АКАДЕМИЯ НАУК СССР

ИЗВЕСТИЯ АКАДЕМИИ НАУК СССР

ТЕХНИЧЕСКАЯ КИБЕРНЕТИКА

(ОТДЕЛЬНЫЙ ОТТИСК)

5

МОСКВА · 1983

ТЕХНИЧЕСКАЯ
КИБЕРНЕТИКА
№ 5 · 1983

УДК 681.142

СЕТИ ПЕТРИ

РОЗЕНБЛЮМ Л. Я.

Введение. Среди множества моделей, ориентированных на решение разнообразных задач моделирования, анализа и синтеза процессов различной природы (в том числе вычислительных процессов и процессов управления) последнее время большим вниманием пользуются так называемые сети Петри (ударение на первом слоге). Исследователей привлекают такие достоинства этой модели, как возможность отражения асинхронности и параллелизма, недетерминированности процессов, динамики их функционирования, простой синтаксис и наглядность модели в сочетании с достаточно широкими функциональными возможностями.

Можно констатировать, что теория сетей Петри (СП) по существу превратилась в самостоятельную научную дисциплину, а области ее применения непрерывно расширяются. Развитие некоторых направлений теории СП мотивируется внутренними ее потребностями, но основным стимулом являются запросы практики и, в первую очередь, практики построения вычислительных, информационных и управляющих систем и комплексов.

Теория СП во многих своих аспектах теснейшим образом связана с теорией автоматов и по существу является развитием последней. Это обстоятельство позволяет с единых методологических позиций рассматривать вопросы программного обеспечения, аппаратной поддержки и информационного обмена при проектировании вычислительных систем. Эти вопросы будут центральными в обзоре, хотя область возможного применения аппарата теории СП этой проблематикой не исчерпывается.

Поскольку целью настоящего обзора является ознакомление с СП достаточно широкого круга читателей, изложение по возможности ведется на содержательном, а не формальном уровне. При желании заинтересованные лица могут обратиться к литературе, список которой приведен в конце обзора, где формализация присутствует в избытке.

1. Описание модели. Сеть Петри (СП) формально определяется как четверка вида

$$S=\langle P, T, E, \mu^0\rangle, \tag{1.1}$$

где P — конечное множество *позиций* (иначе — мест или условий); T — конечное множество *переходов* (событий); E — конечное множество дуг, $E\subseteq(P\times T)\cup(T\times P)$; μ^0 — *начальная маркировка* (разметка), μ^0: $P\to N$, $N=\{0, 1, 2,\ldots\}$ — множество натуральных чисел.

Графическим изображением СП является двудольный ориентированный граф с двумя типами вершин: вершины p_i из множества P изображают обычно кружками, а вершины $t_j\in T$ — полочками. Дуги графа могут быть направлены от кружков только к полочкам, а от полочек — только к кружкам, так что любая позиция может быть входной или (и) выходной позицией одного или нескольких переходов.

Множества входных и выходных позиций перехода $t_j\in T$ будем обозначать через $I(t_j)$ и $O(t_j)$ соответственно. Аналогично, записи $O(p_i)$ и $I(p_i)$ суть обозначения множеств переходов, являющихся соответственно выходом и входом данной позиции.

Очевидно, (1.1) может быть задано в виде

$$S=\langle P,\ T,\ I,\ O,\ \mu^0\rangle, \tag{1.2}$$

если обозначить через I и O объединение $I(t_j)$ и $O(t_j')$ по всем переходам $t_j \in T$, т. е. $I=\bigcup\limits_T I(t_j)$, $O=\bigcup\limits_T O(t_j)$, $P=I \cup O$.

Представление СП в виде двудольного графа позволяет задать структуру СП статически. Динамика в модель вносится механизмом смены маркировки (разметки) позиций и соглашением о правиле срабатывания (реализации) переходов.

Начальная маркировка, как она была определена выше, приписывает некоторым позициям СП некоторые целые числа (в том числе нуль). На графе маркировка задается так: в соответствующих кружках рисуют определенное число (жирных) точек, носящих название маркеров (фишек).

Передвижение маркеров по СП осуществляется посредством срабатывания ее переходов. Сработать может только возбужденный (возможный) переход, т. е. такой переход $t_k \in T$, во всех входных позициях $I(t_k)$ которого содержится хотя бы по одному маркеру (для каждого такого перехода и для каждой $p_s \in I(t_k)$ имеет место $\mu_s \geqslant 1$, где через $\mu_s=\mu(p_s)$ обозначено число маркеров в позиции p_s). Срабатывание перехода может произойти через любой конечный промежуток времени после возбуждения перехода. Результатом срабатывания является изъятие из всех входных позиций сработавшего перехода по одному маркеру и добавление во все его выходные позиции по одному маркеру. Срабатывание перехода считается неделимым актом, т. е. предполагается, что изъятие маркеров из всех входных позиций и их перемещение во все выходные позиции осуществляется мгновенно, с нулевой задержкой. В некоторых приложениях, однако, соглашение о неделимости может быть нарушено.

Соглашение о правиле срабатывания переходов отражает концепцию причинно-следственной связи: после того, как известные нам объективные причины (условия) какого-либо события созрели, это событие может наступить. Однако время его наступления заранее неизвестно, поскольку возможно существование других, неизвестных нам или несущественных для нас причин, которые в конечном счете неизбежно вместе с существенными обусловят наступление интересующего нас события. После наступления события возникнут новые условия — следствия, могущие стать причинами последующих событий. При этом первоначальные причины могут либо исчезать, либо сохраняться. Последнее возможно как за счет того, что ресурс маркеров полностью не исчерпан, так и за счет регенерации маркеров (когда позиции являются одновременно выходными и входными).

Срабатывание какого-либо возбужденного перехода в общем случае вызывает смену маркировки СП. При этом может измениться только маркировка входных и выходных позиций возбужденного перехода. Маркировка позиций, являющихся одновременно входными и выходными для данного перехода, после срабатывания его не изменяется.

Текущая маркировка СП может пониматься как состояние СП в данный момент времени. Срабатывание возбужденного перехода, являющееся локальным актом, в общем случае ведет к изменению маркировки сети, а, следовательно, ее глобального (полного) состояния.

Сеть Петри функционирует, переходя от одной маркировки к другой в результате срабатывания возбужденных переходов. Таким образом, можно считать, что динамика поведения СП (ее функционирования во времени) может быть адекватно описана тройкой

$$S=\langle \mu^0,\ \rightarrow,\ M\rangle, \tag{1.3}$$

где μ^0 — начальная маркировка; $\rightarrow$ — отношение непосредственного следования маркировок (запись $\mu^a \rightarrow \mu^b$ означает, что за маркировкой μ^a непосредственно следует маркировка μ^b; M — множество допустимых маркировок (состояний) данной СП, достижимых из μ^0.

Маркировка может изображаться вектором $\mu=(\mu_1, \mu_2 \ldots \mu_n)$, число компонент которого равно числу позиций СП, а значение i-й компоненты, $1 \leqslant i \leqslant n$, есть натуральное число $\mu_i=\mu(p_i)$ — число маркеров в позиции.

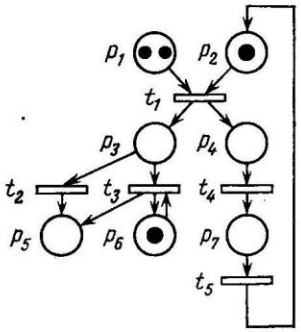


Рис. 1. Пример сети Петри

Представленная на рис. 1 СП с семью позициями $P=\{p_1, p_2 \ldots p_7\}$ и пятью переходами $T=\{t_1, t_2, \ldots, t_5\}$ может быть описана следующим образом: $I(t_1)=\{p_1, p_2\}$, $O(t_1)=\{p_3, p_4\}$, $I(t_2)=\{p_3\}$, $O(t_2)=\{p_5\}$, $I(t_3)=\{p_3, p_6\}$, $O(t_3)=\{p_5, p_6\}$, $I(t_4)=\{p_4\}$, $O(t_4)=\{p_7\}$, $I(t_5)=\{p_7\}$, $O(t_5)=\{p_2\}$.

В векторе ее начальной маркировки $\mu^0=(\mu_1^0, \mu_2^0, \ldots \mu_7^0)$ только три ненулевые компоненты: $\mu_1^0=2$, $\mu_2^0=\mu_6^0=1$. При начальной маркировке имеется лишь один возбужденный переход — t_1. После его срабатывания изымается по одному маркеру из входных позиций p_1 и p_2, и в его выходные позиции p_3 и p_4 помещается по одному маркеру. Таким образом, из начальной маркировки $\mu^0=(2100010)$ непосредственно следует маркировка $\mu^1=(1011010)$: $\mu^0 \rightarrow \mu^1$. Здесь возбуждены переходы t_4, t_2 и t_3. Из двух последних может сработать только один, т. к. срабатывание любого из них снимает возбуждение другого. Время срабатывания возбужденного перехода в модели не оговаривается: оно считается произвольным, хотя и конечным. При срабатывании t_2 или t_3 в данном случае будет достигнута одна и та же маркировка $\mu^2=(1001110)$, при срабатывании t_4 — $\mu^3=(1010011)$. Возможно одновременное срабатывание t_2 и t_4, а t_3 и t_4. В этом случае состояние СП будет определяться маркировкой $\mu^4=(1000111)$.

(Некоторые авторы и, в частности, автор обзора [180] *, отвергают возможность одновременного срабатывания двух и более возбужденных переходов, ссылаясь на малую вероятность такого события. Однако такая точка зрения, будучи удобной в некоторых отношениях, противоречит концепции независимости поведения СП от времен срабатываний переходов).

Для задания динамики поведения СП обычно используют так называемые *диаграммы достижимых состояний* (*маркировок*), далее для краткости именуемые просто *диаграммами*. Диаграмма представляет собой орграф, вершины которого — достижимые маркировки из множества M, а дуги направлены из μ^a в μ^b, μ^a, $\mu^b \in M$, если имеет место $\mu^a \rightarrow \mu^b$ ($\mu^a \rightarrow \mu^a$ означает наличие петли). Дуги, если это необходимо, помечают обозначениями тех переходов, срабатывание которых является причиной смены маркировки.

Диаграмма рис. 2 соответствует СП, изображенной на рис. 1. Пометки вида $(t_2 \vee t_3)t_4$ в данном случае означают, что одновременно срабатывает одна из пар (t_2, t_4) или (t_3, t_4) переходов. Диаграмма отражает возможный параллелизм процесса, представимого СП. Множество допустимых маркировок (состояний) в данном случае состоит из 16 маркировок, причем из выделенной начальной маркировки μ^0 достижима лишь одна результирующая (тупиковая) маркировка $\mu^r=(0100210)$, из которой СП никогда не выходит, что гарантирует однозначное завершение процесса. По диаграмме, в частности рис. 2, можно определить множество последовательностей срабатываний переходов (траекторий процесса). Одна из возможных траекторий такова: $t_1t_3t_4t_5t_1t_4t_5t_2$. Между тем детереминизм по-

* Для удобства читателей список дан в алфавитном порядке.

ведения СП рис. 1 только «внешний»: возможно несколько реализаций процесса.

В общем случае одной начальной маркировке могут соответствовать несколько результирующих маркировок. Достижение каждой из них возможно при определенных распределениях скоростей срабатываний переходов. Именно *асинхронность* процессов, отражаемая в модели отсутствием каких-либо указаний о времени срабатывания переходов (предполагается лишь, что они конечны), гарантирует *недетерминизм* модели. Этот недетерминизм может проявляться двояко: в общем случае неизвестно, по какой траектории будет идти развитие процесса (как он будет реализован) и в каком заключительном состоянии он окажется в финале. Недетерминизм является весьма полезным свойством модели, поскольку он может отражать наше незнание ненужных деталей рассматриваемого процесса и, более того, позволяет игнорировать их.

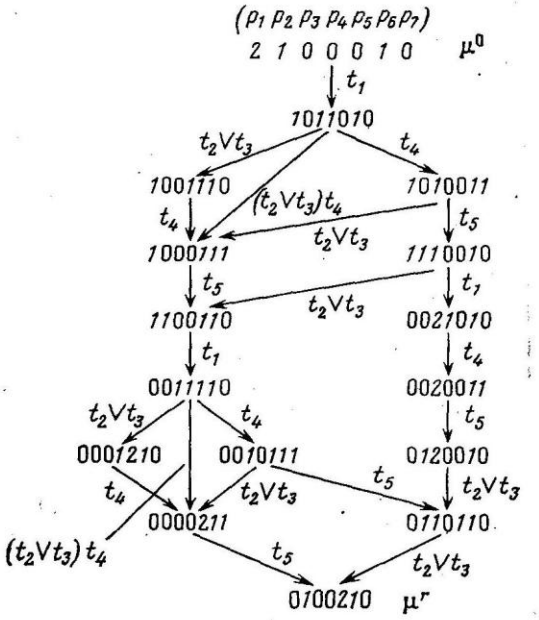


Рис. 2. Диаграмма достижимых состояний для сети рис. 1

Возможность синхронизации и *параллелизм* также являются неотъемлемыми особенностями СП. Так, уже из рис. 1 видно, что только при синхронизации условий p_1 и p_2 (наличие хотя бы одного маркера как в позиции p_1, так и в позиции p_2) возможно срабатывание перехода t_1, а далее события из пар (t_2, t_4) или (t_3, t_4) могут произойти одновременно (параллельность). Заметим, что на диаграмме синхронизм проявляется в фрагментах типа «гамаков», пример которых демонстрирует четверка маркировок (0011110), (0001210), (0010111), (0000211) в левом нижнем углу рис. 2, а параллельность — в фрагментах, замыкаемых дугами с пометками типа $(t_2 \vee t_3)t_4$.

Наличие указанных принципиальных свойств модели порождает широкое разнообразие видов поведения (множеств типов диаграмм). Задача типизации СП связана со следующими эффектами их поведения. Поскольку в некоторых позициях сетей возможно накопление бесконечного числа маркеров, то множество допустимых состояний может не быть конечным. В СП может и не существовать тупиковых маркировок; такого рода сетями моделируются циклические или автономные процессы. Наконец, в цикл могут оказаться втянутыми состояния только из некоторого подмножества допустимых состояний. Сказанное выше вызывает необходимость проведения классификации СП.

2. Классификация сетей Петри. Ясно, что различные СП могут отличаться друг от друга не только структурами графов, но также при одинаковых графах диаграммами и множествами допустимых состояний. Таким образом, исследование модели, какой являются СП, связано с изучением их статических и динамических свойств. Классификация СП базируется на качественных и количественных ограничениях, накладываемых на допустимые конфигурации (статические ограничения) и возможные типы маркирования (динамические ограничения).

Дадим определения некоторых представляющих особый интерес классов СП, порождаемых динамическими ограничениями.

Сеть Петри называется:

— *k-ограниченной* ($k \geqslant 1$ — целое число), если на множестве ее достижимых состояний не найдется ни одной позиции $p_i \in P$, для которой $\mu(p_i) > k$ (в которой при функционировании СП появилось бы более k маркеров);

— *безопасной*, если она 1-ограничена (ни в одной ее позиции не может появиться более одного маркера);

— *ограниченной*, если найдется такое целое k, для которого она k-ограничена;

— *1-консервативной*, если в процессе функционирования СП общее число маркеров в ней остается постоянным, т. е. для любого $t_r \in T$ имеет место

$$\sum_{p_i \in I(t_r)} \mu(p_i) = \sum_{p_j \in O(t_r)} \mu(p_j);$$

— *консервативной*, если существует положительная целочисленная функция f: $P \to N^+$ такая, что для любого перехода $t_r \in T$ имеет место $\sum_{p_i \in I(t_r)} f(p_i) = \sum_{p_j \in O(t_r)} f(p_j)$ (иначе: СП консервативна, если для любого перехода $t_r \in T$ существуют такие положительные целые коэффициенты a_i и a_j, что $\sum_{p_i \in I(t_r)} a_i \mu(p_i) = \sum_{p_j \in O(t_r)} a_j \mu(p_j)$; таким образом, консервативные СП 1-консервативны, если $a_i = 1$ для всех $p_i \in P$ [114]);

— *живой*, если каждый переход $t_j \in T$ является потенциально срабатывающим при любой маркировке из M (другие определения живой СП рассматривались в [51, 122]);

— *устойчивой*, если для всех t_i, $t_j \in T$, $t_i \neq t_j$ и любой допустимой маркировке, при которой t_i и t_j возбуждены, срабатывание одного из них не может снять возбуждение другого. Перечислим некоторые важные в приложениях классы СП, порождаемые статическими ограничениями, накладываемыми на СП. Через $|A|$ обозначим мощность множества A.

Сеть Петри называется:

— *сетью свободного выбора*, если для любых $t_j \in T$ и $p_j \in I(t_j)$ позиция p_i является либо единственной входной позицией перехода t_j, т. е. $|O(p_i)| = 1$, либо этот переход имеет единственную входную позицию, т. е. $|I(t_j)| = 1$ (иначе: если два перехода имеют общую входную позицию, то эта позиция единственна для каждого перехода);

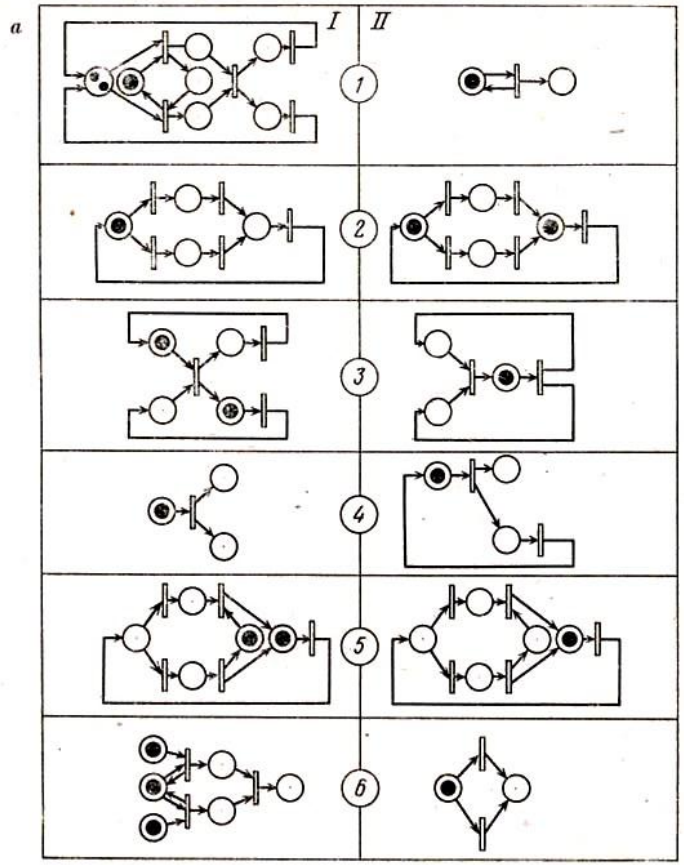
— *простой*, если любая пара переходов t_j, $t_k \in T$ имеет не более одной общей входной позиции $p_i \in P$, т. е. если $|I(t_j) \cap I(t_k)| \leqslant 1$;

— *маркированным графом*, если каждая позиция имеет в точности по одному входному и выходному переходу, т. е. если $|O(p_i)| = |I(p_i)| = 1$;

— *автоматной*, если каждый переход t_j имеет не более одного входа и не более одного выхода, т. е. если $|O(t_j)| = |I(t_j)| = 1$;

— *бесконфликтной*, если либо для каждой ее позиции $p_i \in P$ существует не более одной исходящей дуги ($|O(p_i)| \leqslant 1$, либо для всех $t_j \in O(p_i)$ выполняется $t_j \in I(p_i)$ (любая позиция, являющаяся входной для более чем одного перехода, является одновременно и выходной для каждого такого перехода). Бесконфликтные СП устойчивы, хотя обратное справедливо не всегда.

Основы классификации СП, очевидно, были заложены в работе [88] и почти без изменений использовались в большинстве исследований по СП. Попытка более детальной классификации сделана в [110]; интересно, что она изображается с помощью специально построенной СП. Рис. 3 иллюстрирует общепринятую классификацию.



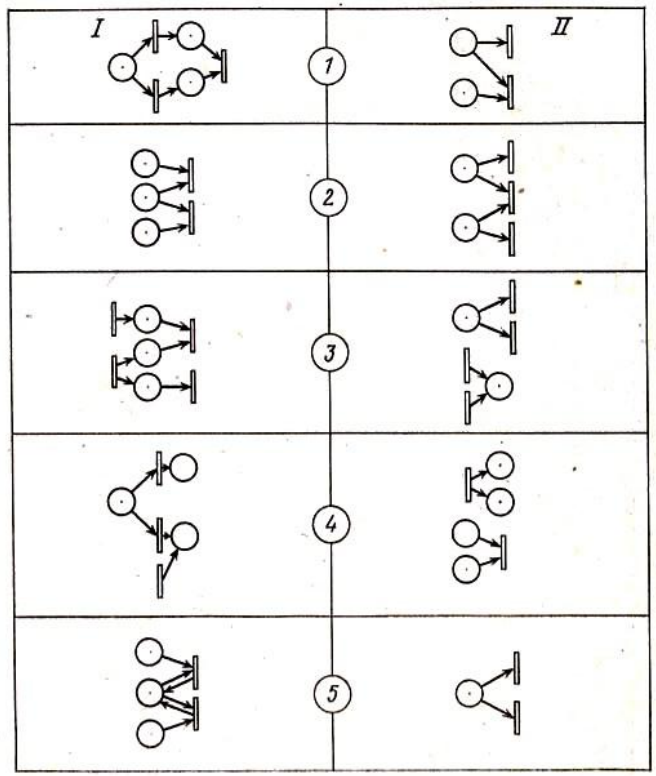


Рис. 3. К классификации сетей Петри: *а* – примеры (I) и контрпримеры (II) сетей следующих типов: ограниченные (1), безопасные (2), 1-консервативные (3), консервативные (4), живые (5), устойчивые (6); *б* – допустимые фрагменты (I) и недопустимые фрагменты (II) сетей следующих типов: свободного выбора (1), простые (2), маркированные графы (3), автоматные (4), бесконфликтные (5)

Маркированные графы изучались в работах [52, 97—99, 104, 105, 116, 117, 162], сети со свободным выбором и простые СП — в [88], бесконфликтные и устойчивые СП — в [131], безопасные — в [219]. С вопросами классификации тесно связано изучение свойств СП, предпринятое в работах [21, 51, 58, 80, 88, 100, 131, 135, 136, 210, 215].

3. Сети Петри как аппарат моделирования. СП широко используются как средство моделирования динамического поведения параллельных процессов самой различной природы: за подтверждениями можно обратиться, например, к [149, 171]. Объекты исследования могут изучаться посредством СП с разными целями (классификация, выявление различных свойств, реализация и т. п.) на разных уровнях детализации.

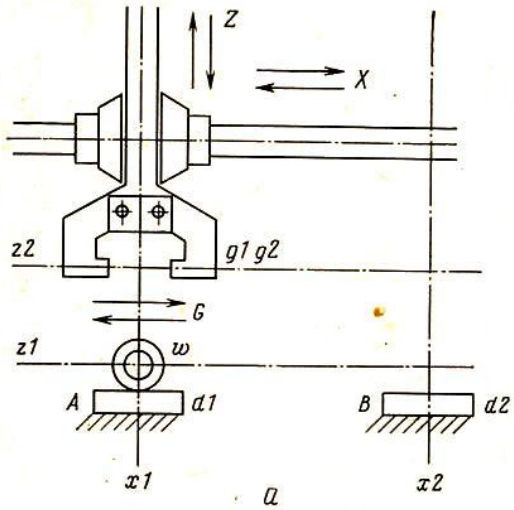
Построение описания моделируемой системы является неформальным актом, выполняемым на этапе формулировки технического задания в режиме диалога проектировщика с заказчиком. Этот акт среди всего прочего требует, чтобы некоторым происходящим в реальной или проектируемой системе событиям и условиям их наступления были поставлены в соответствие понятия СП, т. е. чтобы моделирующая СП была интерпретирована. В *интерпретированной СП* все или некоторые переходы и (или) позиции могут быть помечены некоторыми символами, удобными для представления семантики функционирования системы, а наличие маркеров в некоторых позициях обычно связывают с истинностью некоторых предикатов. Не останавливаясь на деталях, проиллюстрируем этап построения интерпретированных СП следующим примером, являющимся типичным для промышленной автоматики.

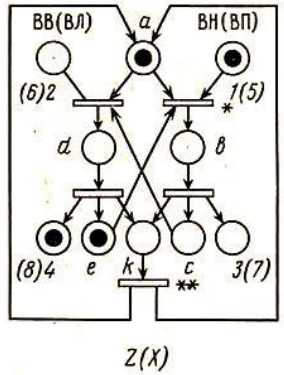
Пример. На рис. 4, *а* схематично показан манипулятор, предназначенный для транспортировки груза (детали) w с площадки A на площадку B. Рука манипулятора с двумя пальцами имеет три степени свободы: она может перемещаться вправо-влево и вниз-вверх при изменении направления вращения не показанных на рисунке двигателей горизонтального перемещения X и вертикального перемещения Z, а ее пальцы — сжиматься или разжиматься посредством двигателя G. Положение руки фиксируется по вертикали и горизонтали замыканием концевых переключателей $z1$, $z2$ и $x1$, $x2$ соответственно. При захвате детали пальцами срабатывает контакт $g2$, при полном разжиме пальцев — контакт $g1$. При наличии груза на площадке A или B замыкается контакт $d1$ и $d2$ соответственно. Транспортировка груза возможна, если $d1$ замкнут (на площадке A установлен груз), а $d2$ — разомкнут (площадка B свободна). Начальному положению руки соответствуют координаты $x1$, $z2$; пальцы разжаты ($g1$).

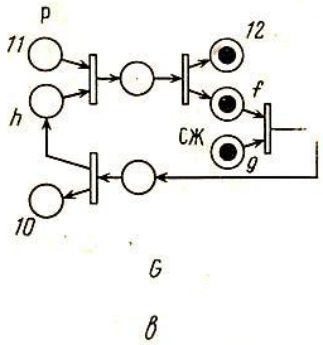
Ясно, что двигатель Z должен останавливаться в положениях $z2$ и $z1$ и после выполнения условий пуска вниз и вверх работать до новой остановки. Функционирование двигателя может быть описано с помощью различных СП; один из возможных вариантов представлен на рис. 4, *б* (см. также [230]).

Наличие маркеров в позициях СП имеет следующую интерпретацию: 1 (2) — выполнены условия запуска в сторону, соответствующую перемещению руки вниз — ВН (вверх — ВВ); 3, c — замкнут концевик $z1$; 4, e — замкнут концевик $z2$; k — замкнут $z1$ или $z2$; a — двигатель остановлен; $b(d)$ — двигатель работает, рука движется вниз (вверх), находясь в промежуточном положении между концевиками $z1$ и $z2$.

Пусть исходно маркеры находятся в позициях a, e и 4, а двигатель остановлен в положении $z2$. Включение в работу (режим ВН) может произойти только после появления маркера в позиции 1. Тогда после срабатывания перехода, отмеченного одной звездочкой, маркер появится в позиции b и будет изъят из его входных позиций. После замыкания концевика $z1$ маркеры появятся в позициях k, c и 3, а затем, после срабатывания помеченного двумя звездочками перехода, маркер из позиции k переместится в позицию a. В конце режима ВН (вниз) маркеры находятся в







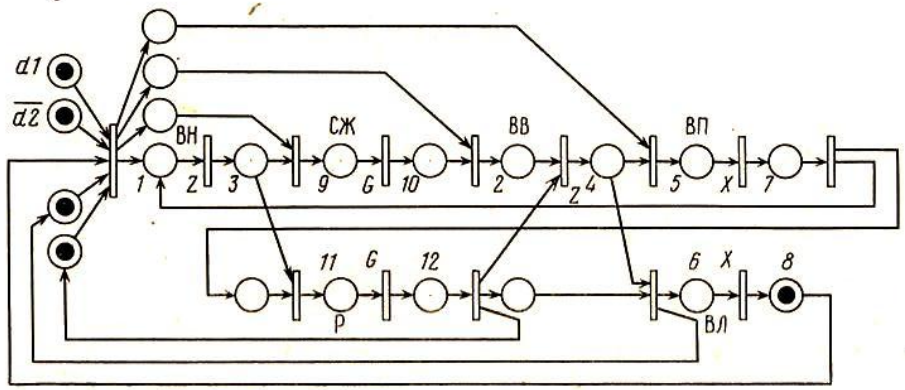


Рис. 4. Пример интерпретированной сети Петри: *а* – манипулятор; *б* – управление двигателями вертикальной и горизонтальной подачи руки; *в* – управление двигателем сжимания и разжимания пальцев руки; *г* – общая модель процесса управления манипулятором

позициях a, c, 3. Эта маркировка является исходной для ВВ (вверх), который выполняется аналогично. Появление маркеров одновременно в позициях 1 и 2 исключается в силу того, что возможен лишь один из этих режимов. Заметим, что позиции c, k и 3 (e, k и 4) по смыслу эквивалентны, но если поименованные буквами позиции нужны для организации собственно процессов управления двигателем Z, то отмеченные числами — для организации процесса управления манипулятором в целом.

Работа двигателя X также моделируется рис. 4, *б*; будем иметь в виду, что позиции для этого процесса и названия режимов обозначены числами и словами, заключенными в скобках.

Работа двигателя G описывается проще (рис. 4, *в*), поскольку он должен останавливаться в одном из рабочих состояний — «пальцы сжаты» (позиции h и 10) и «пальцы разжаты» (позиции f и 12). Маркеры в позициях (сжать — СЖ) 9 и 11 (разжать — Р) суть команды на выполнение одного из этих режимов. Два непомеченных условия соответствуют нахождению пальцев в промежуточных состояниях. Исходная маркировка, показанная на рис. 4, *в*, переходит в 10, h, а последняя при наличии маркера в позиции 11 — вновь в 12, f.

Управление манипулятором в целом изображается СП, представленной на рис. 4, *г*. На нем все числовые обозначения позиций соответствуют приведенным на рис. 4, *б* и *в*, а переходы, поименованные большими латинскими буквами Z, X и G — процессам, изображенным на тех же рисунках. Так, входная позиция 1 и выходная позиция 3 перехода Z — это позиции 1 (ВН) и 3 рис. 4, *б*.

В начальном состоянии маркерами отмечены позиции 4, 7 и 12 (рука в исходном положении), а также условия $d1$ (деталь на месте A) и $d2$ (место B пусто). Из исходного положения работа манипулятора осуществляется по циклу «вниз за деталью (ВН) — сжать деталь (СЖ) — вверх с деталью (ВВ) — вправо с деталью (ВП) — вниз с деталью (ВН) — разжать деталь (Р) — вверх без детали (ВВ) — влево без детали (ВЛ) — вниз за деталью (ВН) и т. д. Запуск второго цикла происходит в том случае, если на месте A появится новая деталь, а с места B перенесенная ранее деталь убрана».

Рис. 4, *г* демонстрирует возможность задания СП по иерархическому принципу. Такое задание осуществляется за счет соответствующей интерпретации позиций и переходов. Любой переход с инцидентными ему входной и выходной позициями может быть интерпретирован как некоторый подпроцесс, поэтому в модели, отражающей взаимодействие подпроцессов, они могут быть изображены примитивно. При необходимости может быть получена детализация подпроцессов и путем композиции построена общая модель. При желании можно было бы объединить рис. 4, *б*, *в* и *г*, но это привело бы к утере наглядности.

Вопросы упрощения описания функционирования различных объектов, выполненного на языке СП, рассматривались в [36—38, 60, 214], декомпозиции СП — в [222], модульного и иерархического представления описаний — в [71, 217, 223, 224].

Моделирование параллельных вычислений. Прежде, чем формально определить параллельную программу, условимся через $S^{\text{п}}=\langle P, T, E, \mu^0\rangle$ обозначать класс СП, обладающий следующими свойствами: свободного выбора; живости; безопасности; наличия двух особых позиций (входной и выходной), не имеющих соответственно входов и выходов; соответствия μ^0 наличию маркера во входной позиции сети.

Тогда параллельная программа есть двойка $\Pi=\langle S^{\text{п}}, \psi\rangle$, где ψ — семантическая интерпретация $S^{\text{п}}$, порождающая П. Интерпретация ψ, в свою очередь, есть четверка $\psi=\langle A, \omega, R, \gamma\rangle$, где $A=\{a_1, \ldots, a_n\}$ — множество операторов, включающее пустой оператор λ; ω — функция распределения операторов, ω: $T\to A$; $R=\{r_1, \ldots, r_m\}$ — множество предикатов; γ — функция распределения предикатов из R по множеству позиций $P'\subset P$, обла-

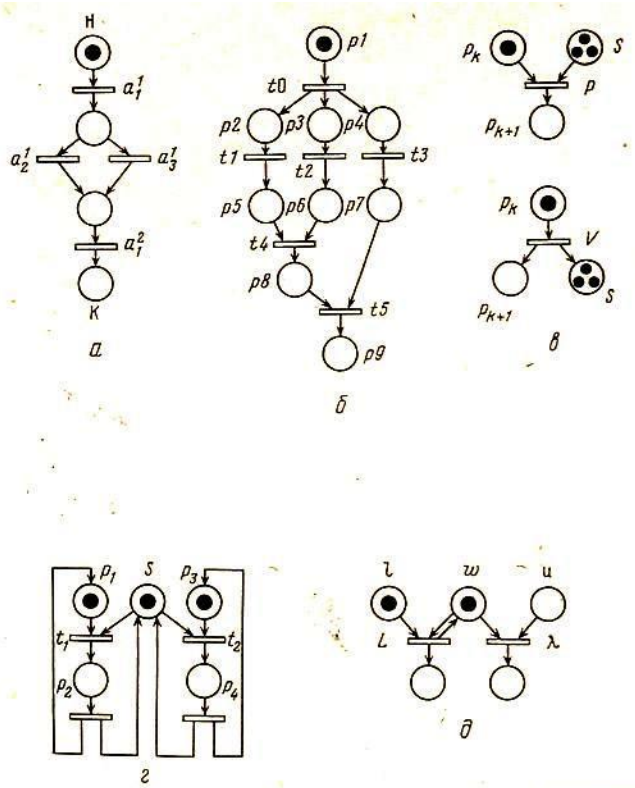


Рис. 5. Примеры моделирования программного обеспечения: *а* – тривиальная параллельная программа (H – начало, K – конец); *б* – процесс вычисления арифметического выражения; *в* – синхропримитивы над семаформами; *г* – задача взаимного исключения; *д* – синхропримитивы типа lock-unlock

дающих свойством свободного выбора, т. е. таких $p \in P'$, которые являются единственной входной позицией нескольких переходов $t_i, t_j, \ldots, t_s$, $i \neq j \neq \ldots \neq s$. Модификация этого определения предлагалась в [115].

Пример тривиальной параллельной программы приведен на рис. 5, *а*. Разным вхождением каждого оператора и предиката в программу присваиваются порядковые номера (верхние индексы).

Посредством СП легко описываются программные конструкции (примитивы), широко используемые для представления параллелизма. Одной из таких конструкций являются примитивы **fork**, **join**, **quit** [65]. По команде **fork** w процесс разветвляется, начиная с команды, помеченной символом w; после разветвления процесс продолжается в обеих ветвях. В каждой ветви процесс заканчивается по команде **quit**. Команда **join** x, w вызывает следующее действие:

$$x := x - 1;$$

if $x = 0$ **then go to** w;.

Вся эта область рассматривается как неделимое действие.

Программный сегмент для вычисления арифметического выражения $(a+b) \times (c+d) - (e/f)$ имеет вид [206]

$n: = 2;$
fork $r3$;
$m: = 2;$

fork $r2$;
$t1$: $=a+b$; **join** m, $r4$; **quit**;
$r2$: $t2$: $=c+d$; **join** m, $r4$; **quit**;
$r4$: $t4$: $=t1\times t2$; **join** n, $r5$; **quit**;
$r3$: $t3$: $=e/f$; **join** n, $r5$; **quit**;
$r5$: $t5$: $=t4-t3$;.

Соответствующая этому фрагменту СП изображена на рис. 5, *б*. Срабатывание переходов $t1-t5$ означает выполнение одноименных операторов программы. Появление маркеров в позициях $p1-p9$ имеет следующую семантику: $p1$ — начать процесс; $p2$, $p3$, $p4$ — начать вычисление оператора $t1$, $t2$, $t3$ соответственно; $p5$, $p6$, $p7$, $p8$ — вычисление $t1$, $t2$, $t3$, $t4$ завершено; $p9$ — вычисление $t5$ (всей формулы) завершено, процесс закончен. Легко видеть, что переход $t0$ моделирует примитив **fork**, переходы $t4$ и $t5$ — примитив **join** (направление перехода w указывает дуга, исходящая из этого перехода), а примитив **quit** моделируется позициями $p5-p9$, появление маркеров в которых фиксирует завершение стадий вычисления формулы.

Из примера видно, как легко трактуется в терминах СП достаточно сложное для понимания неделимости действия (это просто требование одновременности перемещения маркеров из входных позиций в выходные позиции при срабатывании перехода) и как естественно представляется динамика вычислений.

Остановимся на введенных в [67] синхропримитивах над семафорами. Операции над семафором S определяются следующим образом. Операция $V(S)$ увеличивает переменную S на единицу (инкремент) одним неделимым действием; во время ее выполнения к S нет доступа другим процессам. Операция $P(S)$ уменьшает S на 1, если это возможно (семафор должен быть неотрицательным). Если $S=0$, процесс, вызывающий P-операцию, ждет, пока уменьшение S не станет возможным. Эта операция также неделима. Обращение к P и V-операциям одновременно невозможно. Модель этих операций над семафором S приведена на рис. 5, *в*. В позиции S $q\geqslant 1$ маркеров. Тогда при появлении маркера в p_k после выполнения p-операции число маркеров в позициях p_{k+1} и S становится равным соответственно 1 и $q-1$, после выполнения V-операции — 1 и $q+1$.

Решение известной задачи взаимного исключения [67] при помощи указанных синхропримитивов представлено на рис. 5, *г*. Заметим, что поскольку в позиции S имеется лишь один маркер, одновременное появление маркеров в позициях p_2 и p_4 исключается.

Еще одна пара синхропримитивов — это **lock** w, **unlock** w [65], где w — произвольная однобитовая переменная (индикатор) блокировки. Эти примитивы определяются так:

lock w: L: **if** $w=1$ **then go to** L;
else w: $=1$
unlock w: w: $=0$;.

Реализация этой пары показана на СП рис. 5, *д*; λ — пустой оператор. По команде **lock** появляется маркер в соответствующей позиции l, после чего может быть выполнен оператор L; при срабатывании перехода маркер в позиции w сохраняется. По команде **unlock** появляется маркер в соответствующей позиции u и выполняется, например, пустой оператор, а маркер из позиции w изымается.

Аналогично может быть проведено моделирование с помощью СП других синхропримитивов [13, 46, 174].

Для параллельных систем характерна часто встречающаяся ситуация взаимоблокировки (доделка), когда независимые процессы блокируют

друг друга при стремлении захватить один и тот же ресурс. Это явление стало в последнее время предметом пристального изучения [35, 46, 49, 50, 62, 103, 83, 84, 118, 127, 130, 135, 174, 176, 202, 216].

В СП *дедлоком* принято называть множество позиций P' некоторого ее фрагмента, удовлетворяющих условию $I(P') \subseteq O(P')$, где $I(P') = \bigcup_{p_j \in P'} I(p_j)$ и $O(P') = \bigcup_{p_j \in P'} O(p_j)$. Если в процессе функционирования СП все позиции дедлока опустеют (для каждой $p_j \in P'$ число $\mu(p_j) = 0$), то в дальнейшем ни в одной из этих позиций не сможет появиться маркер и, следовательно, не сможет сработать ни один переход, имеющий позицию из P' в качестве входной.

Нежелательные явления в параллельных вычислительных системах исследовались в [130]. СП-модели в теории программирования рассматривались в работах [84, 87, 145, 153, 218]. Семантике параллельных вычислений посвящен ряд работ, опубликованных в [202].

Моделированию асинхронных вычислительных машин и систем с помощью аппарата СП посвящены работы [22, 32, 35, 48, 49, 50, 50, 66, 165, 193, 196, 197, 200, 201], вопросам анализа и верификации параллельных программ — работы [34, 36, 53, 100, 108, 130, 131, 162, 224, 225].

Моделирование и синтез аппаратных средств. Задача аппаратной реализации СП теснейшим образом связана с задачами, возникающими при синтезе асинхронных схем, заданных *моделью Маллера* [158, 161, см. также 2, 14]. Покажем эту связь на примере. На рис. 6, *а* изображена СП, моделирующая работу счетного триггера. Действительно, последовательность переходов в ней имеет вид $t_1 t_3 t_2 t_3 t_1 t_3 \ldots$, т. е. переход t_3 срабатывает вдвое чаще остальных. На рис. 6, *б* приведена так называемая *диаграмма переходов* этой схемы (диаграмма Маллера). Диаграмма переходов задается на множестве векторов с числом компонентов, равным числу переходов СП. Компоненты, соответствующие возбужденным переходам, помечаются звездочками. Диаграммы переходов безопасных СП содержат, естественно, лишь двоичные векторы. Если СП является устойчивой и безопасной, то диаграмме переходов объекта, описываемого этой СП, можно поставить в соответствие систему булевых уравнений. Так, СП рис. 6, *а* соответствует диаграмма переходов рис. 6, *б*, а ей, в свою очередь,— система

$$\left.\begin{aligned} z_1 &= z_1 z_3 \vee \bar{z}_2 \bar{z}_3, \\ z_2 &= z_1 z_3 \vee z_2 \bar{z}_3, \\ z_3 &= z_1 \bar{z}_2 \vee \bar{z}_1 z_2. \end{aligned}\right\} \qquad (3.1)$$

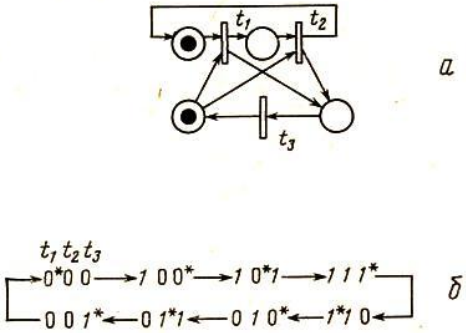


Рис. 6. Пример моделирования поведения асинхронных схем: *а* — сеть Петри счетчика; *б* — диаграмма переходов счетчика

Однако непосредственная реализация асинхронной схемы по этой системе не приводит к желаемому результату из-за возможных состязаний. Стандартным приемом синтеза свободной от состязаний схемы, к тому же инвариантной к разбросу задержек элементов (соединительные линии полагают имеющими нулевую задержку), является построение на доформальном этапе синтеза так называемых *полумодулярных* (в частности, *дистрибутивных* и *последовательных*) диаграмм переходов [2, 14, 158, 161]. От этих диаграмм можно перейти к сокращенным формам правых частей уравнений. Реализация каждого из уравнений предлагает использование триггеров с системами собственных функ-

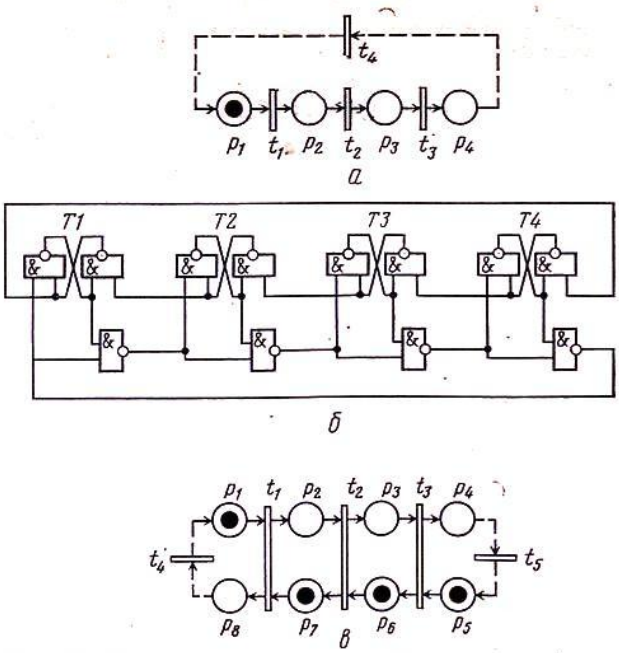


Рис. 7. Пример репозиции сети Петри: *а* – последовательный процесс с простой репозицией; *б* – его реализация на элементах Давида; *в* – его конвейерный аналог

ций вида $z_i=S_i\vee\bar{R}_iz_i$, $S_iR_i=0$. При этом все триггеры должны иметь одинаковые транзитные состояния, т. е. в процессе смены состояния оба плеча триггера должны принимать либо значение 1, либо значение 0. При соблюдении указанных условий поведение реализующей схемы не зависит от величин задержек элементов, т. е. схемы принадлежат к классу *не зависящих от скорости* или, иначе, к *самосинхронизирующимся* или *апериодическим* схемам [2]. При использовании базиса И-ИЛИ-НЕ для реализации n уравнений требуется $2n$ элементов.

Для счетного триггера функции возбуждения составляющих его триггеров будут иметь вид: $S_1=\bar{z}_2\bar{z}_3$, $R_1=z_2\bar{z}_3$, $S_2=z_1z_3$, $R_2=\bar{z}_1z_3$, $S_3=z_1z_2\vee z_1z_2$, $R_3=z_1z_2\vee\bar{z}_1\bar{z}_2$. Отсюда легко получить реализацию, соответствующую (3.1). По-видимому, приведенную выше оценку можно улучшить до величины порядка $n+\log_2 n$.

Нетривиальными оказываются методы синтеза апериодических схем в различных функционально полных для этих классов базисах, в том числе базисах, содержащих элементы с ограниченным числом входов. Однако сведения СП к схемам Маллера является лишь одной из возможных альтернатив.

В соответствии с подходом, предложенным в [62, 176] (см. также [2]), фрагментам простой, устойчивой и безопасной СП можно поставить в соответствие элементы асинхронной схемы следующим образом: позиции без маркера — провод, позиции с маркером — инвертор, переходу с несколькими входами — так называемый Г-триггер [2] или С-элемент [14] с собственной функцией $y=x_1x_2\vee(x_1\vee x_2)y$ (при двух входах x_1 и x_2), переходу с одним входом — элемент задержки или провод.

Еще один подход состоит в том, что каждой позиции устойчивой и безопасной сети ставится в соответствие триггер с вентилем; такая схема иногда называется *ячейкой Давида* [43, 45]. В частности, для СП, изображенной на рис. 7, *а*, реализация показана на рис. 7, *б*.

Реализации такого типа соответствует отличная от принятой трактовки правил срабатывания СП. Ранее говорилось, что срабатывание перехо-

да есть неделимая операция: маркеры изымаются из входных позиций перехода и перемещаются в его выходные позиции строго одновременно. В данном случае после срабатывания i-го триггера, $i=1, 2, 3, 4$, сначала устанавливается $(i+1)$-й триггер, а затем сбрасывается i-й. Это соответствует тому, что после срабатывания возбужденного перехода сначала устанавливаются маркеры в его выходных позициях, а затем происходит изъятие маркеров из входных позиций.

Именно такая интерпретация срабатывания в данном случае допускает переход к аппаратной реализации. В случае нескольких входных и выходных позиций триггеры приходится строить на элементах И-ИЛИ-НЕ [2]. Заметим, что для корректной реализации петли в СП должны содержаться не менее трех позиций. По-видимому, этот подход, несмотря на заметную избыточность, может с успехом применяться в промышленной автоматике, где снабженные световыми индикаторами триггеры могут быть использованы не только в схеме управления технологическим процессом, но также и для визуализации его состояния в мнемосхеме.

Различным аспектам применения СП для модернизации и синтеза аппаратных средств посвящены работы [2, 3, 8, 12, 17, 24, 25, 54, 175–177, 220]. СП также используются для описания блочных асинхронных систем [43, 45, 62, 138, 156, 230], в том числе для исследования надежностных свойств разрабатываемых систем [25, 96, 150, 151, 209, 211]. Неисправности компонентов систем моделируются образованием и исчезновением маркеров. Устойчивость к неисправностям осуществляется за счет избыточности, вводимой путем соответствующего кодирования.

Моделирование конвейерных процессов. Если природа моделируемого процесса такова, что его единственному начальному состоянию однозначно соответствует определенное финальное состояние, то в представляющей такой процесс СП также должна быть *единственная результирующая маркировка*. В этих случаях имеет смысл говорить о подклассе СП, задаваемых пятеркой $S=\langle T, P, E, \mu^0, \mu^r\rangle$.

Поскольку μ^r является тупиковой маркировкой, описание посредством СП может отражать лишь поведение однократно запускаемого процесса. Для возобновления процесса необходимо ввести механизм перехода от результирующей маркировки μ^r к начальной μ^0 или, иначе, механизм *репозиции* [5]. Механизм репозиции сети $S=\langle T, P, E, \mu^0, \mu^r\rangle$ естественно моделировать с помощью некоторой СП $S_1=\langle T_1, P_1, E_1, \mu^0, \mu_1{}^r\rangle$, для которой $\mu_1{}^r=\mu^0$ и, кроме того, либо $\mu_1{}^0=\mu^r$, либо $\mu_1{}^0=\mu^i$, где μ^i — текущая (промежуточная между μ^0 и μ^r) маркировка СП S из множества M допустимых. В первом случае репозицию назовем простой. Вторая (непростая) отличается от нее тем, что в качестве ее начальной маркировки $\mu_1{}^0$ может быть использована текущая маркировка, отличная от μ^r.

Объединение исходной СП и ее репозиции приводит к так называемой *автономной* СП S^A, в которой нет смысла выделять начальную и результирующую маркировки (вернее, таковыми могут являться любые маркировки из M). Для простой репозиции множество маркировок сети S^A совпадает с множеством маркировок сети S, для непростой репозиции $M^A \supset M$: появляются новые маркировки, не фигурировавшие ни в S, ни в S_1.

В проекции S^A на S можно выделить подмножество маркировок, совпадающих с маркировками исходной СП S и среди них μ^0 и μ^r. Если на каждое достижение μ^0 сеть будет отвечать достижением μ^r, то процесс в сети S^A будем называть *конвейерным*. Это определение не является формальным; необходимые детали читатель уяснит из приведенного ниже примера. Конвейерный процесс моделирует не только процесс обработки данных, но, кроме того, и механизм их «подкачки».

Конвейерные процессы и соответствующие им СП моделируют один из эффективных методов введения параллелизма, широко используемый как на уровне программного обеспечения, так и на уровне аппаратной поддержки. В основе конвейерного принципа обработки информации лежит

идея такой загрузки задачами последовательности операторов (модулей системы), при которой каждая следующая задача поступает на решение еще до того, как завершилось решение предыдущей.

Для понимания принципа работы асинхронного конвейера воспользуемся следующей аналогией. Представим себе наклонный желоб, по которому скатываются шарики. Когда выпущен первый шарик, на верхний конец желоба можно поместить другой, потом третий и т. д. Шарик может догнать предыдущий или отстать от него, но не может перегнать: догнав предыдущий шарик, он будет катиться не быстрее его.

При конвейерной обработке каждый оператор может быть инициирован каждый раз, когда непосредственно предшествующий и непосредственно следующий за ним операторы завершили выработку результатов. При этом предшествующий оператор подготовил информацию для обработки данным оператором для очередной задачи, а последующий оператор использовал информацию, выработанную данным оператором при решении предыдущей задачи.

В качестве примера, доказывающего возможность повышения эффективности за счет конвейеризации, приведем модель конвейера рис. 7, *в* для реализации последовательной работы трех операторов — рис. 7, *а*. Операторы здесь моделируются переходами. Расстановка маркеров по позициям соответствует начальной маркировке. СП рис. 7, *а* моделирует процесс решения одной задачи. Каждая следующая задача может быть инициирована после последовательности $t_1t_2t_3t_4$ срабатывания переходов, т. е. после того, как будет решена предыдущая задача (маркер при этом находится в позиции p_4). Репозиция осуществляется фрагментом СП, включающим переход t_4 и инцидентные позиции p_4 и p_1. На рис. 7, *в* уже моделируются две задачи, хотя в различных фазах решения могут одновременно находиться до трех задач. Следующая задача может быть инициирована после срабатывания переходов t_1, t_2 и t_4, когда результатов решения еще нет (отсутствует маркер в позиции p_4). Здесь непростая репозиция осуществляется фрагментом СП, состоящим из переходов t_4 и t_5 с инцидентными позициями p_8, p_1 и p_4, p_5 соответственно.

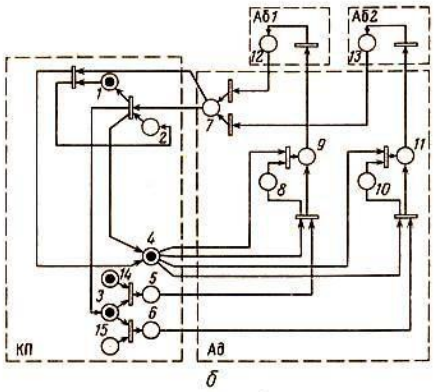


Рис. 8. Пример задания протоколов информационного обмена: *а* — схема связи КП с Аб1 и Аб2 через адаптер; *б* — протокол

Системы с конвейерным способом обработки информации находят все большее применение (см., например, [47]). В литературе на русском языке используются синонимические названия — *конвейерные*, *магистральные*, *желобковые*, *водопроводные* и, наконец, *пайплайновые* («фонетическая калька») с анг. pipeline — трубопровод). Аппаратная реализация конвейерных устройств типа регистров и счетчиков рассматривалась в [2].

Моделирование протоколов информационного обмена. Под протоколами обычно понимают свод правил, фиксирующих порядок установления связи, координированного взаимодействия и обмена данными между объектами систем и сетей. Если первоначально протоколы

задавались на естественных языках, создающих иллюзию легкости понимания, то впоследствии оказалось, что такой способ громоздок и, что хуже, допускает неоднозначное толкование. Продолжающийся поиск удобных языков задания протоколов преследует цель сделать их более компактными, однозначно толкуемыми, пригодными для анализа на полноту и непротиворечивость и допускающими непосредственный переход к синтезу интерфейсных устройств и анализу нежелательных эффектов в процессе взаимодействия. Как показали исследования [5, 46, 61, 68, 151], СП оказались удобным инструментом решения этих задач.

В качестве примера протокола обмена рассмотрим систему, состоящую [5] из канального процессора КП, адаптера Ад и двух абонентов Аб1 и Аб2 (рис. 8, *а*). КП связан с абонентами общей магистралью данных МД, при помощи которой все эти устройства осуществляют прием и передачу информации. Система работает по принципу ведущий-ведомый. Роль ведущего играет ВП, ведомых — абоненты. В исходном состоянии $a=a_1=a_2=b_1=b_2=b=0$. КП устанавливает адрес абонента и затем выставляет сигнал запроса $a=1$. По адресу и этому сигналу адаптер вырабатывает либо $a_1=1$, либо $a_2=1$, инициирующие прием или передачу информации. После окончания работы с информацией соответствующий абонент отвечает установкой $b_i=1$. После приема этой квитанции адаптер устанавливает $b=1$. Затем начинается процесс отключения абонента: КП устанавливает $a=0$ (при этом он может менять адрес), по которому адаптер сбрасывает в 0 тот сигнал a_i, который был равен 1. В ответ на $a_i=0$ i-й абонент устанавливает $b_i=0$, после чего адаптер выставляет $b=0$, и система оказывается в исходном состоянии.

На рис. 8, *б* изображена СП, задающая взаимодействие указанных устройств на уровне установления и разрыва связи. Эта сеть состоит из четырех фрагментов, относящихся к каждому из взаимодействующих устройств. Ниже дано соответствие наличия маркеров в некоторых позициях значениям сигналов обмена: 4,5 — $a=1$, адрес Аб1; 4,6 — $a=1$, адрес Аб2; 2,4 — $a=0$; 7 — $b=1$ или $b=0$; 9 — $a_1=1$ или $a_1=0$; 11 — $a_2=1$ или $a_2=0$; 12 — $b_1=1$ или $b_1=0$; 13 — $b_2=1$ или $b_2=0$; 14 — адрес Аб1; 15 — адрес Аб2. Одновременное появление маркеров в позициях 14 и 15 исключается.

Сеть рис. 8, *б* относится к классу устойчивых и безопасных СП, что позволяет синтезировать управляющий автомат адаптера методами, изложенными в [2].

4. Функциональные возможности сетей Петри. Для исследования «моделирующей способности» СП и сравнения ее с возможностями других известных в теории автоматов объектов (машин Тьюринга, конечных автоматов и т. д.) вновь прибегают к интерпретации СП.

Через $S^{\Sigma}=\langle P, T, E, \mu^0, \mu^r\rangle$ обозначим класс СП, удовлетворяющий следующим ограничениям: 1) начальная маркировка μ^0 представляет собой вектор, одна компонента которого равна 1, а остальные — 0; 2) выделена некоторая финальная маркировка из множества M допустимых. Образуем двойку $\langle S^{\Sigma}, \Psi\rangle$, где Ψ — семантическая интерпретация S^{Σ}, а также двойку $\langle\Sigma, \sigma\rangle$, в которой Σ — конечный алфавит (множество символов, включающее в общем случае пустой символ λ), а σ — функция наименований, отображающая множество переходов СП на алфавит Σ, σ: $T\rightarrow\Sigma$.

Как уже известно, функционирование СП порождает некоторые последовательности (траектории) срабатываний переходов. Поскольку функция наименований ставит в соответствие каждому переходу СП некоторый символ из Σ, данная сеть порождает некоторые строки (последовательности символов), каждая из которых соответствует одной возможной траектории. В силу принятого ограничения все строки должны начинаться символами, именующими переходы, возбужденные при начальной маркировке. Говорят, что множество строк, соответствующих данной СП, образует ее язык. Тогда множество всевозможных СП, удовлетворяющих ограничениям, определяет языки Петри или, короче, *СП-языки*.

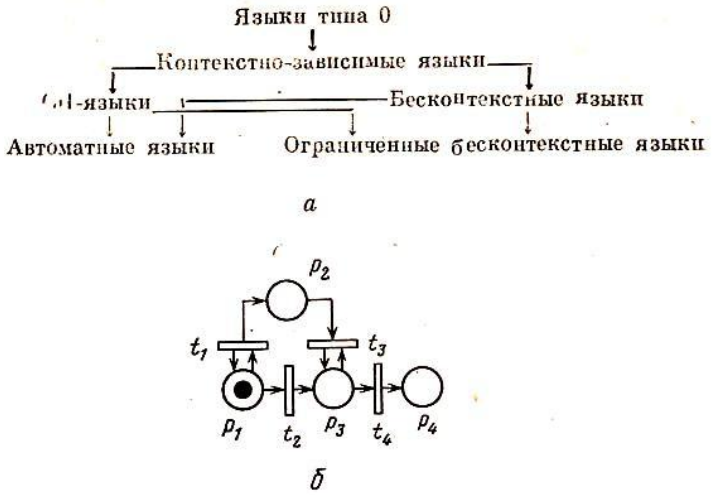


Рис. 9. Языки, порождаемые сетями Петри: *а* – место СП-языков в иерархии языков; *б* – пример

Мощность класса СП-языков в определенной степени позволяет судить о функциональных возможностях СП, об их моделирующей способности. С другой стороны, узость языкового подхода состоит в том, что языковые представления (если не рассматривать языков, порождаемых графовыми грамматиками) по сути дела носят чисто последовательный характер, ибо они не учитывают параллелизм. Эта неадекватность существенна, но не отвергает попыток использования языковых средств для анализа функциональных возможностей СП.

Результаты исследований показали, что СП-языки по своей мощности не уступают бесконтекстным языкам. Класс контекстно-зависимых языков включает класс СП-языков, а язык регулярных выражений и автоматные языки суть подклассы СП-языков. Иерархия различных формальных языков показана на рис. 9, *а*, где дуги означают отношение нестрогого включения. Точнее, если i-й язык включает в себя j-й, то на рисунке это изображается дугой, исходящей из i и входящей в j.

Более детальные исследования базировались на различных ограничениях, накладываемых на функции наименований и результирующую маркировку. Можно выделить три подкласса функций наименований: *бесповторные* — когда каждый переход именуется новым символом, т. е. такие наименования, когда $\sigma(t_i) \neq \sigma(t_j)$ для $i \neq j$; *повторные* — когда разные переходы могут быть обозначены одинаково; *наименования*, в которых некоторые переходы обозначаются пустым символом λ (который в этом случае, естественно, входит в алфавит Σ, но не входит ни в одну из порождаемых СП строк). Очевидно, что разные подклассы функций наименований для одной и той же СП могут приводить к различным СП-языкам.

Действительно, пусть для СП, изображенной на рис. 9, *б*, $\mu^0=(1000)$, а $\mu^r=(0001)$, т. е. рассматриваемый язык есть множество последовательностей срабатываний переходов, соответствующих перемещению маркера из позиций p_1 в позицию p_4. Тогда при бесповторной функции наименований $\sigma_1(t_1)=a$, $\sigma(t_2)=c$, $\sigma_1(t_3)=b$, $\sigma_1(t_4)=d$ порождается язык $\{a^n c b^n d \mid n \geqslant 0\}$, при повторной функции наименований $\sigma_2(t_1)=a$, $\sigma_2(t_2)=a$, $\sigma_2(t_3)=b$, $\sigma_2(t_4)=b$ — язык $\{a^n b^n \mid n>0\}$. Если же $\sigma_3(t_1)=\lambda$, $\sigma_3(t_2=c$, $\sigma_3(t_3)=b$, $\sigma_3(t_4)=d$, то порождается язык регулярных выражений $\{c, b^*d\}$.

В приведенных примерах результирующей являлась единственная маркировка (0001). Вообще говоря, к результирующим маркировкам могут быть отнесены маркировки, определяемые следующими четырьмя способами: 1) все множество допустимых маркировок; 2) множество маркиро-

вок, не меньших любой маркировки из множества заданных; 3) все тупиковые маркировки (т. е. маркировки, при которых не возбужден никакой переход и недостижима никакая другая маркировка; 4) любая одна маркировка из множества допустимых.

Возвратившись к рис. 9, *б* и рассматривая ту же бесповторную функцию наименований, что и выше, в предположении, что результирующей маркировкой является: а) единственная маркировка (0010); б) любая маркировка, не меньшая (0010); в) тупиковая маркировка (0001); все достижимые из (1000) маркировки, получим соответственно языки: а) $\{a^n cb^n \mid n \geqslant 0\}$; б) $\{a^m cb^n \mid m \geqslant n \geqslant 0\}$; в) $\{a^m cb^n d \mid m \geqslant n \geqslant 0\}$; г) $a^m \mid m \geqslant 0\} \cup \cup \{a^m cb^n \mid m \geqslant n \geqslant 0\} \cup \{a^m cb^n d \mid m \geqslant n \geqslant 0\}$.

Итак, имея три типа функций наименований и четыре различных типа результирующих маркировок, можно определить 12 классов СП-языков. Эти классы более или менее подробно исследованы [93, 95, 178—180], хотя полученные результаты, как и большинство результатов по исследованию СП-языков, представляют скорее интерес собственно для теории формальных грамматик и языков, чем для приложений.

Исследованиям по СП-языкам и связанным с ними проблемам посвящены работы [31, 55—57, 59, 69, 82, 93, 95, 109, 129, 132, 133, 172, 179, 180, 227]. Из имеющихся попыток языкового подхода следует выделить те из них, которые связаны с анализом разрешимости некоторых проблем для СП, о чем будет сказано ниже.

Попутно заметим, что поскольку множество строк языков, порождаемых СП, может быть достаточно велико, для описания классов строк вводятся такие операции, как *объединение, пересечение, конкатенация* (смысл которых очевиден), но, кроме того, еще операции *реверса* (прочтение символов строки в обратном порядке, справа налево), а также операция *перемешивания* или *тасовки*. Если обозначить через ab строку, в которой символ a предшествует символу b, через $a+b$ — независимое появление этих символов, а через Δ — оператор перемешивания, то результат его применения определяется следующими выражениями: $ab\Delta c = abc+acb+cab$; $(a+b)\Delta c = ac+ca+bc+cb$. Применение указанных операций дает более экономный способ описания функционирования СП, чем просто через допустимые последовательности срабатываний поименованных переходов.

5. Анализ сетей Петри. Как неоднократно отмечалось выше, СП являются удобным средством моделирования поведения параллельных асинхронных процессов. Семантически интерпретированные СП, являющиеся дескриптивной моделью каких-либо объектов, могут быть подвергнуты анализу с нужной степенью детализации. Здесь будет идти речь только об анализе неинтерпретированных СП, которые содержат всю необходимую информацию о динамическом поведении моделируемых объектов безотносительно к их природе, содержательной интерпретации его условий и событий, соответствующих позициям и переходам СП. Основной целью анализа неинтерпретированных СП является наличие или отсутствие тех или иных свойств моделируемого процесса, в первую очередь — тех, которые были использованы в качестве классификационных признаков в **п. 2.** Таким образом, типовыми *задачами анализа* являются проверка СП на ограниченность, консервативность, безопасность, устойчивость, живость (назовем их задачами первого рода), а также на принадлежность подклассам свободного выбора, простым, бесконфликтным и автоматным сетям, маркированным графам и т. п. (задачи второго рода).

Задачи второго рода здесь рассматриваться не будут в силу того, что они просто решаются методами теории графов на основе рассмотрения локальных признаков.

Разрешимость задач анализа сетей Петри. Поскольку задачи первого рода связаны с возможными типами маркирования СП, их решение естественно связать с решением фундаментальной проблемы,

Таблица

Класс СП	Проблемы				
	Достижимость	Живость	Покрываемость	Ограниченность	Консервативность
Автоматные	NSL-полная				Тривиальная
Маркированные графы	NSL-трудная *	NSL-полная *	NSL-трудная *		NSL-полная
1-консервативные со свободным выбором	DSP-полная	NTP-полная *	DSP-полная		Тривиальная
Произвольные со свободным выбором	DSE-трудная	NTP-полная *	DSE-трудная	DSP-полная	NTP
1-консервативные	DSP-полная	DSP	DSP-полная		Тривиальная
Произвольные	DSE-трудная			DSP-полная	NTP

* Оценки справедливы для случаев, когда сети не обладают соответствующим свойством.

в роли которой в теории СП выступает проблема достижимости. Ее формулировка такова.

Дана СП $S=\langle P, T, E, \mu^0\rangle$. Требуется определить, достижима ли из ее начальной маркировки μ^0 некоторая фиксированная маркировка μ, т. е. принадлежит ли μ множеству M достижимых маркировок.

Принципиальная трудность в решении этой задачи состоит в том, что диаграммы СП могут содержать бесконечное число достижимых маркировок. Идея получения конечного представления диаграммы базируется на «стягивании» всех маркировок, строго больших (покомпонентно) некоторой допустимой, в эту последнюю. Эта идея хорошо иллюстрируется простейшим примером неограниченной СП, приведенной на рис. 3(I) (в верхнем правом углу таблицы). Ее диаграмма, содержащая бесконечное число маркировок, $10\to 11\to 12\to 13\to\ldots$, может быть сведена к следующей: $10\to 1\omega$, где ω можно трактовать как символ сколь угодно большого числа. Наличие символа ω в диаграмме свидетельствует о том, что СП не является ни ограниченной, ни консервативной (в консервативных сетях множество M допустимых маркировок конечно [114]).

Выяснение разрешимости проблемы достижимости потребовало немалых усилий специалистов [32, 33, 44, 53, 56, 86, 89, 90—94, 122, 144, 159, 160, 163, 199, 228]. Первое доказательство разрешимости этой проблемы было получено в [199], но впоследствии оказалось, что в нем были допущены ошибки. Об этом сообщалось, в частности, в [11]. В [159], однако, содержится датированная 17.04.80 информация о том, что E. W. Мауг получил искомое доказательство (пока еще оно не опубликовано). Если факт существования алгоритма анализа достижимости подтвердится, то разрешимой можно считать и проблему живости (принадлежности СП классу живых), поскольку эти задачи рекурсивно сводятся [91] одна к другой.

В ходе исследований выяснилась также эквивалентность проблемы достижимости следующим задачам: достижимости пустой (содержащей нули во всех позициях) маркировки [176]; достижимости подмножества позиций (равны ли значения определенных компонент заданной маркировки и какой-либо достижимой маркировки) [90, 144]; покрываемости (существует ли конечная маркировка μ', не обязательно достижимая, такая, что $\mu'\geqslant\mu$ покомпонентно) [91, 119].

Добавим, что: задача анализа принадлежности СП классу устойчивых

рекурсивно сводится к проблеме достижимости [114], в то время как существование обратной сводимости не доказано [131]; необходимые и достаточные условия живости СП свободного выбора установлены Ф. Коммонером и опубликованы в [95]; алгоритм анализа маркированных графов на живость и безопасность, а также решение для них проблемы достижимости предложены в [52, 102].

Сложность задач анализа сетей Петри. Исследование вычислительной сложности проблем достижимости, живости, покрываемости, ограниченности и консервативности для различных классов СП было проведено в [114, 140]. Результаты этих исследований, базирующихся на использовании теории формальных грамматик, собраны в таблице, в которой даны оценки вычислительной сложности некоторых проблем для различных классов сетей Петри, а обозначения NSE, DSP, DSE, NSL, NTP, NSP расшифровываются следующим образом. Первый символ в них указывает тип распознавающей машины Тьюринга — детерминированная (D) или недетерминированная (N), второй символ — емкостную (S) или временную (T) сложность распознавания (число требуемых ячеек ленты и число шагов соответственно), третий символ — собственно оценку сложности: P — полиноминальная; E — экспоненциальная, L — логарифмическая.

Известно, что DSP=NSP, DSE=NSE, а также что NSL⊆DTP⊆NTP⊆DSP. Имеют ли место другие включения — неизвестно.

Определения полной и трудной проблемы читатель найдет в [114].

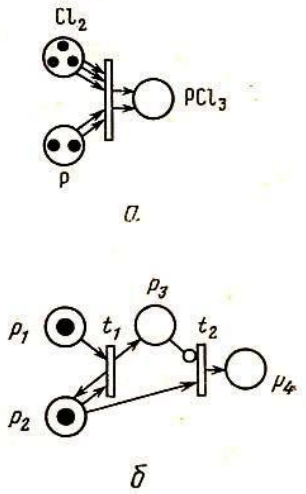


Рис. 10. Примеры обобщений сетей Петри: *а* — задание в виде мультиграфа; *б* — использование тормозящих связей

6. Обобщение сетей Петри и родственные модели. Одной из попыток обобщения СП [54, 190] является переход от графа к мультиграфу, т. е. графу, позиции и переходы которого соединяются несколькими дугами. Оказалось, однако, [90], что такие СП эквивалентны ординарным, имеющим по одной дуге между парой переход-позиция или позиция-переход. Тем не менее, изобразительно такие СП могут быть удобнее в приложениях. В [180] приводится экзотический пример использования такого представления для описания химической реакции хлора и фосфора (рис. 10, *а*), в результате которой получается трехлористый фосфор. При срабатывании перехода из входных позиций изымаются все пять маркеров, а два маркера заносятся в выходную.

Существенное обобщение СП [18, 27, 173] достигается введением так называемых *тормозящих дуг,* изображаемых, например, кружочками на конце вместо стрелок. Тормозящий вход перехода реализует безусловный запрет его срабатывания, если в позиции, из которой исходит тормозящая дуга, имеется маркер. В противном случае (отсутствие маркера) этот вход игнорируется. Так, на рис. 10, *б* переход t_2 считается возбужденным лишь при условии, что в позиции p_3 маркер отсутствует, т. е. если раньше не сработал возбужденный при начальной маркировке 1100 переход t_1. Если же переход t_1 сработает раньше, чем t_2, то появление маркера в p_3 снимает возбуждение t_2. В этом примере p_3 имеет приоритет над p_2, поэтому введение тормозящих входов в некотором смысле эквивалентно введению приоритета между входами. В зависимости от времен срабатываний переходов t_1 и t_2 результирующих маркировок может быть две: либо 0110, либо 1001.

К введению приоритета сводятся и другие известные обобщения СП. К их числу относятся, в частности, так называемые *раскрашенные* СП

[113], в которых маркеры могут быть разноцветными, а также *временные* (синхронизированные) СП [58, 81, 93, 114, 130, 150, 152, 155, 156, 164]. В таких сетях каждому переходу $t \in T$ ставится в соответствие пара вещественных чисел τ_1, τ_2. Если переход стал возбужденным в момент времени τ, то он не может сработать раньше, чем в момент $\tau+\tau_1$ и, более того, должен сработать не позже, чем в момент $\tau+\tau_2$. Доказано, что проблемы достижимости и ограниченности для таких сетей неразрешимы [114]. При доказательстве использовались результаты, полученные в [69].

Оказалось, что попытки сколь-нибудь существенного обобщения модели СП немедленно приводят к эквивалентности с машинами Тьюринга и, следовательно, к неразрешимости проблем анализа.

Несколько особняком стоят такие модельные расширения СП, как стохастические СП [205], в которых переходам приписаны вероятности их срабатывания при возбуждении, модели, описанные в [124], [139], а также преследующие чисто дескриптивные цели — например, [229].

Различные графовые параллельных вычислений, отличные от модели СП, предлагались и изучались в работах [1, 10, 26, 29, 46, 70, 76, 77, 83, 85, 88, 89, 111, 119, 121, 123, 125, 145, 166—170, 198, 213, 226]. Большинство из них обладают одинаковыми с СП функциональными возможностями [19], так что их можно считать модификациями СП. Другие базируются на введении некоторых структурных ограничений на СП.

Возможности и изобразительные средства родственных СП моделей могут варьироваться в широких пределах. Здесь не преследовалась цель сколь-нибудь подробно их анализировать, тем более, что о некоторых из этих моделей упоминается в [6], а достаточно подробный обзор выполнен в [28]. Сравнительное изучение разных моделей проводилось в [32, 70, 79, 112, 142, 152, 182]. Установление наличия отношений включения и эквивалентности является весьма тонкой проблемой хотя бы в силу того, что могут быть предложены самые разные формальные толкования этих отношений [142].

Необходимость применения многих родственных моделей СП вместо удобной и синтаксически весьма простой базовой модели, какой являются СП, вряд ли оправдана, хотя некоторые модели могут быть более удобными в определенных предметных областях. Следует выделить эквивалентные СП [90] так называемые системы векторного сложения [33, 82, 89, 90, 91, 92, 119, 120, 163, 192, 199, 228] и системы векторного замещения [122, 150], оказавшиеся инструментом для получения многих результатов, используемых в теории СП.

Исследования как по самим СП, так и по родственным моделям базировались на изучении таких принципиально важных для параллельных систем и процессов свойств, как параллелизм, синхронизм, одновременность, детерминизм, и вопросов координации процессов [18—20, 118, 141, 153, 178, 191]. Модели асинхронного процесса [5], систем переходов [111, 126] и некоторые другие являются по существу метамоделями, в терминах которых могут быть интерпретированы не только СП, но и вышеперечисленные модели. Общие вопросы проблематики СП обсуждаются в [39—42, 157, 204].

7. Историко-географический комментарий. Теории СП исполнилось два десятилетия. Ее основоположником является Карл Адам Петри, который в 1962 г. предложил новую модель информационных потоков в системах [183]. Выдвинутая им идея (см. также [184]) была подхвачена в США двумя научными коллективами — группой А. Холта из фирмы Applied Data Research, работавшей в рамках проекта по теории информационных структур, и группой вычислительных структур (Computation Structure Group — CSG) Массачусетского технологического института (MIT) во главе с Дж. Деннисом. В работах обеих групп модель получила свое название, определение и способ изображения. Были исследованы различные классы СП, поставлены и частично решены фундаментальные вопросы теории, определены основные направления приложений.

Работы К. А. Петри [185—190], А. Холта [51, 52, 101—103] преследовали цель нового прочтения общей теории систем в части создания эффективных способов параллельного управления асинхронными информационными потоками. Работы группы CSG [33, 62—65, 71, 88—95, 154, 173—177] и др. развивались в несколько ином направлении — с акцентом на вопросы параллельного программирования и организации распределенных систем. Следует отметить органичную связь развиваемой теории СП тематики с традиционными проблемами теории автоматов.

Две школы были созданы в Европе. В Бонне (Institut für Informationssystemforshung, Gesselshaft für Matematik und Datenverarbeitung — GMD) работы велись в направлении создания абстрактной теории СП [73—8Л, 134—137] и др. Во Франции, где наблюдался, пожалуй, наибольший темп роста исследований по теории СП и ее приложениям [24—25, 36—38, 43, 45, 54, 81, 138, 146—148, 155, 156, 195, 207—212, 221—222, 226, 227 и др.], они концентрировались в трех регионах — Тулузе, Гренобле и Париже. в Тулузе, в Центре аэрокосмических исследований (ONERA) основной проблематикой было использование аппарата СП для проектирования схем промышленной автоматики [43, 45, 54]. В Гренобле (Национальный политехнический институт) развивались общие вопросы теории, в особенности методы упрощения СП [36—38, 60], а также ее применения к вопросам программирования и построения надежных систем [211]. Часть работ выполнена в Париже, в Институте программирования. Достаточно полный обзор работ французских ученых по тематике СП содержится в [23].

Несколько томов такого солидного повторяющегося издания, как Lect Notes Comput. Sci. [143, 164], а также ряд последующих томов — vol. 88, 100, 107, не фигурирующих в списке литературы — почти целиком посвящены проблематике СП и смежным вопросам. В 1977 г. был опубликован достаточно подробный обзор Дж. Л. Питерсона [180], а в 1981 г.— его же книга [181].

В СССР основные исследования в области СП и приложений ведутся в Новосибирске (Вычислительный центр и Институт математики СО АН СССР) под руководством В. Е. Котова и О. Л. Бандман, в Ленинграде (Ленинградский электротехнический институт им. В. И. Ульянова (Ленина) и ВЦ Института социально-экономических проблем АН СССР) под руководством В. И. Варшавского, в Киеве (А. Н. Чеботарев, В. В. Литвинов, Институт кибернетики АН УССР) и Москве (А. А. Таль, С. А. Юдицкий, Институт проблем управления). Первое упоминание в отечественной литературе о СП как модели параллельного программирования содержалось, по-видимому, в обзоре [9], первым изданием, в котором описывались СП, следует считать коллективную монографию [2]. В. Е. Котов развил аппарат алгебры регулярных сетей Петри, строящихся из класса элементарных сетей применением определенных операций [11, 128]. Вопросы использования СП для построения асинхронных схем, поведение которых не зависит от величин задержек их элементов (апериодических схем) всесторонне рассматривались в [2]. Существенной особенностью направления, развиваемого школой В. И. Варшавского, является ориентация на развитие аппаратных средств управления асинхронными процессами различной физической природы — формальная модель асинхронного процесса введена в [5] — в том числе конвейерными. Заслуживает упоминания введение нового класса СП — схемных [8] — удобных для анализа динамического поведения схем с произвольными задержками, отличающихся от похожей модели [85] раздельным рассмотрением включения и выключения элементов (что существенно уменьшает сложность описания схем).

Работы ИМ СО АН СССР связаны с использованием СП для решения задач параллельного программирования [4, 12] и синтеза асинхронного микропрограммного управления параллельными процессами [1, 3, 12]. Г. П. Плакс из Таллина применил СП для синтеза управляющих структур вычислительных моделей [15]. Вопросы описания и синтеза систем управ-

ления сложными дискретными процессами на основе СП применительно к схемам промышленной автоматики развивались и использовались в ИПУ [17].

География исследований по СП за последние годы существенно расширилась, выйдя за пределы упомянутых центров. Постепенно увеличивается размах работ по СП в странах СЭВ [7, 191, 215].

Список литературы дает достаточно четкое представление о развитии работ по сетям Петри во всем мире.

После написания обзора автору стало известно о выходе из печати двух книг на немецком языке: Starke P. H. Petri-Netze.— Dautscher Verlag der Wissenchaften, Berlin, 1980; Reisig W. Petrinetze. Eine Einfürung.— Springer-Verlag, Berlin, Heidelberg, New York, 1982.

Автор признателен В. А. Песчанскому, М. А. Кишиневскому и А. Р. Таубину за советы и замечания. Двое последних приняли активнейшее участие в составлении библиографии.

ЛИТЕРАТУРА

1. *Анишев П. А.* О детерминированности параллельных граф схем.— Вычислительные системы. Вып. 73, Новосибирск: 1978.
2. Апериодические автоматы. Под ред. В. И. Варшавского, М.: Наука. 1976.
3. *Бандман О. Л.* Синтез асинхронного микропрограммного управления параллельными процессами. Препринт ОВС-07, ИМ СО АН СССР, Новосибирск: 1978.
4. *Бандман О. Л., Пискунов С. В., Сергеев С. К.* Задачи параллельного программирования.— Вычислительные системы. Вып. 73, Новосибирск: 1978.
5. *Варшавский В. И., Мараховский В. Б., Песчанский В. А., Розенблюм Л. Я.* Асинхронные процессы. I. Определение и интерпретация. II. Композиция и согласование.— Изв. АН СССР. Техн. кибернет., 1980, № 4, № 5.
6. *Головкин Б. А.* Параллельная обработка информации, программирование, вычислительные методы, вычислительные системы.— Изв. АН СССР. Техн. кибернет., 1979, № 2.
7. *Кениг Р.* Минимизация сетей Петри, интерпретированных с точки зрения техники управления системами.— В сб.: Автоматизированное проектирование дискретных управляющих устройств. М.: Наука, 1980.
8. *Кишиневский М. А., Таубин А. Р., Цирлин Б. С.* Сети Петри и анализ переключательных схем.— Кибернетика, 1982, № 4.
9. *Котов В. Е.* Теория параллельного программирования. Прикладные аспекты.— Кибернетика, 1974, № 1, № 2.
10. *Котов В. Е.* О параллельных языках.— Кибернетика, 1980, № 3, № 4.
11. *Котов В. Е.* Алгебра регулярных сетей Петри.— Кибернетика, 1980, № 5.
12. Методы параллельного микропрограммирования. Под ред. О. Л. Бандман. Новосибирск: Наука, 1981.
13. *Миренков Н. Н.* О синхронизации процессов.— Кибернетика, 1979, № 1.
14. *Миллер Р. Е.* Теория переключательных схем. Т. 2. М.: Наука, 1971.
15. *Плакс Т. П.* Синтез параллельных программ на вычислительных моделях.— Программирование, 1977, № 4.
16. *Розенблюм Л. Я., Яковлев А. В.* Инструментальная модель диалога.— В кн.: Диалог в автоматизированных системах. М.: МДНТП, 1981.
17. *Таль А. А., Юдицкий С. А.* Иерархия и параллелизм в сетях Петри. I, II.— Автоматика и телемеханика, 1982, № 7, № 9.
18. *Agerwala T.* A complete model for representing the coordination of asynchronous processes. Hopkins Comput. Res. Rep. 32, J. Hopkins Univ., Baltimore, Md., July 1974.
19. *Agerwala T.* An analysis of controlling agents for asycnhronous processes. Hopkins Comput. Res. Rep. 35, J. Hopkins Univ., Baltimore, Md., Aug. 1974.
20. *Agerwala T., Choed-Amphai Y. G.* A synthesis rule for concurrent systems. Proc. 15th Ann. Des. Autom. Conf., Las Vegas, N. Y., 1978.
21. *Agerwala T., Flynn M.* Comments on capabilities, limitation and correctness of Petri nets. Proc. 1st Ann. Symp. Comput. Architecture, ACM, N. Y., 1973.
22. *Anderson D. W., Sparacio F. J., Tomasulo R. M.* The IBM System/360 Model 91: Machine philosophy and instruction handling. IBM J. Res. and Dev., vol. 11, 1967, No 1.
23. *Andre C., Diaz M., Girault C., Sifakis J.* Survey of French research and aplications based on Petri nets. In [164].
24. *Auguin M., Boeri F., Andre C.* Systematic method of realization of interpreted Petri nets. Digit. Process., vol. 6, 1980, No 1.
25. *Azema P., Valette R., Diaz M.* Petri nets as a common tool for design verification and hardware simulation. 13th Des. Autom. Conf., Palo Alto, June 1976.

26. *Baer J.* Graph models of computations in computer systems. PhD Thesis, Univ. of Calif., Los Angeles, 1968.
27. *Baer J. L.* Modelling for parallel computation: a case study. Proc. 1973 Sagamore Comput. Conf. Parallel Processing. Springer Verlag, N. Y., 1973.
28. *Baer J. L.* A survey of some theoretical aspects of multiprocessing. Comput. Surveys, vol. 5, 1973, No 1.
29. *Baer J., Bouet D., Estrin G.* Legality and other properties of graph models of computation. J. ACM, vol. 17, 1970, No 3.
30. *Baer J. L., Estrin G.* Bounds for maximum parallelizm in a bilogical graph model of computation. IEEE Trans. Comput., C — 18, 1969, No 11.
31. *Baker H. G.* Petri nets and languages. CSG Memo. 68, Proj. MAC, MIT, Cambridge, Mass., May 1972.
32. *Baker H. G.* Equivalence problems of Petri nets. MS Thesis, Dept. Electr. Eng., MIT, Cambridge, Mass., June 1973.
33. *Baker H. G.* Rabin's proof of the undecidability of the reachability set inclusion problem of vector addition systems. CSG Memo. 79, Proj. MAC, MIT, Cambridge, Mass., July 1973.
34. *Bernstein A. J.* Program analysis for parallel processing. IEEE Trans. Electr. Comput., EC-15, 1966, No 10.
35. *Bernstein P. A.* Description problems in the modeling of asynchronous computer systems. TR 48, Dept. Comput. Sci., Univ. of Toronto, Jan. 1973.
36. *Bertholot G., Memmi G.* Analyse et reduction de réseaux de Petri. Rap. DEA, Inst. de Progr., Juin 1975.
37. *Bertholot G., Roucairol G.* Reduction of Petri nets. Lect. Notes Comput. Sci., vol. 45, 1976.
38. *Bertholot G., Roucairol G., Valk R.* Reduction of nets and parallel program. In [164].
39. *Best E.* Atomicity and activities. In [164].
40. *Best E.* The relative strength of k-density. In [164].
41. *Best E.* Adequacy of path program. In [164].
42. *Best E., Schmid H. A.* Systems of open path in Petri nets. Lect. Notes Comput. Scil., vol. 32, 1975.
43. *Blanshard M., Cavarroc J. C., Gillon J., Marshand J., Guidez G., Thuillier G.* Automatismes a séquences. Rap. final du contract DGRST No 71.7.2912. DERA, Toulouse, 1973.
44. *Börger E., Kline B. H.* The reachability problem for Petri nets and decision problems for Skolem arythmetic. Theor. Comput. Sci., vol. 14, 1980, No 2.
45. *Cavarroc J. C., Blandchard M., Gillon J.* An approach to the modular design of industrial switching systems. Int. Symp. Discrete Systems, Riga, Zinatne, vol. 3, 1974.
46. *Cerf V. G.* Multiprocessors, semaphores and a graph model of computation. PhD Thesis, Comput. Sci. Dept., Univ. Calif., Los Angeles, Apr. 1972.
47. *Chen T. C.* Overlap and pipeline processing. In: Introduction to computer architecture, H. S. Stone (Ed.), Sci. Res. Ass., Chicago, Ill., 1975.
48. *Clark W. A.* Macromodular computer systems. In: Proc. 1967, Spring Jt. Comput. Conf., Thompson Book Co., Washington, 1967.
49. *Coffman E. G. et al.* System deadlock. Comput. Surveys, vol. 3, 1971.
50. *Coffman E. G., Denning P. J.* Operating systems theory, ch. 2, Prentice — Hall, Englewood Cliffs, N. Y., 1973.
51. *Commoner F. G.* Deadlocks in Petri nets. CA — 7206—2311, Appl. Data Res., Wakefield, Mass., June 1972.
52. *Commoner F., Holt A. W., Even S., Pnueli A.* Marked directed graphs. J. Comput. Syst. Sci., vol. 5, 1971, No 10.
53. *Cordoza E., Lipton R. J., Meyer A. R.* Exponential space complete problems for Petri nets and commutative semigroups. Proc. 8th Ann. Symp. Theory of Computing, May 1976.
54. *Cottrez G., Blanshard M., Gillon J., Guidez G., Thuillier G.* The simulation of a switchin system's requirements. Int. Symp. Discrete Systems, Riga, Zinatne, vol. 3, 1974.
55 *Crespi-Reghizzi S., Mandrioli D.* Petri nets and commutative grammers. Internal Rep. 74—5, Lab. di Calcolatiri, Inst. di Elettr. ed Electr. del Politecnico di Milano, Mar. 1974.
56. *Crespi-Reghizzi S., Mandrioli D.* A decidability theorem for a class of vector — addition systems. Int. Proc. Letters, vol. 3, 1975, No 3.
57. *Crespi-Reghizzi S., Mandrioli D.* Properties of firing sequences. Proc. MIT Conf. Petri nets and related methods, MIT, Cambridge, Mass., July 1975.
58. *Crespi-Reghizzi S., Mandrioli D.* Some algebraic properties of Petri nets. Alta Frequeza, vol. XLV, 1976. No 2.
59. *Crespi-Reghizzi S., Mandrioli D.* Petri nets and Szilard languages. Inf. and Control., vol. 33, 1977, No 2.
60. *Dadda L.* The synthesis of Petri nets for controlling purposes and reduction of their complexity. 2nd Euromicro Symp. Microprocess. and Microprogram., Venice, 1976, Amsterdam, 1977.
61. *Dantine A. A. S.* Protocol representation with finite-state models. In [68].

62. *Dennis J. B.* Modular asynchronous control structures for a high performance processor. Rec. Proj. MAC Conf. Concurrent and Parallel Computation, ACM. N. Y., 1970.
63. *Dennis J. B.* (Ed.). Record Proj. MAC Conf. Concurrent and Parallel Computation, ACM, N. Y., 1970.
64. *Dennis J. B.* Concurrensy in software systems. CSG Memo. 65—1, Proj. MAC, MIT, June 1972.
65. *Dennis J. B., Van Horn E. C.* Programming semantics for multiprogrammed computations. Comm. ACM, vol. 9, 1966, No. 3.
66. *Devy M., Diaz M.* Multilevel specification and validation of the control in communication systems. Proc. 1st Int. Conf. Distributed Computing Systems, 1979.
67. *Dijkstra E. W.* Cooperating sequentaial processes. In Programming languages, Academic Press, N. Y., 1968.
68. IEEE Trans. Commun. Special issue on computer network architectures and protocols. COM-28, No 4, 1980.
69. *Fischer P. C., Meyer A. R., Rosenberg A. L.* Counter machines and counter languages. Math. Syst. Theory, vol. 2, 1968, No 3.
70. *Foo S. Y., Musgrave G.* Comparison of graph models for parallel computation and their extention. 1975 Int. Symp. Comput. Hardware Description Languages and their Applications Proc., N. Y., 1975.
71. *Furtek F.* Modular implementation of Petri nets. MS Thesis. Dept. Electr. Eng., MIT, Cambridge, Mass., Sept. 1971.
72. *Furtek F. C.* The logic of systems. TR 170, Dept. Electr. Eng., MIT, Cambridge, June 1976.
73. *Genrich H. J.* Einfache nicht-sequentielle Prozesse. GMD, Berlinghoven, 1970.
74. *Genrich H. J.* Einfache nicht-sequentielle Prozesse, GMD, Nr 37, Bonn, 1971.
75. *Genrich H. J.* The Petri net representation of mathematical knowledge. GMD-ISF Internat Rep. 75-06, GMD, Berlin, Berlinghoven, 1975.
76. *Genrich H. J., Lautenbach K.* Synchronization Graphen. Acta Informatica, vol. 2, 1972.
77. *Genrich H. J., Lautenbach K.* The analysis of distributed systems by means of predicate / transition-nets. Lect. Notes Comput. Sci., vol. 70, 1979.
78. *Genrich H. J., Lautenbach K., Thiagarajan P. S.* Elements of general net theory. In [143].
79. *Genrich H. J., Lautenbach K., Thiagarajan P. S.* Substitution systems: a family of system models based on concurrency. In [164].
80. *Genrich H. J., Stankiewicz-Wiechuo E.* A dictionary of some basic notions of net theory. In [143].
81. *Ghosh S.* Some comments on timed Petri nets. J. AFCET sur les reseaux de Petri. Inst. Progr., Paris.
82. *Ginzburg A., Yoeli M.* Vector addition systems and regular languages. J. Comput. Syst. Sci., vol. 20, 1980, No 3.
83. *Gostelow K. P.* Flow of control, resource allocation and the proper termination of program. PhD Thesis, Comput. Sci. Dept., Univ. Calif., Los Angeles, Dec. 1971.
84. *Gostelow K. et al.* Proper termination of flow of control in programs involving concurrent processes. SIGPLAN Notices, vol. 7, 1972, No 11.
85. *Grabowski J.* On the analysis of switching circuits by means of Petri nets. EIK, vol. 14, 1978, No 12.
86. *Grabowski J.* On Hack's conjecture concerning reachability in Petri nets. EIK, vol. 15, 1979, No 7.
87. *Habermann A. N.* Synchronization of communicating processes. Comm. ACM, vol. 15, 1972.
88. *Hack M.* Analysis of production schemata by Petri nets. MAC TR-94, Proj. MAC, MIT, Cambridge, Mass., Feb. 1972. Corrections: Comput. Structures Note 17, June 1974.
89. *Hack M.* A Petri net version of Rabin's undecidability proof for vector addition systems. CSG Memo. 94, Proj. MAC, Cambridge, MIT, Mass., Dec. 1973.
90. *Hack M.* Decision problems for Petri nets and vector addition systems. CSG Memo. 95, Proj. MAC, MIT, Cambridge Mass., March, 1974.
91. *Hack M.* The recursive equivalence of the reachability problem and vector addition systems. Proc. 15th Ann. Symp. Switching and Automata, IEEE, N. Y., 1974.
92. *Hack M.* The equality problem for vector addition systems is undecidable. Theory Comput. Sci., vol. 2, 1976, No 1.
93. *Hack M.* Petri net languages. TR 159, MIT, Cambridge, Mass., June 1976.
94. *Hack M.* Decidability questions for Petri nets. TR 161, MIT, Cambridge, Mass., 1976.
95. *Hack M.* Petri net languages. CSG Memo. 124, Proj. MAC, MIT, Cambridge, Mass., June 1975.
96. *Han Y. W., Heimerdinger W. L.* Transformation a Petri net-like labelled graph to a directed graph for design error detection. Proc. FTCS-7, June 1977.
97. *Hansal A., Schwab G. M.* On marked graphs III. Rep. LN 25.6.038, IBM Vienna Labs, Vienna, Sept. 1972.
98. *Henhapl W.* Firing sequences of marked graphs. Rep. LN 25.6.023, IBM Vienna Labs., Vienna, June 1972.

99. *Henhapl W.* Firing sequences of marked graphs II. Rep. LN 25.6.036, IBM Vienna Labs., Vienna, June 1972.
100. *Herzog O.* State analysis of concurrent processes for dynamic properties using Petri nets. Lect. Notes Comput. Sci., vol. 70, 1979.
101. *Holt A. W., Saint H., Shapiro R. M., Warshall S.* Final report of the information system theory project. TR RADC-68-305, Rome Air Devel. Center, Griffits Air Force Base, N. Y., Sept. 1968.
102. *Holt A. W., Commoner F.* Events and conditions. Rec. Proj. MAC Conf. Concurrent Systems and Parallel Computation, ACM, N. Y., 1970.
103. *Holt R. C.* On deadlock in computer system. PhD Thesis, Dept. Comput. Sci, Cornell Univ., Ithaca, N. Y., Jan. 1971.
104. *Izbicki H.* On marked graphs. Rep. LR 25.6.023, IBM Vienna Labs., Vienna, Sept. 1971.
105. *Izbicki H.* On marked graph II. Rep. LN 25.6.029, IBM Vienna Labs., Vienna, Jan. 1972.
106. *Jack L.* Graphical representation for fault tolerant phenomena. Proc. Seminar Dept. Electr. Eng., Univ. Texas, Austin, Jan. 1976.
107. *Janicki R.* Synthesis of concurrent schemes. Lect. Notes Comput. Sci., vol. 64, 1978.
108. *Janicki R.* On atomic nets and concurrency relation. In [143].
109. *Jantzen M.* On the hierarhy of PN languages. RAIRO Ifn. Theory, vol. 13, 1979, No 1.
110. *Jantzen M.* Structured representation of knowledge by Petri nets as an aid for reaching and research. In [164].
111. *Jantzen M., Valk R.* Formal properties of place / transition nets. In [164].
112. *Jensen K.* A method to compare the descriptive power of differen types of Petri nets. In [143].
113. *Jensen K.* Colloured Petri nets and the invariant method. Theor. Comput. Sci., vol. 14, 1981, No 3.
114. *Jones N. D., Landweber L. H., Lien Y. E.* Complexity of some problems in Petri nets. Theor. Comput. Sci., 1977, No 4.
115. *Jotwani N. D., Jump J. R.* Top-down design in the context of parallel programs. Inf. and Control., vol. 40, 1979, No 3.
116. *Jump J. R., Thiagarajan P. S.* On the equivalence of asynchronous control structures. SIAM J. Comput., vol. 2, 1973, No 2.
117. *Jump J. R., Thigarajan P. S.* On the interconnection of asynchronous control structures. J. ACM, 1975, vol. 22, No 4.
118. *Karp R. M., Miller R. E.* Properties of a model for parallel computations: determinacy, termination, queueing. SIAM J. Appl. Math., vol. 14, 1966, No 6.
119. *Karp R. M., Miller R. E.* Parallel program schemata, J. Comput. Syst. Sci., vol. 3, 1969, No 4.
120 *Kasami T., Tokura N., Peterson W. W.* Vector addition systems and synchronization problems of concurrent processes. Draft manuscript, 1974.
121. *Keller R. M.* Parallel program schemata and maximal parallelism. J. ACM, vol. 20, 1973.
122. *Keller R. M.* Vector replacement systems: a formalism for modelling asynchronous systems. TR 117, Comput. Sci. Lab., Princeton Univ., Princeton, N. J., Dec. 1972.
123. *Keller R. M.* A fundamental theorem of asynchronous parallel computation. Proc. Sagamore Comput. Conf. Parallel Processing, Springer — Verlag, Berlin, 1975.
124. *Keller R. M.* Generalized Petri nets as a model for system verification. TR 202. Dept. Electr. Eng., Princeton Univ., Princeton, N. Y., Aug. 1975.
125. *Keller R. M.* Look-ahead processors. Comput. Surveys, vol. 7, 1975, No 4.
126. *Keller R. M.* Formal verification of parallel programs. Comm. ACM, vol. 19, 1976, No 7.
127. *Kosaraju S. R.* Limitations of Dijkstra's semaphore primitives and Petri nets. Oper. Syst. Review, vol. 7, 1973.
128. *Kotov V. E.* An algebra for parallelism based on Petri nets. Lect. Notes Comput. Sci., vol. 64, 1978.
129. *Kreowski H. J.* A comparison between Petri nets and graph grammars. Lect. Notes Comput. Sci., vol. 100, 1981.
130. *Kwong Y. S.* On the absence of livelocks in parallel programs. Lect. Notes Comput. Sci., vol. 70, 1979.
131. *Landweber L. H., Robertson E. L.* Properties of conflict-free and persistent Petri nets. J. ACM, vol. 25, 1978, No 3.
132. *Lauer P. E., Cambell R. H.* A description of path expression by Petri nets. TR 64, Comput. Lab., Univ. Newcastle Upon Tyne, May 1974.
133. *Lauer P. E.* Path expression as Petri nets, or Petri nets with fewer tears. MRM 70, Comput. Lab., Univ. Newcastle Upon Tyne, Jan. 1974.
134. *Lautenbach K.* Exakte Bedingungen der Lebendigkeit fur eine Klasse von Petri-Netzen. Berichte der GMD Nr. 82, Bonn, 1973.
135. *Lautenbach K.* Liveness in Petri nets. GMD Internal Rep. ISF-75-021, Bonn, July 1975.
136. *Lautembach K.* Wegsysteme in Petri-Netzen. Ber. Ges. Math. und Datenerar., 111, 1979.

137. *Lautenbach K., Schmid H. A.* Use of Petri nets for proving correctness of concurrent process systems. Proc. IFIP Congress 74, North-Holland, Amsterdam, 1974.
138. *Ledanois P., Moalla M., Saucier G., Sifakis G., Zachariades M.* Multilevel description and simulation of parallel cooperating processors. Arbeitsberichte des IMMD, Band 9, Heft 8.
139. *Lien Y. E.* Termination properties of generalized Petri nets. SIAM J. Comput., vol. 5, 1976, No 2.
140. *Lipton R.* The reachability problem and the boundedness problem for Petri nets are exponential-space hard. TR 62, Dept. Comput. Sci., Yale Univ., New Haven, Conn., Jan. 1976.
141. *Lipton R. J., Miller R. E., Snyder L.* Synchronization and computing capabilities of linear asynchronous structures. J. Comput. Syst. Sci., vol. 14, 1977, No 1.
142. *Lipton R. J., Snyder L., Zalcstein Y.* A comperative study of models of parallel computation. Proc. 15th Ann. Symp. Switching and Automata, IEEE, N. Y., 1974.
143. Mathematical Foundations of Comput. Science 1980. Proc. 9th Symp. P. Dembin'ski (Ed.). Lect. Notes in Comput. Sci., vol. 88, 1980.
144. *Mayr E. W.* The complexity of the finite containment problem for Petri nets. TR 181, Lab. Comput. Sci., MIT, Cambridge, Mass., 1977.
145. *Mazurkiewicz A.* Parallel recursive program schemes. Proc. Symp. MFCS 75, Lect. Notes Comput. Sci., vol. 32, 1975.
146. *Memmi G.* Semiflows and invariants. Applications in Petri nets theory. J. sur les réseaux de Petri, Paris, March 1977.
147. *Memmi G.* Fultes dans les réseaux de Petri. RAIRO Inf. Theor. Comput. Sci., vol. 12, 1978.
148. *Memmi G., Roucairol G.* Linear algebra in net theory. In [164].
149. *Meldman J. A., Holt A. W.* Petri nets and legal systems. Jurimetrics J., vol. 12, 1971, No 2.
150. *Merlin P. A.* A study of recoverability of computing systems. PhD Thesis, Dept. Inf. and Comput. Sci., Univ. Calif., Irvine, 1974.
151. *Merlin P. M., Farber D. J.* Recoverability of communication protocols: Implementations of a theoretical study. IEEE Trans. Comm., COM-24, 1976, No 9.
152. *Miller R. E.* A comparison of some theoretical models of paralel computation. IEEE Trans. Comput., C — 22, 1973, No 8.
153. *Miller R., Yap Chee K.* On formulating simultaneity for studying parallelism and synchronization. J. Comput. Syst. Sci., vol. 20, 1980, No 2.
154. *Misunas D.* Petri nets and speed independent design. Comm. ACM, vol. 16, 1973, No 8.
155. *Moalla M., Pulou J., Sifakis J.* Résaux de Petri synchronises. RAIRO Automatique, 12, 1978.
156. *Moalla M., Saucier G., Sifakis G., Zachariades M.* A design tool for the multilevel description and simulation of systems of interconnected modules. 3rd Ann. Symp. Comput. Architecture, Tampa, Fla., Jan. 1979.
157. *Mogens N., Plotkin G., Winskel G.* Petri nets, event structures and domains. Lect. Notes Comput. Sci., vol. 70, 1979.
158. *Muller D. E.* Lecture notes on asynchronous circuit theory. Dig. Comput. Lab., Univ. Illinois, 1971.
159. *Müller H.* Decidability of reachability in persistent vector replacement systems. In [143].
160. *Müller H.* Reachability analysis with assertion systems. Lect. Notes Comput. Sci., vol. 104, 1981.
161. *Muller D. E., Bartky M. S.* A theory of asynchronous circuits. Proc. Int. Symp. Theory of Switching, Harward Univ. Press, Cambridge, Mass., 1959.
162. *Murata T., Church R. W.* Analysis of marked graphs and Petri net by matrix equations. Res. Rep. MDC 1.1.8, Dept. Inf. Ing., Univ. Illinois, Chicago Circle, Nov. 1975.
163. *Nash B. O.* Reachability problems in vector addition systems. Amer. Math. Monthly, vol. 80, 1973, No 3.
164. Net theory and apllications. Proc. of the Advanced Course on General Net Theory of Processes and Systems, Hamburg, 1979.
165. *Noe J. D.* A Petri net model of the CDC 6400. Proc. ACM/SIGOPS Workshop on System Performance Evaluation, ACM, N. Y., 1971.
166. *Noe J. D.* Nets in modelling and simulation. In [164].
167. *Noe J. D.* Abstractions of net models. In [164].
168. *Noe J. D.* Application of net-based models. In [164].
169. *Noe J. D., Nutt G. J.* Macro E-nets for representation of parallel systems. IEEE Trans. Comput., C — 22, 1973, No 8.
170. *Nutt G. J.* Evaluation nets for computer system performance analysis. AFIPS Conf. Proc., vol. 41, pt. 1, 1972.
171. *Oberquelle H.* Nets as a tool in teaching and terminology work. In [164].
172. *Parikh R. J.* On context-free languages. J. ACM, vol. 13, 1966.
173. *Patil S. S.* Coordination of asynchronous events. PhD Thesis, Dept. Electr. Eng., MIT, Cambridge, Mass., May 1970.
174. *Patil S. S.* Limitations and capabilities of Dijkstra's semaphore primitives for coor-

dination among processes. CSG Memo. 67, Proj. MAC, MIT, Cambridge, Mass., Feb. 1971.
175. *Patil S. S.* Circuit implementation of Petri nets. CSG Memo. 73, Proj. MAC, MIT, Cambridge, Mass., Dec. 1972.
176. *Patil S. S., Dennis J. B.* The description and realization of digital systems. RAIRO, 1973, No 1.
177. *Patil S. S.* An asynchronous logic array. Techn. Memo. 62. Proj. MAC, MIT, Cambridge, Mass., 1975.
178. *Peterson J. L.* Modelling of parallel systems. PhD Thesis, Dept. Electr. Eng., Stanford Univ., Calif., Dec. 1973.
179. *Peterson J. L.* Computation sequence sets. J. Comput. Syst. Sci., vol. 13, 1976, No 1.
180. *Peterson J. L.* Petri nets. Comput. Surveys, vol. 9, 1977, No 3.
181. *Peterson J. L.* Petri net theory and the modelling of systems, Prentice-Hall, 1981.
182. *Peterson J. L., Bredt T. H.* A comparison of models of parallel computation. Proc. IFIP Congress 74, North-Holland, Amsterdam, 1974.
183. *Petri K. A.* Kommunikation mit Automaten. Schriften für des Rheinisch — Westfalischen Institutes für Instrumentelle Mathematik an der Universitat Bonn, Heft 2, Bonn, 1962.
184. *Petri C. A.* Fundamentals of a tehory of asynchronous information flow. Proc. IFIP Conf., Munich, 1962.
185. *Petri C. A.* Netztopoloqie. Proc. des. Sommerseminars System — Organisation, GMD, 1972.
186. *Petri C. A.* Concepts of net theory. Proc. Symp. and Summer School on Mathematical Foundations of Computer Science, High Tatras, 1973, Math Inst. Slovak Academy of Sci., 1973.
187. *Petri C. A.* Interpretations of net theory. Interner Bericht 75—07, GMD, Bonn July 1975.
188. *Petri C. A.* General net theory. Proc. Jt. IBM Seminar, Univ. Newcastle Upon Tyne, Sept. 1976.
189. *Petri C. A.* Introduction to general net theory. In [164].
190. *Petri C. A.* Concurrency. In [164].
191. *Prószynski P.* Petri nets and concurrency-like relations. Lect. Notes Comput. Sci., vol. 107, 1981.
192. *Rackoff C.* The covering and boundness problems for vector addition systems. TR 97, Dept. Comput. Sci., Univ. Toronto, Toronto, July 1976.
193. *Ramamoorthy C. V., Gary S. H.* Performance evaluation of asynchronous concurrent systems using Petri nets. IEEE Trans. Software Ing., vol. 6, 1980, No 5.
194. *Ramchandani C.* Analysis of asynchronous concurrent systems by timed Petri nets. PhD Thesis, MIT, Cambridge, Mass., Sept. 1973.
195. *Renalier J.* Analyse et simulation in language de systemes de commande descrits par les réseaux de Petri. Thesis doct., Univ. P. Sabatier, Toulouse, 1977.
196. *Riddle W. E.* The modelling analysis of supervisory systems. PhD. Thesis, Stanford Univ., Stanford, Calif., March 1972.
197. *Riddle M. E.* The equivalence of Petri nets and message transmition models. SRM 97, Univ. Newsactle Upon Tyne, Aug. 1974.
198. *Rodrigues J. E.* A graph model for parallel computation. PhD Thesis, Dept. Electr. Eng., MIT, Cambridge, Mass., Sept. 1967.
199. *Sacerdote G., Tenney R. L.* The decidability of the reachability problem for vector additon systems. COINS TR 77-3, Univ. Massachusetts, Mass.
200. *Schimagh H.* Use do Petri nets in operating system design. Pr. IPI PAN, 1980, No 411.
201. *Scroff R.* Vermeidung von tatalen Verklemmungen in bewerteten Petri-Netzen. Diss. TU München, 1974.
202. Semantics of concurrent computation. Lect. Notes Comput. Sci., vol. 70, 1979.
203. *Shapiro R. M., Saint H.* A new approach to optimization of sequencing decisions. Ann. Review of Automatic Programming, vol. 6, 1970, No 5.
204. *Shapiro R. M.* The application of generl net theory — a personal history. In [164].
205. *Shapiro S. D.* A stochastic Petri net with application to modelling occupancy times for concurrent task systems. Networks, vol. 9, 1979, No. 4.
206. *Shaw A.* The logical design of operating systems. Prentice Hall, Englewood Cliffs, 1974.
207. *Sifakis J.* Etude du comporment permanent des réseaux de Petri temporisés. J. sur les R réseaux de Petri, Paris, March, 1977.
208. *Sifakis J.* Use of Petri nets for performance evaluation. 3rd Int. Symp. Modelling and Performance Evaluation of Comput. Syst., Bonn, Oct. 1977.
209. *Sifakis J.* Homomorphisms of Petri nets: application to the realization of foult tolerant systems. Rap. de recherche No 90, Inst. Nat Polytech. de Grenoble, Oct. 1977.
210. *Sifakis J.* Structural properties of Petri nets. Rap. de recherche No 102, Inst. Nat. Polytech. de Grenoble, Dec. 1977.
211. *Sifakis J.* Realization of fault-tolerant systems by coding Petri nets. J. Des. Autom. and Foult — Tolerant, vol. 3, 1979, No 2.
212. *Sifakis J.* Performance evaluation of systems using nets. In [164].

213. *Slutz D. R.* The flow graph schemata model of parallel computation. PhD Thesis, Dept. Electr. Eng., MIT, Cambridge, Mass., Sept. 1968.
214. *Stutzki G.* Descriptional complexity of concurrent processes. In [143].
215. *Starke P. H.* Semilinearity and Petri nets. Math. Res., vol. 2, 1979.
216. *Suraj Z.* A resource allocation problem. In [143].
217. *Suzuki I., Murata T.* A method for hierarchically representing large scale Petri nets. Proc. IEEE Int. Conf. Circuits and Comput. ICCC80, Port Chester, N. Y., 1980, vol. 1, 2.
218. *Szlanko J.* Petri nets for proving some correctness properties of parallel program. Proc. IFAC/IFIP Workshop Real Time Program, Eindhoven, Oxford, 1977.
219. *Taniguchi K., Matsuura T., Sugiyama Y., Kasami T.* On equivalence of safe Petri nets. Math. Res., vol. 2. 1979.
220. *Thornton J. E.* Design of a computer: the Control Data 6600. Scott, Foresman and Co., Glenvieq, Ill., 1970.
221. *Toulotte J. M.* Réseaux de Petri et automares programmables. Automatisme, vol. 23, No 7, 8.
222. *Toulotte J. M., Parsey J. P.* A method for decomposition interpreted Petri nets and its utilization. Dig. Proc., vol. 5, 1979, No 3, 4.
223. *Tsichritzis D.* Modular system description. TR 33, Dept, Comput. Sci., Univ. Toronton, Toronto, Oct. 1971.
224. *Valette R.* Analysis of Petri nets by stepwise refinements J. Comput. Syst. Sci., vol. 18, 1979, No 1.
225. *Valette R., Dias M.* Top-down formal specification and varification of parallel control systems. Dig. Proc., vol. 4, 1978.
226. *Valk R.* Selfvarying nets. J. sur réseaux de Petri. Inst. Progr., Paris, Mars 1977.
227. *Valk R., Vidal-Naquet G.* On the rationality of Petri nets languages. Lect. Notes Comput. Sci., vol. 48, 1977.
228. *Van Leeuwen J.* A partial solution to the reachability problem for vector addition systems. Proc. 6th Ann. Symp. Theory of Computer. ACM, N. Y., 1974.
229. *Yoeli M., Barzilai Z.* Behaviour description of communication switching systems using extended Petri nets. Dig. Proc., vol. 4, 1977, No 3.
230. *Zuse K.* Petri nets from the engineer's viewpoint. In [164].

Ленинград

Поступила в редакцию
30.VII.1982