathfrak{N}_S$. Имеется одна метка l для вертикальной синхронизации. Множество A пометок сети совпадает с множеством V терминальных символов грамматики.

Для конечных грамматик данный алгоритм конечен. Действительно, в силу конечности R число шагов 1 и 2 конечно. Далее, при каждом выполнении шага 2 происходит уменьшение рассматриваемой строки, а длина строк в R также конечна.

Полученная сеть в любой момент своего функционирования содержит конечное число слоев (уровней вложенности) и ровно по одной сети в каждом слое. Недетерминированность возможна только в самом внутреннем слое, если соответствующий объект получен склеиванием, и в нем не сработал еще ни один переход. Это естественным образом соответствует случаю недетерминированного выбора из нескольких правил вывода в случае грамматик.

Вообще, сработать всегда может только переход во внутреннем слое (возможно, синхронно с переходом в соседнем слое, причем при этом самый внутренний слой просто исчезнет), поскольку, в соответствие со структурой сети $NoTerm$, содержащая объект сеть всегда дожидается его исчезновения, прежде чем сможет сработать сама. Этим обеспечивается автономность элементов сети. Правильная вложенность объектов друг в друга (в соответствие с правилами вывода) позволяет в этом случае гарантировать правильность порождения языка $L(G)$. □

6.2 Вложенные сети Петри и алгебры процессов

Среди большого числа формализмов для спецификации и верификации распределенных параллельных систем важную роль играют алгебры процессов. В алгебрах процессов поведение системы описывается с помощью алгебраического выражения над атомарными действиями. Выразительность того или иного класса алгебр процессов зависит от допускаемого набора операций. Мы будем сравнивать вложенные рекурсивные сети с BPA-алгебрами [21]. Результаты, изложенные в этом разделе, приведены в [7].

Выбор этого класса алгебр обусловлен тем, что класс языков, порождаемых BPA-алгебрами, и класс языков, порождаемых обыкновенными сетями Петри, пересекаются, но не вкладываются один в другой [28]. Также известно, что класс языков, порождаемых BPA-алгебрами, совпадает с классом контекстно-свободных языков, а класс контекстно-свободных языков, как было показано выше, вкладывается в класс языков, порождаемых вложенными рекурсивными сетями Петри. Поэтому можно ожидать, что рекурсивные вложенные сети моделируют процессы, задаваемые BPA-алгебрами. Ниже сравнение вложенных сетей Петри и алгебр процессов будет проводиться с точки зрения бисимуляционной эквивалентности [72, 79], учитывающей причинно-следственные связи более тонко, чем языковая эквивалентность, и позволяющей обнаруживать, в частности, дедлоки.

Класс BPA-программ строится из атомарных действий с помощью операций последовательного выполнения, недетерминированного выбора и рекурсии.

Определение 6.3. Пусть $Act = \{0, a_1, a_2, \dots\}$ — множество *атомарных действий*, $\mathbb{X} = \{X_1, X_2, \dots\}$ — множество переменных. Через E будем обозначать выражение в алгебре процессов BPA, через P — BPA-программу. *BPA-программа* есть набор равенств вида: $\{X_i = E_i\}_{0 \le i \le n}$, где E_i есть выражение, определяемое по

следующим правилам:

$$E ::= 0 \mid a \mid X_i \mid E_1E_2 \mid E_1 + E_2.$$

Равенство $X_i = E_i$ будем называть *компонентой* BPA-программы. Среди компонент выделена главная компонента. Если это не указано явно, то *главной* будем считать компоненту $X_1 = E_1$.

Поведение BPA-программы можно описать, сопоставляя ей систему помеченных переходов. Состояниями будут процессы, переходами — действия.

Семантику BPA-программ определим в стиле Плоткина [82], т. е. посредством системы правил. Запись $E \xrightarrow{a} E'$ означает, что действие a переводит состояние E в состояние E'.

$$(.)\frac{E_1 \xrightarrow{a} E_1'}{E_1.E_2 \xrightarrow{a} E_1'.E_2}; \frac{isnil(E_1)E_2 \xrightarrow{a} E_2'}{E_1.E_2 \xrightarrow{a} E_2'}$$

$$(+)\frac{E_1 \xrightarrow{a} E_1'}{E_1 + E_2 \xrightarrow{a} E_1'}; \frac{E_2 \xrightarrow{a} E_2'}{E_1 + E_2 \xrightarrow{a} E_2'}$$

$$\frac{E \xrightarrow{a} E'}{X \xrightarrow{a} E'}, \text{г}X =_{\text{def}} E.$$

Здесь функция *isnil* определяется следующим образом:

$isnil(0) = \underline{\text{tt}}$; $isnil(aE) = \underline{\text{ff}}$;

$isnil(E_1 + E_2) = isnil(E_1E_2) = isnil(E_1)\&isnil(E_2)$;

$isnil(X) = isnil(E)$, где $X =_{\text{def}} E$.

Таким образом, каждая BPA-программа порождает набор последовательностей действий, а *состояния* системы задаются выражениями, которые определяют дальнейшее поведение системы. Состояние E BPA-программы P называется *терминальным*, если в E не может выполниться ни одно действие.

Определение 6.4. Будем говорить, что слово $\alpha = (a_1 a_2 \dots a_n) \in Act^*$ *порождается BPA-программой* в качестве терминального, если существует последовательность действий $E_1 \xrightarrow{a_1} E_2 \xrightarrow{a_2} \dots \xrightarrow{a_{n-1}} E_n$, где E_1 — начальное, E_n — конечное состояния.

Множество $L(P)$ терминальных слов, порождаемых программой P, будем называть *терминальным языком* для P.

Как и в разделе 6.1, для определения терминального языка, порождаемого вложенной сетью Петри, имена переходов помечаются отличными от меток синхронизации метками.

При определении поведения BPA-программы и NP-сети каждой из них сопоставлялась система помеченных переходов, т. е. четверка $LTS = (S, A, \rightarrow, s_0)$, где S — множество состояний, A — алфавит имен переходов, $\rightarrow \subseteq S \times A \times S$ — отношение переходов, $s_0 \in S$ — начальное состояние. При этом в NP-сетях допускаются непомеченные (невидимые) переходы. Будем считать, что такие переходы помечены специальным символом τ.

На множестве систем помеченных переходов можно определить отношение эквивалентности так, что система LTS_1 эквивалентна системе LTS_2 тогда и только тогда, когда порождаемые ими языки равны. Более тонкой по сравнению с языковой эквивалентностью является бисимуляционная эквивалентность, введенная Р. Милнером и Д. Парком [72, 79].

Определение 6.5. Пусть заданы системы $LTS_1 = (S, A, \rightarrow_1, s_0)$, $LTS_2 = (T, A, \rightarrow_2, t_0)$ помеченных переходов с одним и тем же множеством A имен переходов. Говорят, что системы LTS_1 и LTS_2 *бисимуляционно эквивалентны* (обозначается: $LTS1 \approx LTS_2$), если найдется такое бинарное отношение $\mathcal{R} \subseteq S \times T$ (называемое отношением бисимуляции), что:

1) $(s_0, t_0) \in \mathcal{R}$;

2) $\forall s, s' \in S, \forall t \in T$: если $s \xrightarrow{\tau^* a \tau^*}_1 s'$ и $(s, t) \in \mathcal{R}$, то $\exists t' \in T : t \xrightarrow{\tau^* a \tau^*}_2 t'$ и $(s', t') \in \mathcal{R}$.

3) $\forall s \in S, \forall t, t' \in T$: если $t \stackrel{\tau^* a \tau^*}{\longrightarrow}_2 t'$ и $(t, s) \in \mathcal{R}$,
то $\exists s' \in S : s \stackrel{\tau^* a \tau^*}{\longrightarrow}_1 s'$ и $(t', s') \in \mathcal{R}$.

Отношение "$\approx$" есть наблюдаемая бисимуляция, которая не учитывает невидимых действий τ. Имеет место следующая

Теорема 6.2. *Для любой BPA-программы $P = \{X_i = E_i\}_{0 \le i \le n}$ можно эффективно построить бисимуляционно эквивалентную ей рекурсивную NP-сеть $NPN = \{\mathfrak{N}_1, \mathfrak{N}_2, \ldots, \mathfrak{N}_n\}$ с автономными элементами.*

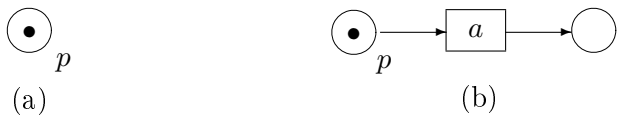
Доказательство. В качестве доказательства приведем
Алгоритм построения по BPA-программе P бисимуляционно эквивалентной ей NP-сети $NPN(P)$:

Фиксируем n имен сетевых компонент $\mathfrak{N}_1, \mathfrak{N}_2, \ldots, \mathfrak{N}_n$, которые будут соответствовать компонентам $X_1, X_2, \ldots, X_n$ процесса P. Для каждой компоненты $X_i = E_i$ построим маркированную сетевую компоненту $\mathfrak{N}(E_i)$ с именем $\mathfrak{N}_i$. При этом в каждой компоненте будем выделять одну начальную позицию и некоторое непустое подмножество заключительных переходов. Построение проведем индукцией по структуре программы.

Базис индукции.

1) Пусть $E = 0$, тогда сетевая компонента $\mathfrak{N}(E)$ состоит из одной начальной позиции p (см. рисунок 6.4 (a)).

2) Пусть $E = a$, где $a \in A$. Соответствующая сетевая компонента изображена на рисунке 6.4 (b).



(a) (b)

Рис. 6.4: (a) Компонента $\mathfrak{N}(0)$; (b) Компонента $\mathfrak{N}(a)$

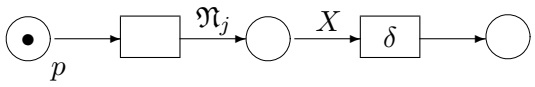


Рис. 6.5: Сетевая компонента $\mathfrak{N}(X_j)$

3) Пусть $E = X_j$, тогда сетевая компонента $\mathfrak{N}(E)$ (см. рисунок 6.5) ссылается на компоненту $\mathfrak{N}_j$, которая получается из $\mathfrak{N}(E_j)$ добавлением дополнительной к δ пометки $\overline{\delta}$ для вертикальной синхронизации на все заключительные переходы сети $\mathfrak{N}(E_j)$.

Теперь в качестве *индукционного предположения* полагаем, что для компонент E_1 и E_2 построены соответствующие сетевые компоненты $\mathfrak{N}_1 = \mathfrak{N}(E_1) = (P_1, T_1, F_1)$ и $\mathfrak{N}_2 = \mathfrak{N}(E_2) = (P_2, T_2, F_2)$ и в них выделены входные позиции p_1 и p_2 соответственно.

Индукционный шаг выполняется разбором случаев.

1) Построим $\mathfrak{N}(E)$ для $E = E_1E_2$. Если $E_1 = 0$, то сетевая компонента $\mathfrak{N}(E)$ будет совпадать с сетевой компонентой $\mathfrak{N}(0)$. В противном случае действия компоненты E_2 могут начать выполнятся только после того, как выполнено какое-либо

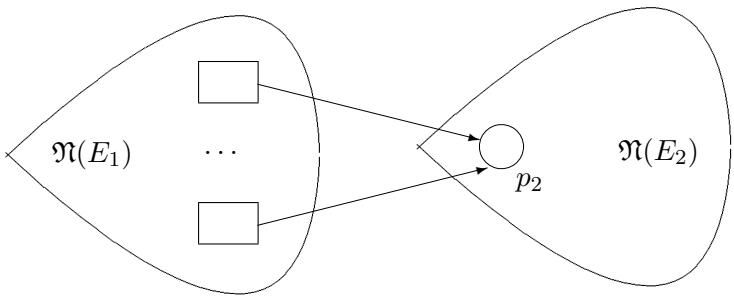


Рис. 6.6: Сетевая компонента $\mathfrak{N}(E_1E_2)$

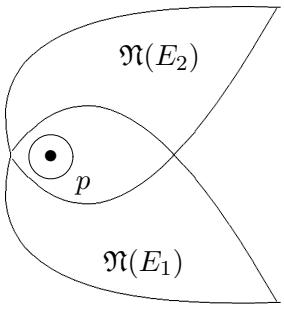


Рис. 6.7: Сетевая компонента $\mathfrak{N}(E_1 + E_2)$

заключительное действие компоненты E_1, и, следовательно, переходы сетевой компоненты $\mathfrak{N}_2$, соответствующие заключительным действиям в E_2, сработают только после срабатывания какого-то заключительного перехода сетевой компоненты $\mathfrak{N}_1$, соответствующего заключительному действию в E_1 (см. рисунок 6.6).

2) Сетевая компонента $\mathfrak{N}(E)$ для $E = E_1 + E_2$ изображена на рисунке 6.7. Для ее построения нужно слить начальные позиции компонент $\mathfrak{N}(E_1)$ и $\mathfrak{N}(E_2)$.

NP-сеть $NPN(P)$, бисимуляционно эквивалентная BPA-программе $P = \{X_i = E_i\}$, состоит из сетевых компонент $\mathfrak{N}(E_1)$, $\mathfrak{N}(E_2), \ldots, \mathfrak{N}(E_n)$, где $\mathfrak{N}(E_1)$ — системная сеть, $\mathfrak{N}(E_2), \ldots, \mathfrak{N}(E_n)$ — элементные сети. Размер получаемой сети пропорционален размеру BPA-программы, а приведенный алгоритм имеет линейную временную сложность. □

Следствие 6.1. *Класс языков, порождаемых BPA-программами, вкладывается в класс языков, порождаемый рекурсивными NP-сетями.*

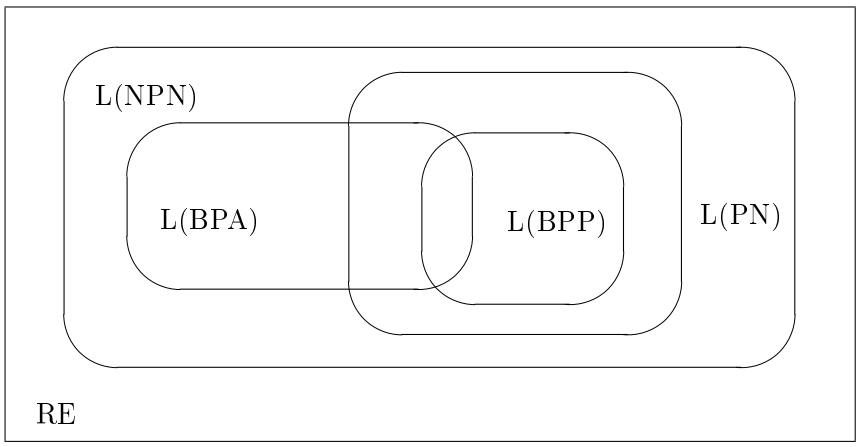


Рис. 6.8: Языки сетей Петри и алгебр процессов

Полученные результаты дополняют схему соотношений между языками алгебр процессов и сетей Петри, приведенную в [28]. Место, которое занимают в этой схеме вложенные рекурсивные сети Петри, показано на рисунке 6.8. Здесь RE — класс рекурсивно перечислимых языков, L(PN) — языки, порождаемые обыкновенными сетями Петри, L(BPP) — языки, порождаемые BPP-программами, L(BPA) — языки, порождаемые BPA-программами, L(NPN) — языки, порождаемые NP-сетями. При этом имеет место не только вложение $L(BPA) \subseteq L(NPN)$, но и возможность симуляции BPA-программ NP-сетями с сохранением бисимуляционной эквивалентности.

6.3 Вложенные сети Петри и другие классы сетей Петри

В этом параграфе мы сравним выразительные возможности NP-сетей с некоторыми другими расширениями сетей Петри.

Прежде всего заметим, что поскольку все фишки в системной сети могут быть атомарными черными фишками, имеет место

Утверждение 6.1. *Обыкновенные сети Петри являются частным случаем NP-сетей.*

Покажем, что NP-сети более выразительны, чем обыкновенные сети Петри. Действительно, в параграфе 3.5.3 было доказано, что проблемы достижимости, живости и ограниченности неразрешимы для двухуровневых NP-сетей. Однако для обыкновенных сетей Петри эти проблемы разрешимы [8, 55, 70]. Отсюда немедленно получаем

Теорема 6.3. *Двухуровневые NP-сети с обыкновенными сетями Петри в качестве элементных сетей более выразительны, чем обыкновенные сети Петри.*

Ингибиторные сети Петри [17] представляют собой расширение модели обыкновенных сетей Петри за счет добавления специальных *ингибиторных дуг* для проверки того, что заданная позиция фишек не содержит. Ингибиторная дуга всегда направлена от позиции к переходу. Срабатывание соответствующего перехода возможно только в том случае, когда позиция, являющаяся началом инцидентной переходу дуги, пуста, т. е. не содержит фишек.

Известно [52, 54], что сети Петри не могут моделировать машины Минского и Тьюринга. Добавление же ингибиторных дуг делает сети Петри "универсально выразительными".

Теорема 6.4 (Murata). *Класс ингибиторных сетей равен по выразительности классу машин Тьюринга.*

При определении NP-сетей были наложены ограничения на выражения, приписываемые входным дугам переходов сети. Выше (см. замечание 3.2) было показано, что без этих ограничений NP-сети не являлись бы вполне структурированными системами переходов относительно частичного порядка $\preceq$ на множестве

разметок сети. Теперь будет показано, что без этих ограничений на выражения, приписываемые входным дугам, NP-сети были бы равны по выразительности машинам Тьюринга.

Утверждение 6.2. *Ингибиторные сети Петри моделируются NP-сетями при условии отмены в определении NP-сетей ограничений на выражения, приписываемые входным дугам.*

Доказательство. Моделирование ингибиторной дуги NP-сетью, в которой две входные для одного и того же перехода t дуги помечены одной и той же переменной x, показано на рисунке 6.3. Здесь n фишек в позиции p моделируются элементной сетью, содержащей n черных фишек (аналогично тому, как это делалось при моделировании обнуляющей дуги). Ингибиторная дуга, проверяющая пустоту позиции p, моделируется с помощью дополнительной позиции p_0 (общей для всех ингибиторных дуг) и дуги с пометкой x от этой позиции к переходу t. Проверка пустоты исходной позиции p производится путем сравнения сетевой фишки в позиции p с сетевой фишкой в позиции p_0. □

Далее, из разрешимости для NP-сетей с локализованными элементами проблемы останова следует

Теорема 6.5. *Рекурсивные NP-сети с локализованными элементами слабее по выразительности, чем машины Тьюринга.*

Заметим также, что в предыдущем разделе было показано, что рекурсивные NP-сети с автономными элементами могут порождать в качестве терминального любой контекстно-свободный язык, в то время как для обыкновенных сетей Петри это не верно. Отсюда следует, что рекурсивные NP-сети с автономными элементами сильнее по выразительности, чем обыкновенные сети Петри.

Сравним теперь рекурсивные NP-сети с рекурсивными сетями Хаддада и Пватрено [45]. Рекурсивные сети Хаддада и Пватрено имеют такую же структуру, как и обыкновенные сети Петри, с

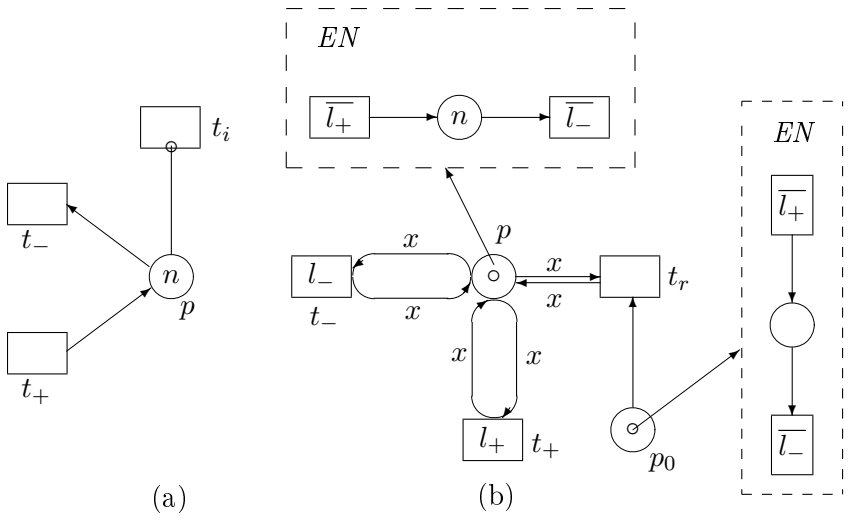


Рис. 6.9: Моделирование ингибиторной дуги

той только разницей, что переходы разбиты на три вида: элементарные, абстрактные и финальные. Срабатывание элементарного перехода выполняется по обычным правилам. Срабатывание абстрактного перехода “забирает” фишки из его входных позиций и порождает копию первоначально заданной сети с некоторой начальной разметкой, определяемой породившим ее переходом. Срабатывание финального перехода приводит к уничтожению сети, в которой сработал этот переход, вместе со всеми ее потомками, и как бы завершает выполнение родительского перехода, помещая фишки в выходные позиции последнего. Разметка рекурсивной сети, таким образом, представляет собой дерево, узлы которого помечены разметками для исходной сетевой структуры. Переходы могут срабатывать в любой вершине дерева разметки.

Нетрудно показать, что рекурсивные сети Хаддада и Пватрено моделируются рекурсивными NP-сетями с автономными элементами. При этом абстрактный переход моделируется перехо-

дом, помещающим новую сетевую фишку в отдельную позицию, а финальный переход моделируется переходом для вертикальной синхронизации, который эту фишку уничтожает.

Таким образом, в рекурсивных сетях Хаддада и Пватрено, в отличие от NP-сетей, структурными являются не фишки, а переходы. Этим они похожи на иерархические сети Черкасовой и Котова [27] и сети из неопубликованной работы Райнхардта [83] (по случайному совпадению они называются вложенными сетями), в которых срабатывание перехода приводит к запуску выполнения некоторой новой сети. При этом в сетях Райнхардта, как и в рекурсивных сетях Хаддада и Пватрено, уничтожение внутренней сети производится при условии достижения разметки, большей заданной (некоторый переход может сработать), что, в частности, обеспечивает выполнение свойства монотонности. Для обоих этих классов сетей разрешима проблема достижимости (путем сведения к проблеме достижимости для обыкновенных сетей Петри).

В иерархических сетях Черкасовой и Котова после запуска сети, сопоставленной переходу, нужно дождаться завершения ее работы (тупикового состояния). Это дает возможность моделировать иерархическими сетями сети Петри с ингибиторными дугами. Отсюда следует, что для иерархических сетей Петри неразрешимы не только проблема достижимости, но и проблема останова.

Напомним, что *сети Петри с обнуляющими дугами* [32] есть расширение обыкновенных сетей Петри за счет добавления специальных обнуляющих дуг (reset arcs). Обнуляющая дуга всегда направлена от позиции к переходу. При срабатывании соответствующего перехода позиция, являющаяся началом инцидентной переходу обнуляющей дуги, обнуляется, т. е. из нее убираются все находившиеся в ней фишки (если таковые имелись). В разделе 3.5 (теорема 3.9) было показано, что сети Петри с обнуляющими дугами моделируются двухуровневыми NP-сетями.

На рисунке 6.10 приведена диаграмма, иллюстрирующая сравнение по выразительности рассмотренных классов сетей Петри.

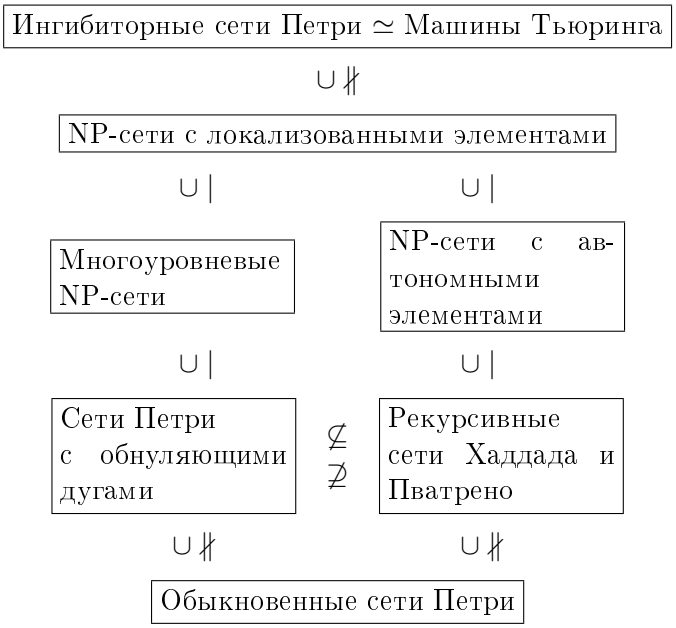


Рис. 6.10:

Перечислим результаты о разрешимости некоторых свойств для указанных классов.

Для NP-сетей с локализованными элементами разрешимы проблемы поддержки активности переходов и останова.

Для многоуровневых NP-сетей и для сетей Петри с обнуляющими дугами [32] разрешимы проблемы поддержки управляющего состояния, поддержки активности переходов, останова; неразрешимы проблемы достижимости и ограниченности.

Для рекурсивных сетей Хаддада и Пватрено разрешимы проблемы достижимости [45], поддержки активности переходов, останова (поскольку эти сети моделируются рекурсивными NP-сетями).

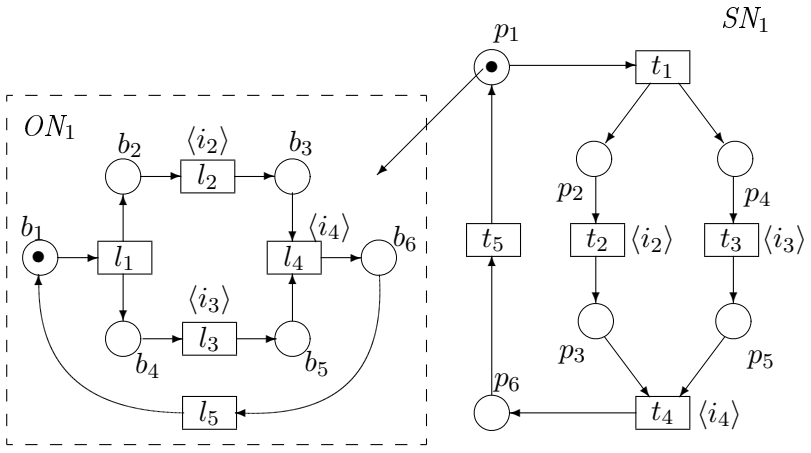


Рис. 6.11: Объектная сеть OPN_1, моделирующая разделение заданий

Хорошо известно, что для обыкновенных сетей Петри разрешимы все перечисленные выше проблемы, для сетей Петри с ингибиторными дугами все они неразрешимы.

Рекурсивные NP-сети без ограничений на локализованность элементов не показаны на рисунке 6.10. Для этого общего класса NP-сетей вопросы о разрешимости проблемы останова, а также о том, является ли вложение этого класса в класс сетей с ингибиторными дугами строгим, остаются открытыми.

6.4 Вложенные сети Петри и объектные сети Фалька

В этом разделе мы сравним вложенные сети Петри с объектными сетями (OPN — Object Petri net) Р. Фалька [92, 93] на примере задачи моделирования систем планирования заданий (task planning systems). В таких системах типичной является ситуация разделе-

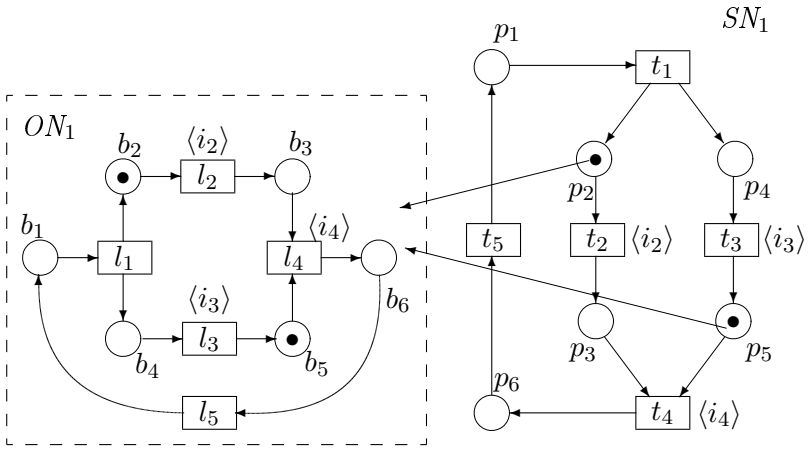


Рис. 6.12: Семантика ссылок для сети OPN_1

ния заданий, когда несколько подзадач должны быть распределены между несколькими агентами и выполнены параллельно.

На рисунке 6.11 представлена объектная сеть OPN_1, моделирующая простую ситуацию разделения заданий, когда два задания даются двум агентам.

Сеть OPN_1 состоит из системной сети SN_1 с сетевой фишкой (объектной сетью) ON_1 в позиции p_1 в начальной разметке. Переходы в сетях SN_1 и ON_1, которые должны срабатывать синхронно, помечены одной и той же меткой в треугольных скобках.

В начальной разметке, изображенной на рисунке 6.11, переходы t_1 в сети SN_1 и l_1 в сети ON_1 могут сработать независимо (автономные срабатывания). После этого системная сеть SN_1 содержит две фишки в позициях p_2 и p_4. Эти фишки интерпретируются или как ссылки на объектную сеть ON_1 с двумя фишками в позициях b_2 и b_4 (семантика ссылок), или как два процесса сети ON_1 (семантика значений). После этого одновременно срабатывают или переходы t_2 и l_2 в сетях SN_1 и ON_1, или переходы t_3 и l_3 в SN_1 и ON_1 (взаимодействующее срабатывание). На рисунке 6.12

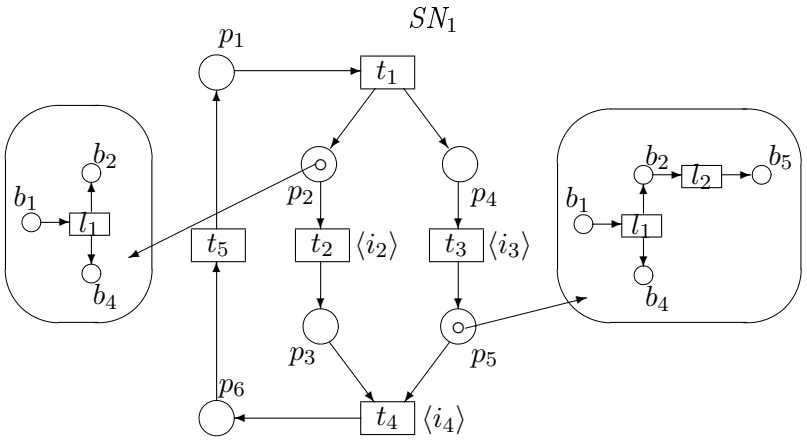


Рис. 6.13: Семантика значений для OPN_1

показан результат взаимодействия переходов t_3 и l_3 согласно семантике ссылок. При этом две фишки в позициях p_2 и p_5 ссылаются на одну и ту же объектную сеть с разметкой, содержащей фишки в позициях b_2 и b_5.

Соответствующая разметка, полученная согласно семантике значений, приведена на рисунке 6.13. В отличие от семантики ссылок, срабатывание перехода t_1 в соответствие с семантикой значений порождает две копии процесса — параллельного исполнения сети ON_1 в позициях p_2 и p_4. Согласно семантике значений переход t_4, "объединяющий" два процесса, может сработать только если в некоторый момент позиции p_3 и p_4 содержат два совместимых процесса, т. е. два процесса, для которых существует общий "содержащий" их процесс. Для рассматриваемого здесь простого примера обе семантики определяют одно и то же поведение системы (как в терминах процессов, так и в терминах последовательных исполнений).

На рисунке 6.14 приведен другой пример: OPN-сеть OPN_2 с той же системной сетью SN_1 и с объектной сетью ON_2, которая

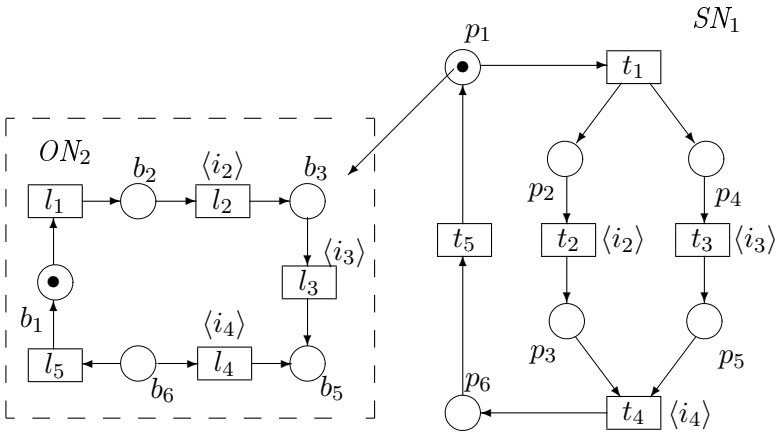


Рис. 6.14: Объектная сеть OPN_2 с последовательным исполнителем

может исполнять задачи l_2 и l_3 только последовательно. Поведение сети $\mathcal{N}_2$ согласно семантике ссылок строится из последовательных срабатываний переходов t_1 и l_1 (в некотором порядке), затем происходят взаимодействия (l_2, t_2), (l_3, t_3), (l_4, t_4), далее — автономные срабатывания l_5, t_5 (в некотором порядке), и затем система возвращается к начальной разметке (семантика интерливинг). Однако согласно семантике значений срабатывание перехода t_3 после срабатываний $l_1, t_1, (l_2, t_2)$ невозможно, так как в этом случае позиция p_4 в сети SN_1 содержит процесс, в котором переход l_3 не является активным.

Содержательно моделирование разделения задач OPN-сетью OPN_1 можно интерпретировать следующим образом: объектная сеть ON_1 моделирует параллельное исполнение заданий t_2, t_3, определенных системной сетью SN_1. Более того, согласно семантике ссылок, в любой момент работы системы имеется ровно одна копия объектной сети, и все фишки в системной сети ссылаются на нее. Фактически, таким образом, имеется одна общая ссылка

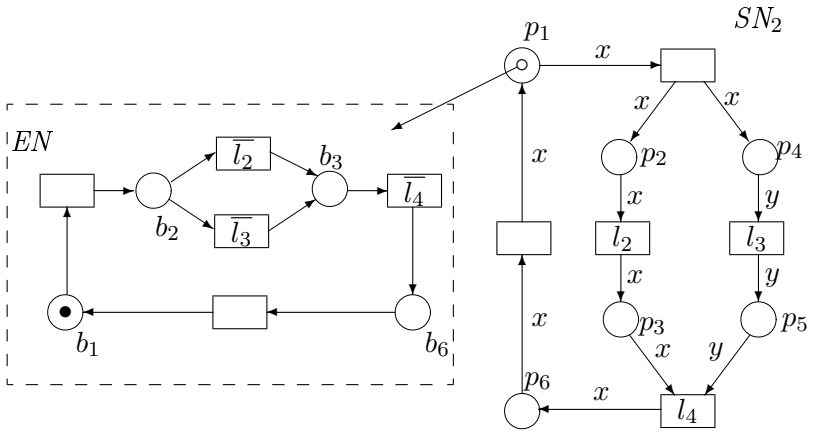


Рис. 6.15: NP-сеть NPN_3, моделирующая разделение заданий

на сеть, с которой взаимодействует системная сеть. Следовательно, с точки зрения поведения всей системы, роли системной и объектной сетей симметричны. Однако при моделировании распределения задач бывает важно явно указать, какую задачу выполняет какой исполнитель, например, в OPN-сети OPN_1 фишка в позиции p_2 системной сети связывается с фишкой в позиции b_2 объектной сети. В семантике ссылок это не может быть сделано явно.

С другой стороны, семантика значений требует от системной и объектной сетей одинакового каузального поведения путем порождения нескольких виртуальных копий истории поведения объектной сети. Таким образом, семантика значений нарушает справедливый для классических сетей Петри принцип Маркова. В марковских системах следующее состояние системы зависит только от его текущего состояния. Нарушение этого принципа порождает проблемы, например, при обработке отказов.

На рисунке 6.15 представлен другой способ моделирования той же ситуации разделения задач NP-сетью NPN_3. Главное отличие

здесь заключается в том, что задания t_2 и t_3 распределяются между двумя отдельными исполнителями, каждый из которых может выполнить любое задание. Сетевые фишки в этом примере — маркированные сети Петри, и срабатывание перехода l_1 создает две копии элементной сети EN, которые далее ведут себя независимо.

Системная сеть SN_2 в NP-сети NPN_3 аналогична системной сети SN_1 в OPN-сети OPN_1. Отличие только в приписанных дугам переменных. Элементная сеть EN в сети NPN_3 отличается от объектной сети ON_1 в OPN-сети OPN_1, так как она представляет одного отдельного агента, а не двух агентов сразу, как в случае OPN-сети OPN_1. Поэтому ветвление (выбор между двумя возможными заданиями) "перемещается" с перехода l_1 в OPN-сети OPN_1 на позицию b_2 в NP-сети NPN_3. Аналогичные изменения сделаны и в объединяющем переходе.

Легко заметить, что OPN-сеть OPN_1 и NP-сеть NPN_3, приведенные на рисунках 6.11, 6.15, моделируют одно и то же поведение системы. Преимущество NP-сетей, по нашему мнению, состоит в более ясной и простой семантике. Важным моментом является также то, что элементная сеть EN в NP-сети моделирует поведение отдельного агента. Это делает систему более структурированной. Здесь копии фишек представляют агентов, поведение которых определяется их текущим "персональным" состоянием. Поэтому NP-сеть с такими агентами может быть легко обобщена до системы с большим числом возможно конкурирующих за получение задания агентов.

Конечно, возможны ситуации, когда необходимо моделировать выполнение одним исполнителем нескольких заданий параллельно. В этом случае такой исполнитель как бы "распределен" по позициям системной сети, представляющим разные задания. Но такую ситуацию, по нашему мнению, лучше моделировать NP-сетями с помощью механизма горизонтальной синхронизации. На рисунке 6.16 показана NP-сеть NPN_4 с двумя сетевыми фишками в позиции q системной сети и с переходами, помеченными метками

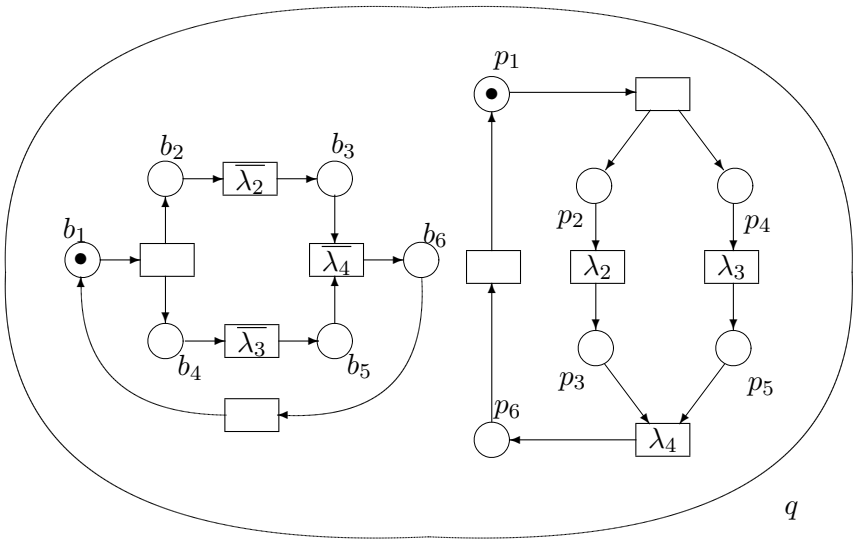


Рис. 6.16: NP-сеть NPN_4

для горизонтальной синхронизации. Эти сетевые фишки совпадают с сетями SN_1 и ON_1 OPN-сети OPN_1 на рисунке 6.11. Поведение NP-сети NPN_4 соответствует поведению сети OPN_1 согласно семантике ссылок. Нетрудно заметить, что путем аналогичного "помещения" сетей SN_1 и ON_2 (см. рисунок 6.14) с переходами, помеченными подходящим образом для горизонтальной синхронизации, в одну общую позицию системной сети можно получить NP-сеть, поведение которой совпадает с поведением сети OPN_2 согласно семантике ссылок.

6.5 Вложенные сети Петри и Линейная логика

Линейная логика Жирара (J.-Y. Girard, [44]) отличается от классической логики тем, что рассматривает посылки вывода как рас-

ходуемые ресурсы. В то время как в классической логике выведенная или изначально заданная посылка может использоваться для вывода новых формул неограниченное число раз, в Линейной логике использование формулы в качестве посылки ограничено. Для этого в Линейной логике рассматриваются два вида конъюнкций и дизъюнкций, а также модальность для представления возможности использовать формулу бесконечное число раз (как в классической логике).

Ресурсно-зависимый характер Линейной логики обуславливает ее связь с формализмом сетей Петри. Известно, что семантика обыкновенных сетей Петри может быть описана в терминах Линейной логики. В этом параграфе будет дано краткое изложение результатов об описании посредством Линейной логики семантики вложенных сетей Петри.

Приведем содержательное толкование используемых далее связок Линейной логики. *Мультипликативная конъюнкция* $\otimes$ имеет тесную связь с операцией сложения мультимножеств и используется для представления объединения ресурсов. Далее будем использовать обозначение:

$$a^n = \underbrace{a \otimes a \otimes \ldots \otimes a}_{n \text{ раз}}.$$

Единица относительно мультипликативного умножения обозначается $\mathbf{1}$.

Линейная импликация $\multimap$ описывает использование имеющихся ресурсов для получения новых и играет в Линейной логике большую роль. Применение линейной импликации возможно при наличии ресурсов, описанных в ее левой части, и приводит к использованию этих ресурсов и получению ресурсов, соответствующих правой части линейной импликации. Такое преобразование задает линейную функцию, что и послужило основанием для термина ‘Линейная логика’ (при этом слово ‘Линейная’ традиционно пишется с заглавной буквы).

Другая важная конструкция, используемая, в частности, при описании семантики сетей Петри — модальность '*конечно*' (обозначается ! — *of course*). Эта модальность означает возможность использовать формулу в качестве ресурса бесконечное число раз.

В своих работах Жирар приводит следующий простой пример вывода в Линейной логике. Пусть D обозначает доллар, C — пачку сигарет. Тогда $D \otimes D$ означает 2 доллара, формула $!(D \multimap C)$ описывает возможность получить за один доллар пачку сигарет. Вывод $D \otimes !(D \multimap C) \vdash C \otimes !(D \multimap C)$ соответствует покупке одной пачки сигарет (возможность совершить покупку при этом сохраняется, хотя долларов в данном случае больше нет).

Для описания семантики сетей Петри используется следующий фрагмент $\mathbf{ILL}_{\mathrm{PN}}$ секвенциального исчисления интуиционистской Линейной логики:

(Тождество)

$$A \vdash A$$

(**1**)

$$\vdash \mathbf{1}$$

(Сечение)

$$\frac{\Gamma \vdash A \quad \Delta, A \vdash B}{\Gamma, \Delta \vdash B}$$

(Перестановка)

$$\frac{\Gamma, A, B, \Delta \vdash C}{\Gamma, B, A, \Delta \vdash C}$$

($\otimes$Л)

$$\frac{\Gamma, A, B \vdash C}{\Gamma, A \otimes B \vdash C}$$

($\otimes$П)

$$\frac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}$$

$(\multimap)$

$$\frac{\Gamma \vdash A \quad \Delta, B \vdash C}{\Delta, \Gamma, A \multimap B \vdash C}$$

(Сокращение)

$$\frac{\Gamma, !A, !A \vdash B}{\Gamma, !A \vdash B}$$

(Оставление)

$$\frac{\Gamma, A \vdash B}{\Gamma, !A \vdash B}$$

(Ослабление)

$$\frac{\Gamma \vdash B}{\Gamma, !A \vdash B}$$

Далее мы покажем, как обыкновенная сеть Петри может быть представлена формулой Линейной логики.

Пусть $PN = (P, T, F, W)$ — обыкновенная сеть Петри, M_0 — ее начальная разметка. Сети PN с начальной разметкой M_0 сопоставим формулу $\Psi_{\mathbf{ILL}_{\mathrm{PN}}}(PN, M_0)$ Линейной логики, называемую *канонической формулой маркированной сети* (PN, M_0).

Каноническая формула сети (PN, M_0) определяется как мультипликативная конъюнкция следующих конъюнктов.

- Для каждого перехода $t \in T$ построим конъюнкт

$$! \left(\bigotimes_{p \in {}^\bullet t} p_{W(p,t)} \multimap \bigotimes_{q \in t^\bullet} q_{W(t,q)} \right).$$

- Для каждой позиции $p \in P$, содержащей фишки при начальной разметке M_0, построим конъюнкт $p^{M_0(p)}$. Тогда всей начальной разметке M_0 будет соответствовать подформула

$$\bigotimes_{p \in P} p^{M_0(p)}.$$

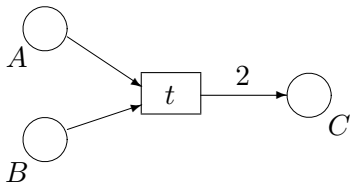


Рис. 6.17: Сеть Петри, состоящая из одного перехода

Так, например, формула

$$!(A \otimes B \multimap C \otimes C)$$

соответствует переходу, изображенному на Рис. 6.17.

В работах [26, 69] было показано, что достижимость заданной разметки для данной маркированной обыкновенной сети Петри соответствует выводимости соответствующей секвенции во фрагменте $\mathbf{ILL}_{\mathrm{PN}}$ Линейной логики. Более точно, имеет место

Теорема 6.6. *Если разметка M достижима в сети (PN, M_0), то секвенция $\Psi_{\mathbf{ILL}_{\mathrm{PN}}}(PN, M_0) \vdash \Psi_{\mathbf{ILL}_{\mathrm{PN}}}(PN, M)$ выводима в $\mathbf{ILL}_{\mathrm{PN}}$.*

Для описания семантики сетей Петри высокого уровня, в которых фишки могут иметь разные цвета, а для переходов определяются разные режимы срабатывания, потребуется уже предикатный вариант Линейной логики. В этом случае формулы могут содержать переменные, и все свободные переменные по умолчанию предполагаются связанными квантором всеобщности.

Покажем, как раскрашенные сети Петри с конечным набором цветов и конечным множеством индивидуальных атомарных фишек могут быть представлены формулами Линейной логики.

Для этого расширим язык $L(\mathbf{ILL}_{\mathrm{PN}})$ интуиционистского фрагмента Линейной логики таким образом, чтобы иметь возможность выразить не просто наличие тех или иных ресурсов, но и степень

их доступности. С этой целью добавим в предикатный язык Линейной логики специальную операцию P, выражающую принадлежность (от англ. *posses* — обладать). В случае кодирования сети Петри формулой Линейной логики запись $P_A(x)$ будет использоваться для выражения того, что позиция A содержит фишку x.

Этот же подход будет далее использоваться для кодирования формулами Линейной логики вложенных сетей Петри. Поскольку фишки во вложенной сети Петри сами могут являться сетями, то формула вида $P_A(\varphi)$ будет означать, что сетевая фишка, описываемая формулой φ, находится в позиции A. Расширение Линейной логики добавлением операции P будем называть *распределенной Линейной логикой*.

Заметим, что распределенная Линейная логика может быть интересна не только как средство описания семантики сложных расширений формализма сетей Петри. Она имеет естественную интерпретацию и может быть использована в самых разных приложениях. Пусть φ — формула Линейной логики, представляющая некоторый ресурс, A — имя владельца ресурса (или возможное местоположение ресурса). Тогда формула $P_A(\varphi)$ означает, что A владеет ресурсом φ.

Продолжая пример Жирара с долларами и сигаретами, запишем в распределенной Линейной логике:

$P_A(D)$ — A имеет доллар;

$P_A(!(D \multimap C))$ — A может покупать (сколько угодно раз) сигареты по цене доллар за пачку;

$P_A(D) \multimap P_B(D)$ — A может отдать свой доллар B;

$(\forall x)(P_A(x) \multimap P_B(x))$ — A готов отдать B что угодно (но только один раз).

Определим описываемое расширение языка Линейной логики более точно. Пусть имеется набор $\mathbf{A} = \{A_1, A_2, \dots\}$ атомарных символов, которые представляют владельцев и/или места расположения ресурсов, набор $\mathbf{R} = \{a, b, c, \dots\}$ атомов, представляющих ресурсы и набор $\mathbf{X} = \{x, y, \dots\}$ переменных. Операцию P_{A_i},

где $A_i \in \mathbf{A}$, можно рассматривать как параметризованную модальность. Формулы (термы) распределенной Линейной логики определим по индукции:

- атомы из $\mathbf{R}$ и переменные из X — термы;
- если φ — терм Линейной логики и $A_i \in \mathbf{A}$, то $P_{A_i}(\varphi)$ — тоже терм;
- если φ, ψ — термы Линейной логики, то $\varphi \otimes \psi$, $\varphi \multimap \psi$, $!\varphi$, $(\forall x)\varphi$, $(\exists x)\varphi$ — тоже термы Линейной логики.

Как обычно в секвенциальных исчислениях определим формулу как выражение вида $\Gamma \ \vdash \ \phi$, где Γ — последовательность термов, ϕ — терм.

Переменные в формулах интерпретируются как термы. Исчисление $\mathbf{DILL}_{\mathrm{PN}}$ получается из $\mathbf{ILL}_{\mathrm{PN}}$ добавлением следующего правила подстановки терма вместо переменных:

$$\frac{\Gamma \ \vdash \ \phi}{\Gamma[\psi/x] \ \vdash \ \phi[\psi/x]}$$

Определим каноническую формулу Линейной логики для раскрашенной сети Петри CPN. Без ограничения общности можно считать, что каждой дуге в сети CPN приписана либо константа, либо переменная (тогда выражение вида $x + 2y$ может быть представлено кратными дугами).

Определение 6.6. *Каноническая формула* $\Psi_{\mathbf{DILL}_{\mathrm{PN}}}(CPN, M)$ для раскрашенной сети Петри $CPN = (\Omega, N, C, W, G, M_0)$ с текущей разметкой M определяется как мультипликативная конъюнкция следующих сомножителей:

- Для каждого перехода $t \in T$ строится множитель

$$! \left(\bigotimes_{p \in {}^\bullet t} p(W(p,t)) \multimap \bigotimes_{q \in t^\bullet} q(W(t,q)) \right),$$

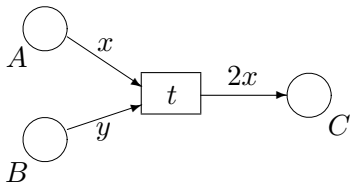


Рис. 6.18: Раскрашенная сеть Петри с одним переходом

- Для представления текущей разметки M каждой позиции $p \in P$ и фишке a такой, что $a \in M(p)$, сопоставляется множитель $p(a)$. Тогда вся разметка представляется подформулой

$$\bigotimes_{\substack{p \in P \\ a \in M(p)}} p(a),$$

где кратные вхождения фишек представляются кратными вхождениями соответствующих множителей в конъюнктивное произведение.

Так, например, формула

$$!(A(x) \otimes B(y) \multimap C(x) \otimes C(x))$$

соответствует переходу, изображенному на рисунке 6.18.

Можно показать, что каноническая формула для раскрашенной сети Петри $CPN = (\Omega, N, C, W, G, M_0)$ удовлетворяет следующему свойству: формула

$$\Psi_{\mathbf{DILL}_{\mathrm{PN}}}(CPN, M_0) \vdash \Psi_{\mathbf{DILL}_{\mathrm{PN}}}(CPN, M)$$

выводима тогда и только тогда, когда разметка M достижима для маркированной сети CPN с начальной разметкой M_0.

Перейдем теперь к описанию в терминах Линейной логики семантики вложенных сетей Петри, используя подход, описанный в [37, 15].

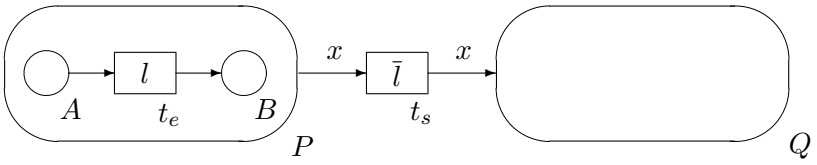


Рис. 6.19: Пример вложенной сети Петри

Простой пример вложенной сети Петри приведен на рисунке 6.19. Здесь позиция P содержит сетевую фишку. Единственный переход t_e этой сетевой фишки помечен меткой l, что означает, что этот переход может сработать только синхронно с переходом t_s, помеченным дополнительной меткой $\bar{l}$.

Для описания семантики вложенных сетей Петри к обычным правилам Линейной логики достаточно добавить следующее правило, которое означает, что если из φ можно получить ψ, и A имеет φ, то A может получить ψ:

$$\frac{\varphi \ \vdash \ \psi}{P_{A_i}(\varphi) \ \vdash \ P_{A_i}(\psi)}$$

В качестве иллюстрации приведем формулу $\Psi_{\mathbf{DILL}_{\mathrm{PN}}}(NPN, M_0)$ распределенной Линейной логики, кодирующую вложенную сеть Петри NPN, изображенную на рисунке 6.19. Метки синхронизации l и $\bar{l}$ используются в ней как атомарные ресурсы выделенного типа.

$$\begin{aligned}
\Psi_{\mathbf{DILL}_{\mathrm{PN}}}(NPN, M_0) \equiv\ & !(l \otimes A \multimap \bar{l} \otimes B) \otimes && (\textit{переход } t_e)\\
& !(((lx \multimap \bar{l} \otimes y) \multimap P(x)) \multimap Q(y)) \otimes && (\textit{переход } t_s)\\
& P(!(l \otimes A \multimap \bar{l} \otimes B) \otimes A \otimes A) && (\textit{разметка})
\end{aligned}$$

Из этой формулы выводима следующая формула, кодирующая сеть NPN с разметкой после синхронного срабатывания переходов t_e и t_s:

$$!(l \otimes A \multimap \bar{l} \otimes B) \otimes \qquad (\textit{переход } t_e)$$

$$!(((l \otimes x \multimap \bar{l} \otimes y) \multimap P(x)) \multimap Q(y)) \otimes \qquad (\textit{переход } t_s)$$
$$Q(!(l \otimes A \multimap \bar{l} \otimes B) \otimes A \otimes B) \qquad (\textit{новая разметка})$$

В этом выводе применяется правило подстановки формулы вместо свободной переменной. Правило применятся для подстановок $x = !(l \otimes A \multimap \bar{l} \otimes B) \otimes A \otimes A$; $y = !(l \otimes A \multimap \bar{l} \otimes B) \otimes A \otimes B$.

Теорема 6.7. *Разметка M достижима для маркированной вложенной сети (NPN, M_0) в том и только том случае, когда секвенция $\Psi_{\mathbf{DILL}_{\mathrm{PN}}}(NPN, M_0) \vdash \Psi_{\mathbf{DILL}_{\mathrm{PN}}}(NPN, M)$ выводима в $\mathbf{DILL}_{\mathrm{PN}}$ — дистрибутивном расширении интуиционистского фрагмента Линейной логики $\mathbf{ILL}_{\mathrm{PN}}$.*

Таким образом, распределенная Линейная логика позволяет описывать семантику вложенных сетей Петри в терминах выводимости подобно тому, как обычная Линейная логика описывает семантику обыкновенных сетей Петри. Вместе с тем, распределенная Линейная логика представляет и самостоятельный интерес как формальное средство описания ресурсной зависимости в распределенных системах. При этом для использования ее в конкретных предметных областях можно не только менять интерпретацию модальности P , но и добавлять специальные правила вывода. Например, правило

$$\frac{\Gamma \vdash A(\varphi), \Gamma \vdash A(\psi)}{\Gamma \vdash A(\varphi \otimes \psi)}$$

не применимо для моделирования поведения сетей Петри, но может быть полезно в других приложениях.

Основываясь на идеях объектных сетей Р. Фалька [92] и представлении сетей Петри формулами Линейной логики Б. Фарвер [33, 34] определил класс LLP-сетей (Linear Logic Petri nets). В сетях Фарвера системная сеть является сетью Петри высокого уровня, а фишки представлены формулами Линейной логики, описывающими поведение объектов. Фарвер доказал, что LLP-сети так же, как и обыкновенные сети Петри, могут быть транслированы

в формулы Линейной логики. Вместе с тем использование охран в виде формул Линейной логики в LLP-сетях существенно расширяет выразительность модели.

В работах [35, 36] Б. Фарвер также установил, что между независимо определенными классами объектно-ориентированных сетей — определенными им LLP-сетями и вложенными сетями Петри — существует тесная взаимосвязь, обусловленная возможностью описания семантики той и другой модели в терминах выводимости в фрагменте $\mathbf{ILL}_{\mathrm{PN}}$ Линейной логики. В частности, Б. Фарвер доказал, что двухуровневые NP-сети можно транслировать в поведенчески эквивалентные LLP-сети.

Обратная симуляция NP-сетями возможна для сильно ограниченного класса LLP-сетей, поскольку определение LLP-сетей является весьма общим. Оно допускает наличие кратных переменных на входных дугах, что (как было показано в утв. 6.2) приводит к универсальным вычислительным моделям, и, более того, допускает использование охран переходов без каких-либо ограничений на эффективность проверки описываемых ими условий. В [35, 36] приведены некоторые достаточные ограничения на LLP-сети, делающие возможной их симуляцию NP-сетями.

Библиографический комментарий: сети Петри и объектно-ориентированный подход

В этом параграфе приводится краткий обзор различных расширений и модификаций сетей Петри для описания поведения распределенных систем со сложной структурой и мультиагентных систем.

Понятия *агента* и *мультиагентной системы* становятся одними из основных понятий как искусственного интеллекта, так и областей информатики, связанных с разработкой больших программных систем. В широком смысле слова считается, что агент – это нечто, наделенное элементами интеллектуального поведения и способное взаимодействовать с другими агентами. В узком смысле агентом называют обычно компьютерную систему, обладающую следующими свойствами [94]:

- *автономность*: агенты действуют без прямого вмешательства человека или какого-нибудь другого общего управления, управление их действиями и состояниями носит локальный характер;

- *социальный характер поведения*: агенты взаимодействуют с другими агентами либо путем обмена информацией, либо посредством прямых коммуникаций и совместного выполнения некоторых действий;

- *реактивность*: агенты могут воспринимать окружение, в котором находятся (это может быть и внешняя среда, и пользователь, и другие агенты, а также произвольная комбинация всего перечисленного), и реагировать на его изменение;

- *инициативность поведения*: агенты могут действовать не только в ответ на запросы окружения, но и проявлять собственную активность и иметь собственные цели.

Понятию мультиагентной системы можно сопоставить понятие распределенной системы [61], используемое в теории программных систем. Компоненты распределенной системы находятся в различных точках пространства (физического или мыслимого) и действуют в соответствии с собственными программами. При этом, естественно, интерес представляют такие системы, в которых компоненты не полностью независимы, а могут взаимодействовать, передавая друг другу информацию и/или выполняя некоторые действия совместно. Таким образом, важнейшими свойствами распределенных систем являются отсутствие единой для всех компонент (агентов) шкалы времени и наличие механизма взаимодействия агентов между собой. Отсюда видно, что понятия мультиагентной и распределенной систем тесно связаны между собой.

При использовании мультиагентных систем для моделирования задач искусственного интеллекта обычно говорят об интеллектуальных агентах и наделяют их такими характеристиками, как знание, вера, цель, интерес и др. Если же отвлечься от такого рода модальных свойств и считать, что поведение агента задается некоторой программой (возможно, недетерминированной), то

опыт, накопленный в теории моделирования и анализа распределенных систем, может оказаться весьма полезным для формального описания поведения мультиагентных систем.

В теоретической информатике известен достаточно большой набор формализмов для представления распределенных систем. Важный класс моделей распределенных вычислительных систем составляют модели процессов [61]. В таких моделях предполагается, что система состоит из нескольких последовательных процессов, которые в ходе своей работы могут взаимодействовать либо путем передачи сообщений через некоторый канал, либо путем доступа к глобальным переменным (shared variables), либо посредством синхронных коммуникаций (handshaking).

Принципиально другой подход к моделированию распределенных систем представлен сетями Петри [84]. Отдельная сеть Петри может рассматриваться и как одноагентная, и как мультиагентная система. Все зависит от точки зрения. Мультиагентная система характеризуется главным образом распределенным поведением, когда нет управляющего устройства, которое в каждый момент времени указывает на определенную точку в программе. Если описывать поведение сети Петри посредством последовательной семантики (исполнение есть последовательность срабатываний переходов), то это есть одноагентная система с недетерминированным поведением.

С другой стороны, на ту же сеть можно смотреть как на мультиагентную систему, если считать, что каждый переход сети моделирует поведение отдельного агента. В этом случае распределенное поведение всей системы описывается посредством так называемых сетей событий [84, 87], т. е. используется каузальная семантика для сетей Петри.

Мы назвали два крайних случая. В более общей ситуации поведение каждого отдельного агента может описываться своей собственной сетью Петри и возникает задача описания механизмов взаимодействия между такими сетями.

Во многих работах предлагаются средства структурирования и повышения уровня абстрактности сетей Петри на синтаксическом уровне. Это облегчает проектирование реальных сложных систем и не выводит за рамки выразительных возможностей сетей Петри. Часто такие расширенные сети путем структурных преобразований могут быть сведены к обыкновенным сетям Петри, после чего могут использоваться традиционные средства анализа. Классическим примером такого расширения являются иерархические раскрашенные сети Йенсена [51]. В этих сетях переход может быть структурным — ему сопоставляется отдельная раскрашенная сеть (страница), которая задает более детальное описание действия, задаваемого переходом. Это позволяет создавать модульные иерархические описания. При этом доказано, что для всякой иерархической раскрашенной сети существует поведенчески эквивалентная неиерархическая раскрашенная сеть. Чтобы получить такую неиерархическую сеть, достаточно просто заменить каждый составной переход (и инцидентные ему дуги) экземпляром соответствующей подстраницы, "склеивая" позиции-порты этой страницы и позиции, смежные исходному переходу. Таким образом, составные переходы можно рассматривать как макроопределения. Иерархические конструкции в раскрашенных сетях задают абстрагирование на синтаксическом, а не на семантическом уровне. Между выражениями на дугах, инцидентных структурному переходу, и поведением сети, заданной соответствующей подстраницей, вообще говоря, нет формальной семантической взаимосвязи.

Другим примером являются иерархические сети Петри, рассмотренные Р. Фелингом в [38]. Для облегчения моделирования больших реальных систем Р. Фелинг расширяет сети Петри за счет конструкций для уточнения позиций и переходов. С теоретической точки зрения такие обобщения могут рассматриваться как удобная графическая нотация, которая не увеличивает возможности моделирования (выразительность). Целью такого расширения

является поддержка методов пошаговой разработки сетевой модели путем последовательного уточнения.

Известны и теоретические обобщения стандартных сетей Петри путем структурирования переходов. Отметим в этой связи иерархические сети Черкасовой и Котова [27], в которых срабатывание абстрактного перехода приводит к запуску сети, задающей его уточнение. Завершение работы внутренней сети и продолжение исполнения абстрактной сети в этой модели происходит при условии достижения внутренней сетью тупикового состояния. Такое свойство иерархических сетей нарушает условие монотонности сетей Петри. Условие монотонности содержательно можно представить следующим образом: при бóльшем количестве ресурсов, представленных фишками сети, для сети возможны все те исполнения, что и при первоначальном запасе ресурсов, и, возможно, еще некоторые дополнительные. Проверка наличия дедлока во внутренней сети равносильна проверке некоторой позиции на пустоту (отсутствие фишек). Известно [8], что иерархические сети Петри взаимно сводимы с сетями Петри с ингибиторными дугами, и, следовательно, равны по выразительности машинам Тьюринга.

В работе Хаддада и Пватрено [45] введены рекурсивные сети Петри, в которых также возможно "абстрактное" срабатывание перехода, приводящее к запуску исполнения копии первоначально заданной сети. При этом в рекурсивных сетях Хаддада и Пватрено уничтожение внутренней сети производится при условии достижения разметки, большей заданной (условие, что некоторый переход может сработать). Такое определение поддерживает свойство монотонности, и для сетей Хаддада и Пватрено разрешима не только проблема останова, но и проблема достижимости. При этом эти сети строго более выразительны, чем обыкновенные сети Петри.

В работах Н.А. Анисимова [1] для структурного описания поведения протоколов используется алгебра регулярных макросетей, построенная на основе алгебры регулярных сетей Петри [8] и

позволяющая компактно описывать структуру управления сложных протоколов. На основе операций алгебры структур (включая операции с параметрами и переменными) на модели сетей Петри высокого уровня строятся правила композиции протоколов, сохраняющие их корректность и являющиеся мощным инструментом композиционного подхода.

Таким образом, среди подходов к объединению нескольких сетей Петри в одну систему можно выделить модульную композицию посредством сопряжения позиций или переходов, а также иерархические сети. При этом, как правило, сетевые агенты моделируются сетями Петри одного и того же класса. Однако во многих приложениях естественно возникают системы, в которых разные агенты моделируются сетями Петри разных классов. В этом случае обычная операция сопряжения переходов может оказаться недостаточной и возникает необходимость в новом механизме синхронизации переходов, который в [47] назван мульти-модельной синхронизацией. Определенные трудности возникают также при определении механизма синхронизации агентов, представленных сетями Петри высокого уровня и раскрашенными сетями Петри. В этих классах сетей через пометки на дугах можно выбрать агента для коммуникации или обращаться к нескольким агентам одновременно. Вопросы, связанные с формальной семантикой таких систем, пока остаются открытыми.

Широко распространенной и эффективной парадигмой для поддержки разработки реальных сложных и мультиагентных систем является объектно-ориентированный подход. Многие исследования по расширению сетей Петри направлены на расширение формализма сетей Петри за счет объектно-ориентированных конструкций, таких как абстракция, инкапсуляция, наследование и др. с целью получения структурированных моделей, явно отражающих иерархическую и мультиагентную структуру системы (см. обзор [95], а также [20, 62, 18]).

Среди первых работ по сочетанию формализма сетей Пет-

ри и понятий и конструкций объектно-ориентированного подхода отметим систему OBJSA/CLOWN [23, 22]. В OBJSA сетях, предложенных Е. Баттистоном, фишки представлены алгебраическими спецификациями. OBJSA сети соответствуют семантической модели, описываемой посредством алгебраической нотации, а CLOWN (CLass Orientation With Nets) задает синтаксические описания. Поведение класса (в объектно-ориентированном смысле) задается автоматной сетью. Для композиции сетевых модулей используется сопряжение переходов.

Подобно языку CLOWN, язык спецификаций CO-OPN/2 (Concurrent Object-Oriented Petri Nets/2) [24, 25] основывается на алгебраических спецификациях и сетях Петри, которые объединяются путем, аналогичным тому, как это делается в алгебраических сетях Петри [86]. Отличие его состоит в том, что CO-OPN/2 поддерживает абстрактные типы данных и допускает их использование в других классах, при этом определенные там методы используются как интерфейсные переходы. В языке CO-OPN/2 объект определяется как инкапсулированная алгебраическая сеть (не обязательно автоматная), в которой позиции задают внутреннее состояние, а переходы моделируют параллельные действия объектов. Важной особенностью этой системы является также возможность динамического порождения объектов.

В G-сетях [80] и кооперативных сетях (cooperative nets) [89] взаимодействие модулей, представляющих объекты, организовано по принципу клиент-сервер через взаимодействие позиций, а не переходов. Для этих видов сетей возможны трансформации, переводящие их в поведенчески эквивалентные "обычные" сети Петри.

Ш. Лакос определил класс объектно-ориентированных сетей LOOPN++ [58, 59] (Language for Object-Oriented Petri Nets). Одной из главных характеристик LOOPN++ является наличие абстрактных позиций и переходов, используемых для представления вложенной структуры сетей, при этом LOOPN++ явно поддерживает дуальность позиций и переходов в сетях Петри. Семантика

объектных сетей Лакоса определяется не путем трансформаций в раскрашенные сети Петри, а в терминах абстрактных разметок. LOOPN++ хорошо поддерживает пошаговую технологию проектирования систем путем последовательного уточнения. При этом абстрактная сеть накладывает ограничения на поведение более конкретной сети, так как в противном случае не удается поддерживать сохранение свойств разметок и последовательностей шагов срабатывания при ее конкретизации.

Отметим также несколько других подходов, ориентированных на поддержку проектирования реальных больших систем. В работе Й. Нютцеля, Б. Дэна и В. Фенглера [78] рассматривается метод CON (Concurrent Object Net) — объектно-ориентированный метод проектирования распределенных и встроенных систем – графическое представление классов, объектов и их взаимодействия. Сети Петри используются для внутреннего скрытого представления спецификации, что дает разработчику возможность симуляции и верификации системы с помощью инструментария для сетей Петри. При этом спецификация на языке объектных сетей сначала преобразуется в одноуровневую "плоскую" объектную сеть, а затем транслируется в сеть Петри высокого уровня.

Дж. Хонг и Д. Баэ [48] определяют сети HOONets (Hierarchical Object-Oriented Petri Net for SYstem Modeling and Analysis). Это сети Петри высокого уровня, поддерживающие объектно-ориентированные конструкции. Поведение класса описывается посредством модификации раскрашенной сети Петри, которая может содержать, помимо обычных, абстрактные позиции, переходы и фишки. Уточнение (refinement) абстрактных позиций и переходов производится путем сопоставления им сетей такого же вида, при этом сети, сопоставляемые позициям (переходам) должны иметь выделенные входной и выходной переход (соответственно, входную и выходную позицию). Абстрактные фишки представляют собой кортеж, составленный из обычных цветных фишек (тип данных "запись"). Для сетей HOONets определен алгоритм раз-

вертки (unfolding) в обычную раскрашенную сеть Петри. Анализ HOONets сетей (проверка свойств достижимости, живости переходов, наличия дедлоков и др.) предлагается проводить путем порождения графа достижимости

В отличие от сетей со структурными или абстрактными переходами и/или позициями в Объектных сетях Р. Фалька [92] структурными являются фишки, задающие разметку сети. При этом фишки сами могут являться сетями и иметь автономное поведение. Поскольку, очевидно, число фишек в ходе работы сети Петри может меняться, такая постановка приводит к системам с порождаемыми и исчезающими объектами. Объектные сети Фалька ориентированы на решение специальных задач планирования. Объекты (сетевые фишки) представляют в них подзадачи, причем объекты в сетях Фалька в определенном смысле не локализованы в позициях системной сети, что делает определение состояния сети и правил срабатывания переходов достаточно сложными. Проблему представляет и распространение семантики на многоуровневый случай. Сравнение объектных сетей Фалька с вложенными сетями Петри приведено в разделе 6.4.

В работах К.-П. Ноендорфа, Д. Киритсиса и П. Ксирухакиса [76, 77] рассматривается класс сетей Хамелеон (Chameleon systems), основанный на объектных сетях Р. Фалька. На дугах системной сети в Хамелеон-сетях могут быть только переменные, при этом на них накладываются те же ограничения, что и для вложенных сетей: все переменные на входных дугах конкретного перехода должны быть попарно различны, и всякая переменная, приписанная выходной дуге, должна также встречаться на некоторой входной дуге этого же перехода. Синхронизация переходов может осуществляться двумя способами: либо один переход принадлежит системной сети, а другой — сетевой фишке, находящейся во входной позиции первого перехода, либо синхронизируются переходы в двух сетевых фишках, находящихся в одной и той же позиции системной сети.

Существенным ограничением Хамелеон-сетей является то, что системная сеть фактически представляет собой автомат, который может перемещать сетевые фишки из одной позиции в другую. Авторы отмечают, что более общий случай, когда допускается копирование, слияние и уничтожение сетевых фишек, приводит к трудностям при определении семантики. Появление новых сетевых фишек в системной сети также невозможно в Хамелеон-сетях по причине отсутствия констант на выходных дугах и механизма копирования. Для Хамелеон-сетей в силу указанных ограничений возможно естественное обобщение на многоуровневый случай. Очевидно, что число уровней в таких сетях всегда фиксировано и определяется начальной разметкой сети, так что в этом случае не приходится говорить о сетях с динамической архитектурой.

Другим классом сетей, основанным на идеях Р. Фалька, являются LLP-сети (Linear Logic Petri nets) Б. Фарвера [33, 34]. В сетях Фарвера системная сеть является сетью Петри высокого уровня, а фишки представлены формулами Линейной логики [44], описывающими поведение объектов. Как известно, специальный подкласс интуиционистской Линейной логики задает семантику сетей Петри. LLP-сети также могут быть транслированы в формулы Линейной логики. В этом смысле LLP-сети могут быть обобщены на многоуровневый случай. О связи сетей Петри и Линейной логики см. раздел 6.5 данной книги.

Вложенные сети Петри, описанные в данной книге, представлены в работах [5, 7, 11, 12, 13, 14, 15, 37, 64, 65, 66, 67].

Литература

1. *Анисимов Н.А.* Формальная модель для разработки и описания протоколов на основе теории сетей Петри // Автоматика и вычислительная техника. — 1988. №6. — С. 3–10.

2. *Ачасова С.М., Бандман О.Л.* Корректность параллельных вычислительных процессов. Новосибирск: Наука, 1990.

3. *Бандман М.К. и др.* Территориально-производственные комплексы: Прогнозирование процесса формирования с использованием сетей Петри. — Новосибирск: Наука (Сиб. отд-ние), 1990.

4. *Башкин В.А., Ломазова И.А.* О языках вложенных рекурсивных сетей Петри // Интеллектуальное управление: новые интеллектуальные технологии в задачах управления (ICIT'99). Труды международной конференции, Переславль-Залесский, 6-9 декабря 1999. — М.: Наука. Физматлит, 1999.

5. *Башкин В.А., Ломазова И.А.* Бисимуляция ресурсов в сетях Петри // Известия Российской Академии Наук. Теория и системы управления.— 2003. №4. — С. 115–123.

6. *Дистель Р.* Теория графов. Новосибирск: Издательство Института математики, 2002.

7. *Зазовская Т.А., Ломазова И.А.* О сравнительной выразительности вложенных сетей Петри и алгебр процессов // Седьмая национальная конференция по искусственному интеллекту с международным участием (КИИ-2000). Труды конференции. Том 1. — М.: Физматлит, 2000. — С. 305–314.

8. *Котов В.Е.* Сети Петри. — М.: Наука, 1984.

9. *Куратовский К., Мостовский А.* Теория множеств. — М.: Мир, 1970.

10. *Курош А.Г.* Лекции по общей алгебре. — М.: Наука, 1973.

11. *Ломазова И.А.* Моделирование мультиагентных динамических систем вложенными сетями Петри // Программные системы: Теоретические основы и приложения. — М.: Наука. Физматлит, 1999. — С. 143–156.

12. *Ломазова И.А.* Некоторые алгоритмы анализа для многоуровневых вложенных сетей Петри // Известия РАН. Теория и системы управления. — 2000. №6. — С. 965–974.

13. *Ломазова И.А.* Рекурсивные вложенные сети Петри: анализ семантических свойств и выразительность // Программирование. — 2001. №4. — С. 21–35.

14. *Ломазова И.А.* Объектно-ориентированные сети Петри: формальная семантика и анализ // Системная информатика. — 2002. №8. — С. 143–205.

15. *Ломазова И.А.* Моделирование ресурсной зависимости в распределенных системах // Труды первой Всероссийской научной конференции Методы и средства обработки информации. — М. 2003. — С. 418–423.

16. *Abdulla P.A., Čerāns K., Jonsson B., Yih-Kuen T.* General decidability theorems for infinite-state systems // Proc. 11th

IEEE Symp. Logic in Computer Science (LICS'96). New Brunswick, NJ, USA, July 1996. — P. 313–321.

17. *Agerwala T., Flinn M.* Comments on capabilities, limitaions and "correctness" of Petri nets // Proc. 1st Annual Symposium on Computer Architecture. New York, 1973. — P. 81–86.

18. *Agha G.A., De Cindio F., Rozenberg G. (Eds.)* Concurrent objects oriented programming and Petri nets: Advances in Petri nets. Lecture Notes in Computer Science. — 2001. Vol. 2001.

19. *Araki T., Kasami T.* Some decision problems related to the reachability problem for Petri nets // Theoretical Computer Science. — 1977. Vol. 3(1). — P. 85–104.

20. *Badouel E.* Réseaux de Petri à structures dynamiques, une bibliographie commentée // Modélisation et vérification des processus paralléles (MOVEP'98), Actes de l'école d'été, École Centrale de Nantes, 6-9 Juillet 1998. — P. 201–206.

21. *Baeten J.C.M., Bergstra J.A., Klop J.W.* Decidability of bisimulation equivalence for processes generating context free languages // Lecture Notes in Computer Science. — 1987. Vol. 259.

22. *Battiston E., Chizzoni A., De Cindio F.* Inheritance and Concurrency in CLOWN // Proc. 2nd Workshop on Object-Oriented Programming and Models of Concurrency, Turin, 1995.

23. *Battiston E., Cindio F.D., Mauri G.* OBJSA nets: a class of high-level nets having objects as domains // Lecture Notes in Computer Science. — 1988. Vol. 340. — P. 20–43.

24. *Biberstein O., Buchs B.* An object-oriented specification language based on hierarchical algebraic Petri nets // Proc. IS-CORE Workshop, Amsterdam, September, 1994. — 1994. Tech. Rep. EPFL-DI. — P. 94–76.

25. *Biberstein O., Buchs B., Guelfi N.* Modeling of cooperative editors using COOPN/2 // Proc. Int. Workshop on Object-Oriented Programming and Models of Concurrency, Osaka, Japan, June 1996.

26. *Brown C.* Relating Petri nets to formulae of Linear logic. Technical report. Department of Computer Science, University of Edinburgh, 1989.

27. *Cherkasova L.A., Kotov V.E.* Structured nets // Proc. 6th MFCS. Lecture Notes in Computer Science. — 1981. Vol. 118. — P. 242–251.

28. *Christensen S.* Decidability and decomposition in Process Algebra: PhD thesis. University of Edinburgh, 1993.

29. *Ciardo G.* Petri nets with marking-dependent arc cardinality: Properties and analysis // Proc. 15th Int. Conf. Application and Theory of Petri Nets, Zaragoza, Spain, June 1994. Lecture Notes in Computer Science. — 1994. Vol. 815. — P. 179–198.

30. *Dershowitz N., Jouannaud J.-P.* Rewrite systems // J. van Leeuwen, (ed.) Chapter 6 of the Handbook of Theoretical Computer Science. B: Formal Methods and Semantics. North-Holland, Amsterdam, 1990. — P. 243–320.

31. *Dershowitz N., Manna Z.* Proving termination with multiset orderings // Communications of the ACM. — 1979. Vol. 22(8). — P. 465–476.

32. *Dufourd C., Finkel A., Schnoebelen Ph.* Reset nets between decidability and undecidability // Proc. 25th Int. Coll. Automata, Languages, and Programming (ICALP'98), Lecture Notes in Computer Science. — 1998. Vol. 1443. — P. 103–115.

33. *Farwer B.* A Linear Logic view of object Petri nets // Fundamenta Informaticae. — 1999. Vol. 37, №3. — P. 225–246.

34. *Farwer B.* A multi-region Linear Logic based calculus for dynamic Petri net structure // Fundamenta Informaticae. — 2000. Vol. 43, №1–4. — P. 61–79.

35. *Farwer B.* Relating formalisms for non-object-oriented object Petri nets // Proc. of the Concurrency Specification and Programming (CS&P'2000) Workshop. 9–11 October 2000. Vol. 1. Informatik-Bericht Nr. 140. Humboldt-Universität zu Berlin, Informatik-Berichte, Berlin, 2000. — P. 53–64.

36. *Farwer B.* Linear logic based calculi for object Petri nets: PhD thesis. Logos Verlag, ISBN 3-89722-539-5, Berlin, 2000.

37. *Farwer B., Lomazova I.A.* A systematic approach towards object-based Petri net formalisms// Lecture Notes in Computer Science. — 2002. Vol. 2244. — P. 255–267.

38. *Fehling R.* A concept of hierarchical Petri nets with building blocks // G. Rozenberg (Ed.) Advances in Petri nets. Lecture Notes in Computer Science. — 1993. Vol. 674. — P. 148–168.

39. *Finkel A.* Reduction and covering of infinite reachability trees // Information and Computation. — 1990. Vol. 89(2). — P. 144–179.

40. *Finkel A., Schnoebelen Ph.* Well-structured transition systems everywhere! // Theoretical Computer Science. — 2001. Vol. 256(1–2). — P. 63–92.

41. *Finkel A., Schnoebelen Ph.* Fundamental structures in well-structured infinite transition systems // Proc. 3rd Latin American Theoretical Informatics Symposium (LATIN'98), Campinas, Brazil, Apr. 1998. Lecture Notes in Computer Science. — 1998. Vol. 1380. — P. 102–118.

42. *Genrich H.* Equivalence transformations of PrT-nets // Advances in Petri nets, 1989. Lecture Notes in Computer Science. — 1990. Vol. 424. — P. 179–208.

43. *Genrich H., Lautenbach K.* System modeling with high-level Petri nets // Theoretical Computer Science. — 1981. Vol. 13. — P. 109–136.

44. *Girard J.-Y.* Linear Logic // Theoretical Computer Science. — 1987. Vol. 50. — P. 1–102.

45. *Haddad S., Poitrenand D.* Theoretical aspects of recursive Petri nets // Proc. ATPN'99. Lecture Notes in Computer Science. — 1999. Vol. 1639. — P. 228–247.

46. *Higman G.* Ordering by divisibility in Abstract Algebra // Proc. London Math. Soc. — 1952. Vol. 3(2). — P. 326–336.

47. *Holvoet T.* Agents and Petri Nets // Petri Nets Newsletter. — 1995. Vol. 49. — P. 3–8.

48. *Hong J.E., Bae D.H.* HOONets: Hierarchical Object-Oriented Petri Net for SYstem Modeling and Analysis // TR CS-TR-98-132, http://cs.kaist.ac.kr/library/tr/.

49. *Jantzen M.* Language Theory of Petri Nets // Lecture Notes in Computer Science. — 1987. Vol. 254.

50. *Jensen K.* Coloured Petri nets and the invariant method // Theoretical Computer Science. — 1981. Vol. 14. — P. 317–336.

51. *Jensen K.* Coloured Petri Nets. Vol.1. — EATCS Monographs on TCS, Springer-Verlag, 1994.

52. *Keller R.M.* Vector replacement systems: a formalism for modelling asynchronous systems. Tech. Report, Princeton University, 117, 1972.

53. *Keller R.M.* Formal verification of parallel programs // Communications of the ACM. — 1976. Vol. 1. №7. — P. 371–384.

54. *Kosaraju S.R.* Limitations of Dijkstra's semaphore, primitives and Petri nets // Operating Systems Review. — 1973. Vol. 7. №4. — P. 122–136.

55. *Kozaraju S.R.* Decidability of reachability in vector addition systems // Proc. 14th Annual ACM Symp. on Theory of Computing, San Francisco, 1982. — P. 267–281.

56. *Kruskal J.B.* Well-quasi-ordering, the tree theorem, and Vázsonyi's conjecture // Trans. Amer. Math. Soc. — 1960. Vol. 95. — P. 210–225.

57. *Kwong Y.S.* On the absence of livelocks in parallel programs // Lecture Notes in Computer Science. — 1979. Vol. 70. — P. 179–190.

58. *Lakos C.A.* From Coloured Petri Nets to Object Petri Nets // Proc. Int. Conf. on Appl. and Theory of Petri Nets. Lecture Notes in Computer Science. — 1995. Vol. 935. — P. 278–297.

59. *Lakos C.A.* On the Abstraction of Coloured Petri Nets // Proc. Int. Conf. on Application and Theory of Petri Nets. Lecture Notes in Computer Science. — 1997. Vol. 1248. — P. 42–61.

60. *Lakos C.A.* Composing Abstractions of Coloured Petri Nets // Application and Theory of Petri Nets. — Aarhus, Denmark, 2000. — P. 323–345.

61. *Lamport L., Lynch N.* Distributed Computing: Models and Methods // J. van Leeuwen (ed.) Handbook of Theoretical Computer Science. — Elsevier Science Publ. Co., 1990.

62. *Lomazova I.A.* Multi-Agent Systems and Petri Nets // Proc. Int. Workshop "Distributed Artificial Intelligence and Multi-Agent Systems" DAIMAS'97, St.Petersburg, June 15-18, 1997. — P. 147–152.

63. *Lomazova I.A.* On Proving Large Distributed Systems: Petri Net Modules Verification // Proc. 4th Int. Conference on Parallel Computing Technologies. Lecture Notes in Computer Science. — 1997. Vol. 1277. — P. 70–75.

64. *Lomazova I.A.* Nested Petri nets — a Formalism for Specification and Verification of Multi-Agent Distributed Systems // Fundamenta Informaticae. — 2000. Vol. 43. №1–4. — P. 195–214.

65. *Lomazova I.A.* Nested Petri nets: multi level and recursive systems // Fundamenta Informaticae. — 2001. Vol. 47, №3-4. — P. 283–293.

66. *Lomazova I.A.* Modeling dynamic objects in distributed systems with Nested Petri nets // Fundamenta Informaticae. — 2002. Vol. 51, №1-2. — P. 121–133.

67. *Lomazova I.A., Schnoebelen Ph.* Some Decidability Results for Nested Petri Nets // Proc. Andrei Ershov 3rd Int. Conf. Perspectives of System Informatics (PSI'99), Novosibirsk, Jul. 1999. Lecture Notes in Computer Science. — 2000. Vol. 1755. — P. 207–219.

68. *Manna Z., Pnueli A.* The Temporal Logic of Reactive and Concurrent Systems. — Springer-Verlag, 1992.

69. *Marti-Oliet N., Meseguer J.* From Petri nets to Linear logic. Technical report, SRI International, Computer Science Laboratory, Stanford, 1989.

70. *Mayr E.W.* An Algorithm for the General Petri Net Reachability Problem // SIAM Journal on Computing. — 1984. Vol. 13. — P. 441–460.

71. *Melliès P.-A.* On a duality between Kruskal and Dershowitz theorems // Proc. 25th Int. Colloquium on Automata, Languages and Programming, Aaborg, 1998 (ICALP'98).

72. *Milner R.* A calculus of communicating systems // Lecture Notes in Computer Science. — 1980. Vol. 92.

73. *Milner R.* Communication and concurrency. — Prentice Hall Int., 1989.

74. *Moldt D., Wienberg F.* Multi-Agent Systems Based on Coloured Petri nets // Proc. Int. Conf. on Application and Theory of Petri Nets. Lecture Notes in Computer Science. — 1997. Vol. 1248. — P. 82–101.

75. *Nash-Williams C.St.J.A.* On well-quasi-ordering finite trees // Proc. of the Cambridge Philosophical Society. — 1963. Vol. 59(4). — P. 833–835.

76. *Neuendorf K.-P., Schmidt K., Kiritsis D., Xirouchakis P.* Workflow Modelling and Analysis with Chameleon Nets // Burkhard H.-D., Czaja L. and Starke P., eds. Workshop Concurrency, Specification and Programming, 28–30 September 1998. Humboldt-Universität, Informatik-Berichte №110. Berlin, 1998. — P. 156–161.

77. *Neuendorf K.-P., Kiritsis D., Xirouchakis P.* Chameleon Systems — a Class of Multi-level Petri nets // Philippi S., ed. Workshop Algorithmen und Werkzeuge für Petrinetze, 2–3 Oktober 2000, Koblenz, Germany. Fachberichte Informatik №7. Universität Koblenz-Landau, Institut für Informatik, 2000. — P. 90–95.

78. *Nützel J., Däne B., Fengler W.* Object Nets for the Design and Verification of Distributed and Embedded Applications // Proc. 3rd International Workshop on Embedded High Performance Computing (EHPC'98) at the First Merged Symposium IPPS/SPDP'98, Orlando, March 30 – April 3, 1998.

79. *Park D.S.M.* Concurrency and automata on infinite sequences // Lecture Notes in Computer Science. — 1981. Vol. 104.

80. *Perkusich A., Figueiredo J.C.A.* G-nets: A Petri nets based approach for logical and timing analysis of complex software systems // J. of Systems and Software. — 1997. Vol. 39. — P. 39–59.

81. *Petri C.A.* Kommunikation mit Automaten: PhD thesis, Institute für Instrumentelle Mathematik, Bonn, Germany, 1962.

82. *Plotkin G.* A structural approach to operational semantics // Diami FN-19, Computer Science Department, Aarhus University, 1981.

83. *Reinhardt K.* Reachability in Petri nets with inhibitor arcs. Unpublished manuscript. http://www-fs.informatik.uni-tuenbingen.de/reinhard, 1995.

84. *Reisig W.* Petri nets. — Springer-Verlag, 1985.

85. *Reisig W.* Petri Nets in Software Engineering // Advances in Petri Nets 1986 - Part 2. Lecture Notes in Computer Science. — 1987. Vol. 255. — P. 63–96.

86. *Reisig W.* Petri nets and algebraic specifications // Theoretical Computer Science. — 1991. Vol. 80. — P. 1–34.

87. *Reisig W.* Petri net models of distributed algorithms // Lecture Notes in Computer Science. — 1995. Vol. 1000.

88. *Rozenberg G., Thiagarajan P.S.* Petri nets: basic notions, structure, behavior // Current Trends in Concurrency. Lecture Notes in Computer Science. — 1986. Vol. 224. — P. 585–668.

89. *Sibertin-Blanc C.* Cooperative nets // Proc. 15th Int. Conf. on Application and Theory of Petri Nets. Lecture Notes in Computer Science. — 1994. Vol. 815. — P. 471–490.

90. *Smith E.* Principles of high-level net theory. Lectures on Petri nets // Lecture Notes in Computer Science. — 1998. Vol. 1491. — P. 174–210.

91. *Thiagarajan P.S.* Elementary net systems // Lecture Notes in Computer Science. — 1987. Vol. 254. — P. 26–59.

92. *Valk R.* Petri Nets as Token Objects: An Introduction to Elementary Object Nets // Proc. Int. Conf. on Application and Theory of Petri Nets. Lecture Notes in Computer Science. — 1998. Vol. 1420. — P. 1–25.

93. *Valk R.* Reference and value semantics for object Petri nets // H. Weber, H. Ehrig, and W. Reisig (eds.) Coloquium on Petri Net Technologies for Modelling Communication Based Systems. Frauenhofer Institute for Software and System Engeneering ISST, Berlin, 1999. — P. 169–188.

94. *Wooldridge M.J., Jennings N.R.* Agent Theories, Architectures, and Languages: A Survey // Intelligent Agents. Proc. ECAI-94 Workshop on Agents Theories, Architectures, and Languages. Amsterdam, The Nehterlands, August 8–9, 1994. — Springer-Verlag, 1995.

95. *Zapf M., Heinzl A.* Techniques for integrating Petri nets and object-oriented concepts. Working Papers in Information Systems, №1/1998. University of Bayreuth, 1998.

Научное издание

И. А. Ломазова
ВЛОЖЕННЫЕ СЕТИ ПЕТРИ:
моделирование и анализ распределенных систем
с объектной структурой

«Научный мир». Тел./факс (007)(095) 291-28-47
E-mail: naumir@ben.irex.ru
Internet: http://195.178.196.201/N_M/n_m.html
Лицензия ИД № 03221 от 10.11.2000
Подписано к печати 12.11.2003
Формат $60 \times 90/16$. Печать офсетная. Усл. печ. л 13.
Тираж 500 экз. Заказ
Издание отпечатано в типографии