УДК 658.52.011.56

ГРУППОВОЕ УПРАВЛЕНИЕ В ДИСКРЕТНО-СОБЫТИЙНЫХ СИСТЕМАХ

А.А. Амбарцумян, А.И. Потехин

Под групповым управлением понимается способ управления конфигурацией автономных компонентов объекта в целях выполнения заданной технологической функции. Идея заключается в создании двухуровневой системы управления с использованием на нижнем уровне моделей поведения каждого компонента, а групповое управление реализуется моделью управляющего компонента, взаимодействующего с моделями нижнего уровня. Рассмотрены два типа группового управления: заданы последовательности совместного выполнения операций автономными компонентами, заданы ограничения на сочетание технологических операций, выполняемых различными компонентами.

Ключевые слова: групповое управление, дискретно-событийные модели, сети Петри, автономные компоненты, структурно-сложные системы.

ВВЕДЕНИЕ

Групповое управление (ГУ) технологическим оборудованием как способ управления конфигурацией автономных компонентов объекта (*структурой потоков материалов и изделий*) в целях выполнения заданной технологической функции (задачи) широко применяется в промышленности. В настоящее время ГУ выполняется на основе «жесткой схемы» конечно-автоматных моделей. Однако это несовместимо с объективно изменяющейся на сооружаемом объекте технологией, поскольку при сооружении объекта (5—7 лет после проекта), как правило, имеют место отклонения в поставляемом оборудовании от проекта и изменения в технологических процессах. Поэтому смена конфигурации оборудования закрепляется в регламенте, должностных инструкциях и неформальных действиях персонала с использованием дистанционного управления каждым исполнительным механизмом. Тем самым закладывается *чрезвычайно низкий уровень автоматизации управления*. Это служит причиной гипертрофированно высокой степени участия оператора в управлении, вследствие чего наблюдается рост потерь по причине человеческого фактора и как следствие снижение эффективности и безопасности системы. В настоящей статье предлагается подход к ГУ на основе дискретно-событийных моделей автономных компонентов (АК) объекта и заданных логических ограничений на их поведение. Основные концепты подхода: раздельное моделирование АК объекта, формирование требований к их совместному поведению, синтез управляющего компонента системы.

Подход к проектированию ГУ оборудованием с дискретным поведением, опирающимся на формальную модель объекта, на наш взгляд, состоит в дискретно-событийном моделировании, включая парадигму супервизорного управления в рамках теории дискретно-событийных систем (ДСС), которое активно развивается в ряде научных центров мира [1—7]. В работах А.А. Амбарцумяна [8, 9] исследованы так называемые «структурированные ДСС» в рамках триплета [G, K, S]: поставлена и решена задача формального синтеза «неблокирующего» супервизора S при компонентном задании объекта G и описании ограничений K на его компоненты.

В настоящей работе ставится задача разработать модель и методы формального синтеза ГУ технологическим оборудованием объекта при заданных моделях АК и требований (ограничений) к групповому управлению. Рассмотрены два типичных ограничения на формирование группового управления:

— ограничения K заданы в виде требуемой последовательности выполнения технологических операций АК;

— ограничения K заданы в виде допустимого сочетания технологических операций.

Наша модель группового управления предполагает сетевую систему, в которой модели нижнего уровня определяются функциональностью объекта, а компонент верхнего уровня, обеспечивающий групповое управление, будет ограничивать порядок выполнения операций, определяемый только назначением группы. Такое разделение позволяет предположить, что *локальные изменения в алгоритме функционирования отдельных АК будут приводить лишь к локальным изменениям управляющего компонента*.

1. БАЗОВЫЕ ПОНЯТИЯ И ПОСТАНОВКА ЗАДАЧИ

Модель ДСС, по нашему мнению, служит подходящей моделью для представления группового управления АК в виде двухуровневой структуры. Напомним, что ДСС — это триплет [G, K, S], где G — объект, K — спецификация (ограничение на поведение объекта), а S — супервизор (управляющий компонент ДСС), обеспечивающий поведение объекта G в соответствии с ограничениями K. Отличительная особенность данной работы состоит в моделировании поведения объекта G и супервизора S сетями Петри. Сеть Петри S определяется через структуру и разметку: $S = (P, T, Pre, Post)$, где P — множество из n позиций, T — множество из m переходов t_i (множества конечны, не пусты и в графической форме сеть S изображается двумя типами вершин — кружками и полочками). Для всякого перехода t_i по функциям инцидентности определяются множества его входных $pre(t_i)$ и выходных $post(t_i)$ элементов. Аналогично для позиций. Эти функции расширяют и на соответствующие подмножества множеств P и T. Вектор μ: $P \to N$ (вектор разметки сети) каждой позиции ставит в соответствие целое положительное число (при графическом представлении сети Петри разметка позиции представляется точками в кружке, которые называют метками). Число меток в позиции p_j обозначается как $\mu(p_j)$. Сеть Петри — это пара (S, μ_0), где μ_0 — начальная разметка. В работе используются как традиционный способ, так и способ задания сети Петри матрицей инцидентности (различий) W, т. е. $S = (P, T, W, \mu_0)$. Терминология и нотация по сетям Петри заимствована из работ [4, 5].

Уточним постановку задачи группового управления: объект G задан совокупностью сетей Петри всех его АК, спецификация K определена в виде множества ограничений на совместное поведение компонентов. Далее рассмотрим два типичных ограничения:

— заданы последовательности совместного выполнения операций автономными компонентами объекта;

— заданы ограничения на сочетание технологических операций, выполняемыми различными компонентами.

Требуется синтезировать алгоритм группового управления, т. е. построить супервизор S в виде сети Петри.

Первый шаг конструирования группового управления состоит в моделировании всех компонентов объекта G сетями Петри. Как показано в работе А.А. Амбарцумяна [8], для решения задачи ГУ модель каждого компонента можно предварительно упростить путем сжатия всех состояний, не участвующих во взаимодействии с другими компонентами, в одно условно-подготовительное состояние, и таким образом получить *упрощенную* модель компонента, содержащую только состояния активности: «работа», «не работа» и соответствующие события.

2. СИНТЕЗ АЛГОРИТМА ГУ ПРИ ЗАДАННЫХ ОГРАНИЧЕНИЯХ НА ПОСЛЕДОВАТЕЛЬНОСТИ ОПЕРАЦИЙ

Теоретические основы данного параграфа изложены в работах А.А. Амбарцумяна [8, 9]. Пусть ДСС — это триплет [G, K, S]. Объект $G = [G^1, G^2, ..., G^n]$ — это совокупность автономных компонентов, которые будем моделировать автоматными сетями Петри. Каждая сеть Петри — это совокупность позиций, переходов, входных и выходных дуг и начальной разметки. Напомним, все переходы сети сопоставляются событиям и классифицируются на неуправляемые, ожидаемые и управляемые (последние на рисунках будем изображать полочками).

Спецификация K — это множество строк, определяющих требуемую последовательность срабатывания переходов АК. Требуется синтезировать ГУ в виде сети Петри супервизора S_c такой, чтобы объединенная сеть $S = [S_p, S_c]$ выполняла бы только те последовательности переходов, которые определены в спецификации K.

Пусть $K = [u_1, u_2, ...]$ — это язык, заданный строкам u, где каждая u_i-я последовательность срабатывания переходов в объекте G. Например, последовательность $u = t_1, t_2, t_5, t_6, t_3, t_4, t_7, t_8, t_1$ соответствует заданному порядку открытия и закрытия компонентов (например, задвижек трубопровода), рис. 1.

Определение 1. Пара соседних переходов (t_i, t_j) в строке u называется несвязной парой (*н-парой*), если в сети S объекта не существует позиции p_k такой, что $t_i \in pre(p_k)$ и $t_j \in post(p_k)$. ♦

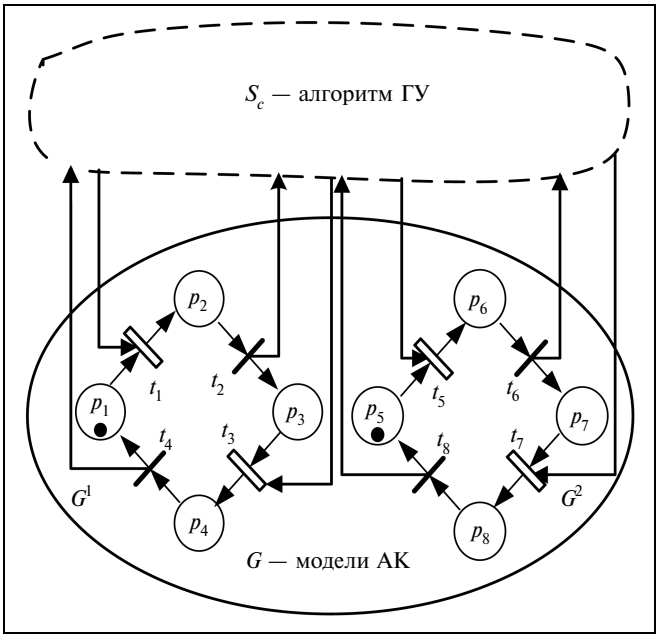

Рис. 1. Объединенная сеть $S = [S_p, S_c]$

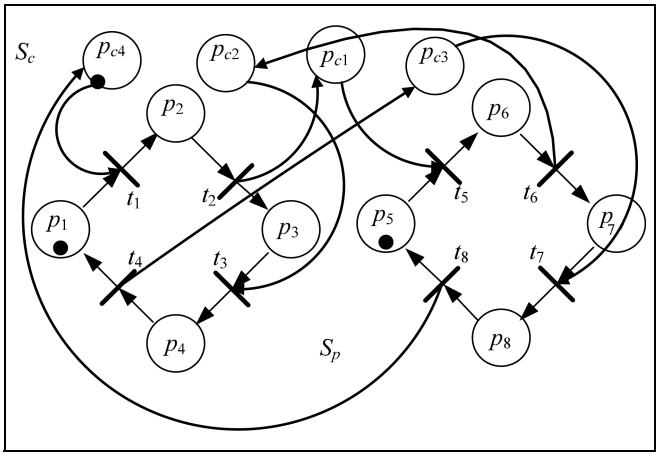

Рис. 2. Результирующая сеть $S = [S_p, S_c]$

Вначале кратко изложим базовый метод синтеза алгоритма ГУ, когда ограничение задано одной бесповторной строкой u срабатывания переходов [8]. Согласно определению 1 находим множество *н-пар* переходов в заданной строке u: (t_2, t_5), (t_6, t_3), (t_4, t_7), (t_8, t_1). Заметим, что второй переход в каждой *н-паре* является управляемым.

Далее, каждой *н-паре* переходов (t_i, t_j), в сети S_c строится структура (t_i, p_c, t_j), где p_c — позиция супервизора S_c, а переходы $t_i \in pre(p_c)$ и $t_j \in post(p_c)$. В нашем случае имеем (t_2, p_{c1}, t_5), (t_6, p_{c2}, t_3), (t_4, p_{c3}, t_7), (t_8, p_{c4}, t_1).

Результатом работы метода служит сеть S_c из четырех управляющих позиций $(p_{c1}, ..., p_{c4})$ и соответствующих связей. Результирующая сеть, состоящая из сети S_c и исходной сети S_p объекта, представлена на рис. 2.

В сети S_c определим ее начальное состояние. Нетрудно видеть, что метка начального состояния сети S_c должна находиться в позиции p_{c4}. Как и декларировалось ранее, при проектировании алгоритма ГУ на первом уровне сохраняются алгоритмы управления автономными компонентами (сеть S_p), а на втором уровне сеть S_c, содержащая позиции $p_{c1}, ..., p_{c4}$, реализует алгоритм ГУ. В общем случае имеет место многократность и вариантность срабатывания механизмов рассредоточенного объекта, что приводит к многообразию вариантов взаимодействия компонентов. Это отражается в типичной логической схеме последовательности строк u: $= \downarrow^{m1} t_k t_l ... t_k t_r \downarrow^{m2} t_j t_r ... t_l \{t_{a1} \uparrow^{m1}, ..., t_{an} \in \uparrow^{m2}\}$, в которой возможны различные *н-пары*: с совпадающими первыми (например $t_k t_l$ и $t_k t_r$) или вторыми событиями ($t_k t_r$ и $t_j t_r$) (символы событий заменены символами переходов). В работе [9] разработан метод синтеза алгоритма ГУ для общего случая. Изложим основные идеи метода. Кратко метод состоит из двух этапов. На первом из них по всем строкам спецификаций выявляются *н-пары* и конструируется матрица запусков *MF*. Ее строки помечены переходами, являющимися первыми компонентами в *н-парах*, а столбцы помечены переходами, являющимися вторыми компонентами в *н-парах*. Далее $MF(j, i) = 1$, если t_j, t_i — *н-пара*, иначе $MF(j, i) = 0$. На втором этапе для каждой строки матрицы запусков *MF* формируется звено передачи управления (типовой подсети в зависимости от типа *н-пар*). Эти звенья преобразуются в типовые расширения матрицы $W(n \times m)$ — матрицы инцидентности сети процесса S_p. Типовые подсети в зависимости от типа *н-пар* изображены на рис. 3 и 4.

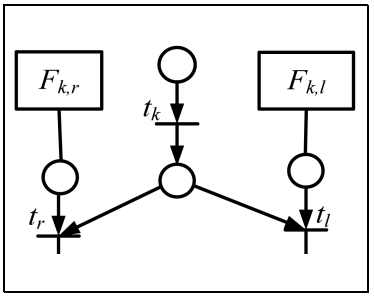

Рис. 3. Входная часть сети S_c для *н-пар* типа $t_k t_l$ и $t_k t_r$

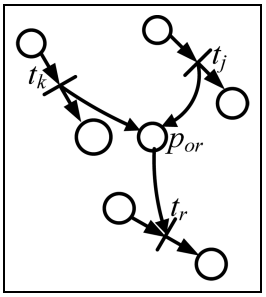

Рис. 4. Выходная часть сети S_c для *н-пар* типа $t_k t_r$ и $t_j t_r$

Так на рис. 3 изображена входная часть структуры прямого взаимодействия (СПВ), т. е. структуры S_c, обеспечивающей срабатывание переходов в сети S_p в порядке, определенном *н-парой* с совпадающим первым событием ($t_k t_l$ и $t_k t_r$). Здесь $F_{k,l}$ и $F_{k,r}$ — выходы так называемого автомата — агента F. Агент F — автомат Мура, единичное значение этих выходов зависит от предыстории события p_k и всегда ортогональны. Подробности синтеза автомата F изложены в работе [9]. На рис. 4 изображена выходная часть СПВ для *н-пар* с совпадающими вторыми событиями ($t_k t_r$ и $t_j t_r$). В этом случае в СПВ вводится дополнительная позиция p_{or}, выполняющая функцию сборки (операция «ИЛИ»).

3. СИНТЕЗ АЛГОРИТМА ГУ ПРИ ЗАДАННЫХ ОГРАНИЧЕНИЯХ НА СОЧЕТАНИЕ СОСТОЯНИЙ

Для решения задачи ГУ при ограничениях на сочетание состояний модель АК желательно упростить путем сжатия всех состояний, не участвующих в взаимодействии АК, в одно состояние и включить в упрощенную модель только состояния, соответствующие активности («работа» — «не работа») и соответствующие события.

В качестве примера рассмотрим упрощенный вариант пекарни — комплекс, состоящий из трех печей и двух роботов, обслуживающих выгрузку выпечки, рис. 5. (Этот пример — несколько модифицированная версия примера из работы [7]).

Ограничения на функционирование комплекса: все агрегаты работают циклически; любая печь, как только выпечка готова, должна немедленно

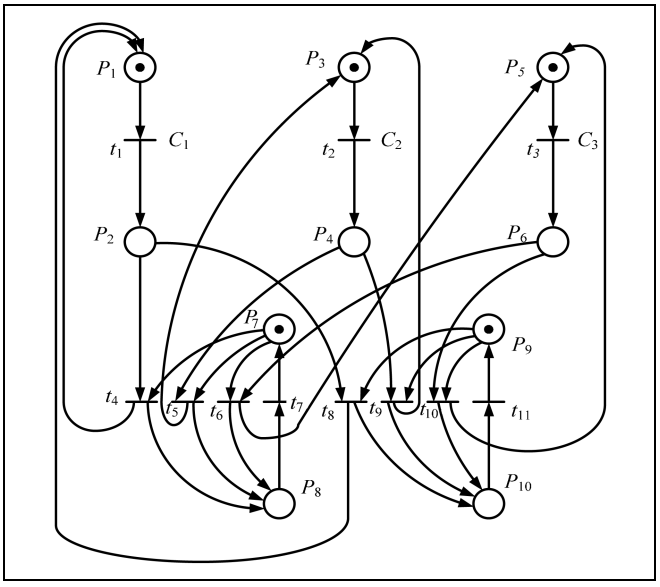

Рис. 5. Упрощенная модель пекарни

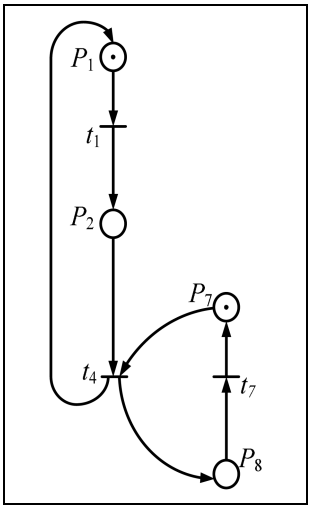

Рис. 6. Комплекс «одна печь — один робот» (П1 — Р1)

разгрузиться; печь разгружается любым свободным роботом. Работа печей представлена позициями: 1-я — p_1, p_2, 2-я — p_3, p_4, 3-я — p_5, p_6; робот 1 — p_7, p_8 робот 2 — p_9, p_{10}. Переходы t_1, t_2, t_3 управляемые и соответствуют командам (событиям) запуска соответствующей печи, переходы t_4, t_5, t_6 и t_8, t_9, t_{10} неуправляемые и выполняют функцию синхронизации работы роботов и печей.

Несмотря на простоту рассматриваемого комплекса, задача ГУ имеет достаточно общий характер. Вот одна из обобщающих формулировок задач. Дано две группы параллельно работающих агрегатов $A = \{a_i | i = \overline{1, n}\}$ и $B = \{b_i | i = \overline{1, k}\}$, $k < n$, таких, что для выполнения функционального назначения каждого $a_i \in A$ требуется его партнерство (коалиция) с любым из вспомогательных агрегатов (сервисов) $b_j \in B$. Задача группового управления заключается в динамическом (по мере готовности и запросов на группирование) формировании непересекающихся пар агрегатов, отвечающих требованиям к «партнерству».

Модель взаимодействия печей и роботов подробно рассмотрим на фрагменте комплекса «одна печь — один робот» (П1 — Р1), изображенного на рис. 6. Здесь позиция p_1 соответствует состоянию «ожидание» печи 1, переход t_1 — управляемый, его срабатывание переводит печь в позицию p_2 (загрузка и выпечка), переход t_4 — неуправляемый (ожидание выпечки), его срабатывание переводит печь в исходную позицию p_1 и одновременно переводит робот 1 в позицию p_8 (разгрузка печи). По окончании разгрузки — срабатывание перехода t_7 — робот переходит в позицию p_7. Функционирование данного фрагмента можно описать в виде таблицы переходов (табл. 1).

Таблица 1

Таблица переходов фрагмента П1–Р1

	t_1	t_4	t_7	p_1	p_2	p_7	p_8	p_1	p_2	p_7	p_8
1	0	0	0	1	0	1	0	1	0	1	0
2	1	0	0	1	0	1	0	0	1	1	0
3	0	0	0	0	1	1	0	0	1	1	0
4	0	1	0	0	1	1	0	1	0	0	1
5	0	0	0	1	0	0	1	1	0	0	1
6	0	0	1	1	0	0	1	1	0	1	0
7	0	0	0	1	0	1	0	1	0	1	0

В левой части табл. 1 (слева от двойной линии) представлено текущее состояние фрагмента, в правой части — следующее состояние. Значение перехода $t_i = 1$ означает срабатывание i-го перехода, значение позиции $p_j = 1$ означает нахождение метки в j-й позиции. Данную таблицу можно рассматривать как способ построения так называемого дерева достижимости (ДД), используемого в теории сетей Петри.

3.1. Основные понятия и определения

Рассмотрим некоторые понятия и определения, необходимые для дальнейшего построения супервизора при заданных ограничениях.

Определение 2. Состояние сети — это ее разметка μ: $P \to N$ (вектор разметки сети S) — каждой позиции, *ставящая в соответствие* целое положительное число.

Определение 3. Пусть $\mu(i)$ произвольная разметка сети с $P = \{p_1, ..., p_m\}$. Опорной функцией разметки $\mu(i) = \mathrm{Sup}(\mu(i))$ будем называть множество имен маркированных (имеющих метку) позиций в разметке $\mu(i)$. ♦

Для краткости значение $\mathrm{Sup}(\mu(i))$ будем называть состоянием s_i сети S (имеется в виду состояние при разметке $\mu(i)$). Например, если $P = \{p_1, ..., p_5\}$ и $\mu(i) = [011000]$, то $s_i = p_2p_3$. Очевидно, что s_i — это иное обозначение разметки $\mu(i)$.

Определение 4. Пусть M_R — множество достижимых маркировок сети S из μ_0, тогда $M_F \subset M_R$ называется множеством запрещенных (forbidden) состояний двух типов: состояний, противоречащих исходным ограничениям (устанавливаются разработчиком на основе технологического анализа объекта) или тупики (dead lock) и состояний, которые при возникновении неуправляемого события приводят в состояния уже отнесенные к запрещенным. ♦

В нашем примере каждый агрегат (печи и роботы) моделируются сетью Петри, содержащей всего две позиции — активная и пассивная (начальная). Позиции p_2, p_4, p_6, p_8, p_{10} — активные позиции, позиции p_1, p_3, p_5, p_7, p_9 — начальные. Исходя из технологического анализа функционирования объекта безусловно запрещенным состоянием сети S является состояние $s = p_2, p_4, p_6, p_8, p_{10}$, а также пять состояний, содержащих по четыре активных позиции (например, состояние $p_1, p_4, p_6, p_8, p_{10}$ и т. д.), так как технологически невозможна одновременно выгрузка печей и их загрузка-выпечка.

В работах [6, 9] конструирование супервизора (контроллера) базируется на знании запрещенных состояний сети, получаемых по полному ДД. Это сопряжено, прежде всего, с необходимостью анализа всех достижимых состояний, а также с получаемой сложностью управляющего компонента ДСС, реализующего собственно алгоритм ГУ, прямо пропорциональной числу запрещенных состояний.

Вместе с тем появлению запрещенного состояния всегда предшествует факт другого события — срабатывание некоторого управляемого перехода, допустимого для состояния предшествующего запрещенному. В настоящей работе мы исследуем именно этот феномен.

В множестве разрешенных состояний $M_A = M_R \backslash M_F$, важным подмножеством являются граничные разрешенные состояния M_{BA} (border authorized states).

Определение 5. Пусть T_c — подмножество управляемых переходов. Тогда *граничные разрешенные состояния* M_{BA} — это подмножество допустимых состояний ($M_{BA} \subseteq M_A$), из которых имеется управляемый переход ($t_j \in T_c$) в запрещенные состояния M_F. ♦

Такие переходы t_j будем называть *граничными управляемыми переходами*. В нашем примере граничными управляемыми переходами являются t_1, t_2 и t_3.

3.2. Нахождение граничных разрешенных состояний

Обычно нахождение достижимых состояний (разрешенных и запрещенных) достигается путем построения полного ДД. Покажем, как найти множество разрешенных состояний, а также множество граничных разрешенных состояний M_{BA} без явного построения ДД. Кроме этого, разработаем процедуру нахождения логических условий с целью блокирования граничных управляемых переходов.

Обратимся к табл. 1, которая является полным описанием ДД фрагмента «печь 1 — робот 1» (П1 — Р1), ее можно упростить, оставив те строки, где имеется единичное значение какого-либо из переходов. Нетрудно видеть, что состояние p_1p_2

Таблица 2

Таблица переходов (П1–Р1)

	p_2	p_8	p_2	p_8
t_1	0	0	1	0
t_4	1	0	0	1
t_7	0	1	0	0

Таблица 3

Таблица переходов (П1–Р2)

	p_2	p_8	p_2	p_8
t_1	0	0	1	0
t_8	1	0	0	1
t_{11}	0	1	0	0

Таблица 4а

Таблица переходов (П2–Р1)

	p_2	p_8	p_2	p_8
t_1	0	0	1	0
t_8	1	0	0	1
t_7	0	1	0	0

Таблица 4б

Таблица переходов (П2–Р2)

	p_4	p_{10}	p_4	p_{10}
t_1	0	0	1	0
t_8	1	0	0	1
t_{11}	0	1	0	0

Таблица 5а

Таблица переходов (П3–Р1)

	p_6	p_8	p_6	p_8
t_3	0	0	1	0
t_6	1	0	0	1
t_7	0	1	0	0

Таблица 5б

Таблица переходов (П3–Р2)

	p_6	p_{10}	p_6	p_{10}
t_3	0	0	1	0
t_{10}	1	0	0	1
t_{11}	0	1	0	0

Таблица 6

Варианты параллельных процессов

П1	П2	П3
Р1	Р2	0
Р2	Р1	0
0	Р1	Р2
0	Р2	Р1
Р1	0	Р2
Р2	0	Р1

может быть либо 01, либо 10. Поэтому в результирующей таблице вместо столбцов p_1p_2 можно оставить один столбец например, p_2: если $p_2 = 0$, то это означает, что метка находится в позиции p_1 ($p_1 = 1$) и наоборот. В результате таблица переходов (ТП) имеет вид табл. 2. Аналогично получаем ТП для печи 1 и робота 2 (П1 — Р2) (табл. 3), и ТП для фрагментов П2 — Р1 и П2 — Р2 (табл. 4а и 4б) и для фрагментов П3 — Р1 и П3 — Р2 (табл. 5а и 5б).

Из табл. 2—5 видно, что в позициях p_i, $p_j = 10$ или 01 соответствующий фрагмент объекта ($П_i$ — $Р_j$) находится в активном состоянии.

В рассматриваемом объекте возможно одновременное выполнение параллельных процессов, например, печь 1 обслуживается роботом 1, а печь 2 — роботом 2. Возможные варианты параллельных процессов представлены в табл. 6, где печь обозначена как П, робот — как Р, через «0» обозначен факт неработающей печи (одновременное обслуживание сразу трех печей двумя роботами невозможно).

Каждая строка табл. 6 соответствует некоторому варианту фрагментов, находящихся одновременно в активном состоянии. Например, в первой строке таблицы печь П1 с роботом Р1, а также печь П2 с роботом Р1 находятся в активных состояниях. При этом печь П3 должна быть «заблокирована», т. е. должна находиться в нерабочем состоянии. Из табл. 2 видно, что активные состояния фрагмента (П1 — Р1): $p_2\,p_8 = 01$ и $p_2\,p_8 = 10$, из табл. 4б — для фрагмента (П2 — Р2): $p_4\,p_{10} = 01$ и $p_4\,p_{10} = 10$. Получим всевозможные сочетания активных состояний этих фрагментов: $p_2\,p_4\,p_8\,p_{10} = (0011, 0110, 1001, 1100)$.

Аналогично из табл. 3 и 4а имеем активные состояния двух фрагментов (П1 — Р2) и (П2 — Р1): $p_2\,p_4\,p_8\,p_{10} = (0011, 0101, 1010, 1100)$.

Таким образом, мы получим множество граничных разрешенных состояний, в которых необходимо блокировать запуск печи П3 (путем блокирования перехода t_3): $p_2\,p_4\,p_8\,p_{10} = (1100, 1010, 1001, 0110, 0101, 0011)$.

Аналогичным образом по табл. 4 и 5 получаем множество граничных разрешенных состояний, в которых необходимо блокировать запуск печи П1 (путем блокирования перехода t_1): $p_4\,p_6\,p_8\,p_{10} = (1100, 1010, 1001, 0110, 0101, 0011)$.

Аналогично по таблицам 2, 3 и 5 — блокировать запуск для печи П2 (путем блокирования перехода t_2): $p_2\,p_6\,p_8\,p_{10} = (1100, 1010, 1001, 0110, 0101, 0011)$.

3.3. Представление множества граничных допустимых состояний булевыми формулами

Сопоставим позиции p_i логическую переменную x_i. Обозначим через f_i, $i = 1, 2, 3$, булеву функцию, принимающую единичное значение на множестве граничных разрешенных состояний, в которых необходимо блокировать запуск печи $П_i$:

$$f_1 = x_4x_6\overline{x_8}\,\overline{x_{10}} \vee x_4\overline{x_6}\,x_8\overline{x_{10}} \vee x_4\overline{x_6}\,\overline{x_8}\,x_{10} \vee \vee \overline{x_4}\,x_6x_8\overline{x_{10}} \vee \overline{x_4}\,x_6\overline{x_8}\,x_{10} \vee \overline{x_4}\,\overline{x_6}\,x_8x_{10};$$

$$f_2 = x_2x_6\overline{x_8}\,\overline{x_{10}} \vee x_2\overline{x_6}\,x_8\overline{x_{10}} \vee x_2\overline{x_6}\,\overline{x_8}\,x_{10} \vee \vee \overline{x_2}\,x_6x_8\overline{x_{10}} \vee \overline{x_2}\,x_6\overline{x_8}\,x_{10} \vee \overline{x_2}\,\overline{x_6}\,x_8x_{10};$$

$$f_3 = x_2x_4\overline{x_8}\,\overline{x_{10}} \vee x_2\overline{x_4}x_8\overline{x_{10}} \vee x_2\overline{x_4}\,\overline{x_8}x_{10} \vee$$
$$\vee\,\overline{x_2}x_4x_8\overline{x_{10}} \vee \overline{x_2}x_4\overline{x_8}x_{10} \vee \overline{x_2}\,\overline{x_4}x_8x_{10}.$$

Определим булеву функцию-импликацию, отражающую условие «не срабатывания» соответствующего граничного перехода. Вторая компонента импликации для перехода t_i определяется как инверсия булевой переменной, соответствующей позиции $post(t_i)$:

$$F_1 = f_1 \to \overline{x_2}; \quad F_2 = f_2 \to \overline{x_4}; \quad F_3 = f_3 \to \overline{x_6}.$$

Обобщенная функция «не срабатывания» граничных переходов F равна $F = F_1F_2F_3$.

По известной эквивалентности ($a \to b = \bar{a} \vee b$) переходим к ДНФ (дизъюнктивной нормальной форме) каждой функции, а затем выполним конъюнкцию этих функций и представим ее в совершенной конъюнктивной нормальной форме:

$$(\overline{x_2} \vee x_4 \vee x_6 \vee \overline{x_8} \vee \overline{x_{10}})(x_2 \vee x_4 \vee \overline{x_6} \vee \overline{x_8} \vee \overline{x_{10}}) \cdot$$
$$\cdot\,(\overline{x_2} \vee x_4 \vee \overline{x_6} \vee \overline{x_8} \vee x_{10})(\overline{x_2} \vee x_4 \vee \overline{x_6} \vee x_8 \vee \overline{x_{10}}) \cdot$$
$$\cdot\,(x_2 \vee \overline{x_4} \vee \overline{x_6} \vee x_8 \vee \overline{x_{10}})(x_2 \vee \overline{x_4} \vee \overline{x_6} \vee \overline{x_8} \vee x_{10}) \cdot$$
$$\cdot\,(\overline{x_2} \vee \overline{x_4} \vee \overline{x_6} \vee x_8 \vee x_{10})(\overline{x_2} \vee x_4 \vee x_6 \vee \overline{x_8} \vee \overline{x_{10}}) \cdot$$
$$\cdot\,(x_2 \vee \overline{x_4} \vee x_6 \vee \overline{x_8} \vee \overline{x_{10}}) \cdot$$
$$\cdot\,(\overline{x_2} \vee \overline{x_4} \vee x_6 \vee x_8 \vee \overline{x_{10}}) = F. \qquad (10)$$

3.4. Переход к линейным ограничениям

Сделаем несколько замечаний по поводу логических преобразований и обобщенной функции «не срабатывания» граничных переходов F.

- При выполнении перехода к ДНФ автоматически произошел переход к полностью определенным булевым функциям несрабатывания, поэтому в функцию F вошли и переходы из запрещенных состояний.
- Если $F \equiv 0$, то условия отдельных переходов несовместимы и не существует ни одного общего ограничения. Однако возможны всевозможные допустимые комбинации из отдельных функций.
- Соотношение нулей и единиц в корнях уравнения (1) показывает на соотношение меток в позициях сети, и, следовательно, может быть использовано для поиска линейных ограничений.

Для нашего примера функция F тождественно не равна нулю, следовательно, можно найти одно общее свойство, соответствующее ограничению на соотношение меток, и контроллер (супервизор) нашего примера будет содержать одну позицию, обозначим ее p_c. В соответствии с первым из сделанных замечаний, не все корни уравнения (1) являются приемлемыми решениями. Прежде всего, среди корней нет наборов, содержащих три единицы. Поскольку в наши ограничения входит требование, что всегда в один и тот же момент времени срабатывает один переход (что означает включение в работу одного агрегата), то решения с четырьмя и более единицами физически недостижимы. Поэтому приемлемыми решениями являются наборы значений $x_2\ x_4\ x_6\ x_8\ x_{10}$ для уравнения (1), у которых не более двух единичных значений переменных. Отсюда следует ограничение, задаваемое линейным неравенством на суммарное число меток в позициях $p_2\ p_4\ p_6\ p_8\ p_{10}$:

$$\mu(p_2) + \mu(p_4) + \mu(p_6) + \mu(p_8) + \mu(p_{10}) \leqslant 2. \qquad (2)$$

3.5. Синтез контроллера (супервизора)

Задачи синтеза состоят в:

— определении числа позиций в сети супервизора S_c;

— нахождении матрицы инцидентности W_c сети S_c;

— определении исходного состояния сети $S_c(\mu_{co})$.

Применим известную методику синтеза контроллера по линейным ограничениям, применяемую в школе P.J. Antsaklis [6].

Как было показано ранее, сеть S_c в нашем примере содержит одну позицию — p_c, т. е. $n_c = 1$.

Из работы [6] следует, что

$$W_c = -L \times W_p, \qquad (3)$$

где W_p — матрица инцидентности сети S_p. Матрица L размерности $(n_c \times n_p)$, в которой i-й разряд строки j содержит 1, если позиция p_i входит в ограничение. В нашем примере $n_c = 1$, $n_p = 10$, и в соответствии с формулой (2) матрица L имеет вид

1	2	3	4	5	6	7	8	9	10
0	1	0	1	0	1	0	1	0	1

Матрица W_p строится по исходной сети S_p и имеет вид

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}
1	−1	0	0	1	0	0	0	1	0	0	0
2	1	0	0	−1	0	0	0	−1	0	0	0
3	0	−1	0	0	1	0	0	0	1	0	0
4	0	1	0	0	−1	0	0	0	−1	0	0
5	0	0	−1	0	0	1	0	0	0	1	0
6	0	0	1	0	0	−1	0	0	0	−1	0

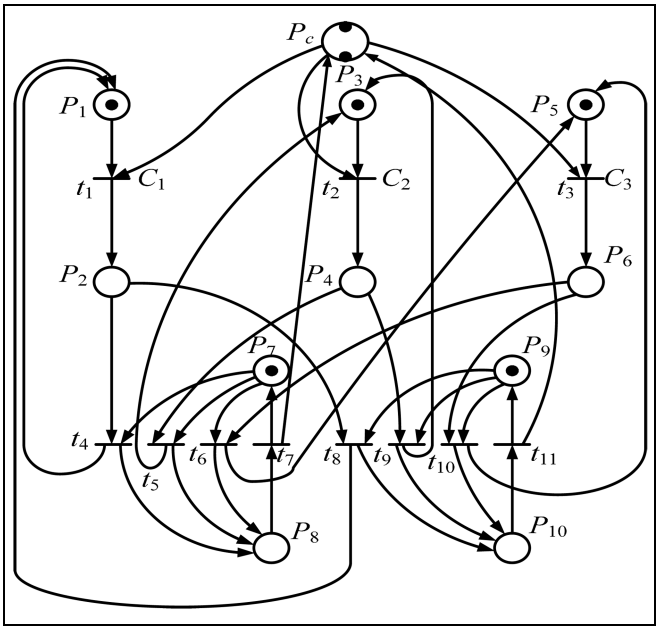

Рис. 7. Объединенная сеть $S = S_p \cup S_c$

В результате вычисления по формуле (3) получим матрицу W_c:

t	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}
P_c	−1	−1	−1	0	0	0	1	0	0	0	1

Исходное состояние μ_{c0} сети S_c определим по формуле $\mu_{p0} L + \mu_{c0} = b$, где $b = 2$ из формулы (4), $\mu_{p0} = 1010101010$.

Таким образом, $\mu_{c0} = 2 - L\,[1110001100] = 2$.

Формируем объединенную управляемую сеть $S = S_p \cup S_c$ (рис. 7).

ЗАКЛЮЧЕНИЕ

В смысле вклада данной работы в теорию дискретно-событийных систем, заметим, что предложенная техника синтеза группового управления по логическим функциям перехода оригинальна и содержит, на наш взгляд, значительный потенциал в плане развития формализмов в определении линейных ограничений по спецификации допустимых переходов.

В прикладном плане, в данной работе предложена модель и методы синтеза группового управления в рамках дискретно-событийной модели на сетях Петри. Принципиальные отличия группового управления в структуре этой модели от традиционной архитектуры реализации группового управления заключаются в возможности формально синтезировать алгоритм группового управления на базе более простых моделей объекта $G = [G^1, G^2, ..., G^n]$ как совокупности автономных компонентов и спецификации K в виде ограничений; а также в возможности конструирования двухуровневой архитектуры группового управления на базе модульности, определенной в модели. На первом уровне сохраняются алгоритмы управления автономными компонентами, соответствующие модулям $G^1, G^2, ..., G^n$, а на втором уровне — собственно алгоритм группового управления, формально синтезированный по дискретно-событийной модели объекта G и спецификации K.

ЛИТЕРАТУРА

1. *Ramadge J.G.*, *Wonham W. M.* Supervisory control of a class of discrete event processes // SIAM Journal of Control and Optimization. — 1987. — N 25. — P. 206—230.
2. *Cassandras C.G.*, *Lafortune S.* Introduction to discrete event systems. — Dordrecht: Kluwer, 1999.
3. *Holloway L.E.*, *Krogh B.H.* Synthesis of feedback logic for a class of controlled Petri nets // IEEE Trans. Autom. Control. — 1990. — AC-35. — N 5. — P. 514—523.
4. *Giua A., DiCesare F., Silva M.* Generalized mutual exclusion constraints on nets with uncontrollable transitions // Proc. IEEE Conf. Syst. Man, Cybern. — 1992. — P. 974—979.
5. *Giua A., and Seatzu C.* Supervisory control of railway networks with Petri nets // Proc. IEEE Conf. Decis. Contr. — 2001. — P. 5004—5009.
6. *Yamalidou K.*, *Moody J O.*, *Lemmon M.D.*, *Antsaklis P.J.* Feedback control of Petri nets based on place invariants // IEEE Trans. on Robotics and Automation. — 1996. — N 32 (1). — P. 15—28.
7. *Dideban A.*, *Alla H.* Reduction of Constraints for Controller Synthesis based on Safe Petri Nets // Automatica. — 2008. — N 44 (7). — P. 1697—1706.
8. *Амбарцумян А.А.* Моделирование и синтез супервизорного управления на сетях Петри для рассредоточенных объектов. Ч. 1. Механизм взаимодействия и базовый метод // Автоматика и телемеханика. — 2011. — № 8. — С. 151—169.
9. *Амбарцумян А.А.* Моделирование и синтез супервизорного управления на сетях Петри для рассредоточенных объектов. Ч. 2. Метод синтеза супервизора по множеству последовательностей общего вида // Автоматика и телемеханика. — 2011. — № 9. — С. 173—189.

Статья представлена к публикации членом редколлегии О.П. Кузнецовым.

Амбарцумян Александр Артемович — д-р техн. наук, зав. лабораторией,

Потехин Анатолий Иванович — канд. техн. наук, вед. науч. сотрудник, ☎ (495) 334-90-29, ✉ apot@ipu.ru,

Институт проблем управления им. В.А. Трапезникова РАН, г. Москва.