УДК 62-51

А.П. Верёвкин, Т.М. Муртазин

СИНТЕЗ ИЕРАРХИЧЕСКИХ КОНЕЧНО-АВТОМАТНЫХ УПРАВЛЯЮЩИХ СИСТЕМ

Верёвкин Александр Павлович, *доктор технических наук, профессор, окончил Уфимский нефтяной институт. Профессор кафедры «Автоматизация, телекоммуникация и метрология» Уфимского государственного нефтяного технического университета (УГНТУ). Имеет статьи, монографии, изобретения в области автоматизации управления технологическими процессами и производствами. [e-mail: apverevkin@mail.ru].*

Муртазин Тимур Мансурович, *кандидат технических наук, доцент, окончил УГНТУ. Доцент кафедры «Автоматизация, телекоммуникация и метрология» УГНТУ. Имеет статьи, монографии, изобретения в области автоматизации управления технологическими процессами и производствами. [e-mail: tm.murtazin@mail.ru].*

Аннотация

Применение известных методов синтеза логических выражений для управления исполнительными устройствами позиционного типа на основе автомата Мура и карт Карно для большого количества входных переменных и управлений встречает принципиальные трудности, обусловленные высокой размерностью задачи. Кроме того, задача построения автомата Мура для алгоритмов с «длинными» цепочками управлений является сложной с точки зрения обеспечения отсутствия тупиковых состояний и зацикливаний. В связи с этим задача синтеза конечно-автоматных управляющих устройств для задач высокой размерности должна решаться на основе принципов системного подхода декомпозицией исходной задачи на подзадачи. В статье предложены подход к формализации процедур формирования и последовательности выполнения логических выражений при вычислении управлений на основе применения иерархических сетей Петри и способ согласования работы подсетей на основе согласующих меток, который облегчает разработку логических управляющих устройств для объектов с большим количеством входных переменных и управлений.

Ключевые слова: конечный автомат, иерархические сети, сети Петри, логическое выражение, автоматное управление.

doi: 10.35752/1991-2927_2022_4_70_109

THE SYNTHESIS OF THE HIERARCHICAL FINITE-STATE CONTROL SYSTEMS

Aleksandr Pavlovich Verevkin, *Doctor of Sciences in Engineering, Professor; graduated from Ufa Oil Institute; Professor of the Department of Automation, Telecommunications and Metrology of Ufa State Petroleum Technological University; an author of articles, monographs, inventions in the field of process and production automation. e-mail: apverevkin@mail.ru.*

Timur Mansurovich Murtazin, *Candidate of Sciences in Engineering, Associate Professor; graduated from Ufa State Petroleum Technological University; Associate Professor at the Department of Automation, Telecommunications*

and Metrology of Ufa State Petroleum Technological University; an author of articles, monographs, inventions in the field of process and production automation. e-mail: tm.murtazin@mail.ru.

Abstract
The known methods of Boolean expression synthesis to control the positioning actuators based on the Moore machine and the Karnaugh maps for a large number of input variables and controls present crucial difficulties due to the high dimension of the task. Moreover, the task of constructing the Moore machine for algorithms with long control chains is a difficult one in terms of preventing lock-ups and loops. In this regard, the task of synthesis of finite-state machine devices in high-dimensional tasks should be solved on the basis of the principles of a systematic approach by partitioning the main task into subtasks. The article proposes a method for characterization of forming procedures and sequences of Boolean expressions for calculating controls based on the use of hierarchical Petri nets. It also proposes a method for matching the subnet operation based on matching tokens, which facilitates the development of logical control devices for objects with a large number of input variables and controls.

Keywords: finite-state machine, hierarchical nets, Petri nets, logical expression, automatic control.

Введение

Логические устройства и алгоритмы (ЛУ, автоматы) широко используются в следующих областях:

- математическая лингвистика;
- логическое управление;
- моделирование поведения человека;
- коммуникационные протоколы;
- теория формальных языков, вычислимости и вычислительной сложности;
- антивирусные программы и т. д.

Типичными задачами синтеза сложных конечно-автоматных моделей [1–12] являются алгоритмы пуска-останова технологических процессов, управление взаимосвязанными агрегатами, например, совокупностью насосов, агрегатов воздушного охлаждения, агрегатами транспортных систем и их сообществ.

Отдельное большое направление – это разработка и верификация программного обеспечения на основе конечно-автоматного представления работы взаимодействующих программ [13–18].

При синтезе систем управления сложными объектами, в которых используются дискретные булевы логические переменные, проблемой является получение логических моделей (логических последовательностей) в условиях большой размерности по входным логическим переменным.

Известные методы синтеза логических конечно-автоматных моделей базируются на так называемых автоматных моделях Мура и Мили [1–7]. Сложность процедуры синтеза по этим моделям растет экспоненциально от числа входных переменных и линейно от числа состояний моделей, и уже при числе переменных больше 5–6 процедуры синтеза становятся технически очень сложными. Поэтому единственным способом уйти от «проклятия размерности» является декомпозиция исходной задачи на подзадачи синтеза субавтоматов с их последующей интеграцией для достижения целей систем.

1 Процедура синтеза иерархических автоматов на основе сетей Петри

В [7–9, 11, 13 и др.] развивается подход к синтезу иерархических автоматных систем, базирующийся в основном на моделях автоматов Мура.

В данной статье рассматривается подход и метод его реализации для класса логических систем управления, которые взаимодействуют с окружающей средой путем обмена сообщениями в темпе, задаваемом средой («реактивные системы»). К этому классу относится подавляющее число систем управления промышленными объектами. Подход базируется на комплексном использовании сетей Петри, процедур декомпозиции на сетях Петри и трансформации сетей Петри в конечно-автоматные модели [7–9, 19, 20].

В отличие от работ [15–18], где применение алгоритмов предполагает использование языков программирования высокого уровня, в данной статье целевой задачей является реализация в контроллерах, где используются языки стандарта МЭК 1131-3. Поэтому результатом синтеза является получение совокупности логических выражений (логических последовательностей) в терминах булевой логики, которые легко реализуются на большинстве контроллеров.

Оценка возможностей языка сетей Петри позволяет заключить, что на его основе успешно может быть решена первая часть процедуры синтеза, т. е. формализация постановки задачи синтеза ЛУ в виде модели, проверка работоспособности синтезируемого ЛУ, а также вопросы декомпозиции модели на подсети для синтеза субавтоматов.

Некоторые трудности возникают в связи с тем, что в теории сетей Петри используются понятия, не адекватные понятиям теории конечных автоматов. В частности, позиции соответствует некоторое условие, причем метка означает истинность условия, а переходу соответствует некоторое действие, т. е. понятие «состояние» не используется.

При переходе от сети Петри к автомату Мура примем во внимание следующее. Активизация перехода t соответствует истинности логического условия, представляющего собой логическое произведение условий, соответствующих входным позициям перехода, а его срабатывание – изменению значений выходных переменных. Поэтому при переходе к автоматному представлению ЛУ удобно связать реализацию перехода с условием перехода, а совокупность входных позиций перехода – с состоянием автомата перед осуществлением перехода.

На основе изложенного схема синтеза ЛУ с декомпозицией задачи может быть следующей.

1. Строится «правильная» (живая, безопасная) сеть Петри, отвечающая условиям содержательной постановки задачи.

2. Из сети выделяются подсети, которые могут рассматриваться как автономные части. Координация работы подсетей и соответствующих им субавтоматов может осуществляться по **трем вариантам**:

2.1. Через использование входных логических переменных; при этом синтез проводится через модели в виде таблиц состояния автоматов Мура. Для каждой подсети количество состояний определяется количеством переходов, а количество входных переменных – набором переменных, входящих в логические условия срабатывания переходов данной подсети. Указываются также значения выходных переменных, определяемых состоянием данного автомата;

2.2. Для каждой подсети выделяется множество позиций, через которые подсеть связана с остальными подсетями (позиции связи), и позиции, направленные к данной подсети, включаются в состав входных позиций переходов, условия активизации которых связаны с другими подсетями. Синтез проводится с использованием укрупненных таблиц состояния (УТС) [7– 9].

2.3. Через использование вспомогательных логических переменных («флагов»), формируемых дополнительно к исполнительным командам на переходах, после активизации которых осуществляется выход из подсети, либо дополнительным (координирующим) субавтоматом [9, 11, 13, 16]. «Флаги» выполняют роль функций запрета-разрешения на активизацию отдельных переходов подсетей. Синтез проводится с использованием УТС.

Основная идея построения иерархических автоматных систем – это использование кодов состояний автоматов нижнего уровня на более высоких уровнях иерархии управления (координации) сообществом автоматов, которые выражаются через логические выражения активизируемых переходов.

3. Для каждой позиции через входные переменные ЛУ записывается логическое выражение, играющее роль условия срабатывания следующего за позицией перехода.

4. Для каждого перехода подсети записываются условия его срабатывания как логическое произведение условий, соответствующих входным позициям перехода, а также значения выходных переменных (исполнительных команд), связанных активизацией перехода.

5. Проводится синтез субавтоматов и «сборка» автомата либо через переменные, входящие в логические выражения позиций связи, либо через «флаги».

Синтез логических последовательностей с использованием УТС проводится следующим образом.

Для каждой подсети с использованием УТС записывается система логических последовательностей, связывающих условия возникновения ситуаций и выраженных через имена позиций $P_j\,(j=1,2,\ldots)$, и перечень активизируемых управлений в этих ситуациях, кодируемых через имена переходов.

Для подсети вида, рисунок 1, например.

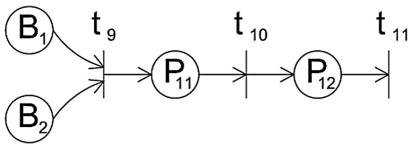

Рис. 1. Пример подсети

На рисунке 1: B_1, B_2 – «флаги», разрешающие работу субавтомата по условию $B_1 \Lambda B_2$ и активизирующие переход t_9 , P_{11}, P_{12} – позиции, представляющие собой некоторые условия, по которым происходит активизация переходов t_{10}, t_{11} соответственно, Λ – функция логического «И» .

УТС для подсети на рисунке 1 будет иметь вид (табл. 1).

Таблица 1

УТС для подсети на рисунке 1

№	$B_1 \Lambda B_2$	P_{11}	P_{12}	Активизация переходов, решения
1	2	1	1	-
2	2	3	2	t_9
3	3	3	4	t_{10}
4	$*$	$*$	4	t_{11}

Примечание:

1. Символ «$*$» означает, что переход не определен; уточнение проводится для конкретной сети.

2. Для обеспечения бесконфликтных переходов между состояниями (отсутствие состязаний) может потребоваться корректировка УТС.

Для получения модели в виде логических выражений (последовательностей) состояния кодируются промежуточными (вспомогательными) переменными $P_j\,(j=1,2,\ldots)$. Количество промежуточных переменных n определяется по числу кодируемых состояний N: $N \le 2^n$.

R_1 \ R_2		
	1	2
	4	3

Рис. 2. Таблица соответствия состояний-кодов

Для рассматриваемого примера N = 4, n = 2, коды состояний представлены в таблице соответствия (рис. 2).

Получение логических выражений для промежуточных переменных проводится аналогично тому, как это делается с использованием карт Карно [3, 4, 7]. Для каждой промежуточной переменной (функции) формируется таблица, в которой номера состояний заменены на коды; при этом номерам состояний внутри таблицы соответствуют коды соответствующей промежуточной функции (рис. 3).

Синтез логического выражения проводится известными методами, путем записи через конъюнктивную или дизъюнктивную нормальные формы [4, 7]. Так же, как и в картах Карно, для минимизации логических выражений можно формировать контуры («склеивать» ячейки таблицы). Например, используя дизъюнктивную форму записи логических выражений, получим:

$$R_1 = R_1 \vee (\mathrm{P}_{11} \wedge R_2) \vee (\mathrm{P}_{12} \wedge R_1),$$

где первый дизъюнкт формируется «склеиванием» (объединением в контур) шести единиц в строках, для которой R_1 = «1». Второй дизъюнкт формируется «склеиванием» двух единиц в столбце P_{11}, для которых R_2 = «1». Третий дизъюнкт формируется «склеиванием» двух единиц в столбце P_{12}, для которых R_1 = «1».

Таблица для R_1

R_2 R_1	$B_1 \wedge B_2$	P_{11}	P_{12}
	0	0	0
	0	1	0
	1	1	1
	*	*	1

Таблица для R_2

R_2 R_1	$B_1 \wedge B_2$	P_{11}	P_{12}
	1	0	0
	1	1	1
	1	1	0
	*	*	0

Рис. 3. Таблицы формирования логических выражений для промежуточных переменных

Аналогично для R_2:

$$R_2 = \overline{R}_1 \wedge R_2 \vee \mathrm{B}_1 \wedge \mathrm{B}_2 \vee R_1 \wedge \mathrm{P}_{11}.$$

Логические выражения для активизации переходов записываются как коды соответствующих состояний, в которых происходит активизация:

$$\mathrm{t}_9 = \overline{R}_1 \wedge R_2;\ \mathrm{t}_{10} = R_1 \wedge R_2;\ \mathrm{t}_{11} = R_1 \wedge \overline{R}_2.$$

Активизация переходов связывается с исполнительными действиями (командами на исполнительные устройства (ИУ)) через матрицу инициализаций вида (рис. 4).

	t_9	t_{10}	t_{11}
$К_1$	1	*	0
$К_2$	*	1	0
$К_3$	*	*	1

Рис. 4. Пример матрицы инициализаций («*» означает, что состояние ИУ не изменяется)

2 Пример синтеза логического управляющего устройства объектом с последовательными технологическими операциями

Подход и процедуру синтеза рассмотрим на примере схемы автоматической системы (рис. 5). Изображены три емкости E_1, E_2, E_3; ИУ: клапаны K_1, K_2, K_3, K_4, K_5, устройство включения мотора мешалки K_6. Входные переменные синтезируемого ЛУ: датчики уровня $\mathrm{Д}_1$, $\mathrm{Д}_2$, $\mathrm{Д}_3$, $\mathrm{Д}_4$, $\mathrm{Д}_5$. Значения датчиков «1» соответствуют достижению уровня расположения датчиков. $\mathrm{T}_\mathrm{м}$ – сигнал с таймера ($\mathrm{T}_\mathrm{м}$ = 1 соответствует включенному таймеру). Разрешение на запуск или останов цикла осуществляется тумблером $\mathrm{K}_\mathrm{н}$: при $\mathrm{K}_\mathrm{н}$ = 1 запускается цикл, при $\mathrm{K}_\mathrm{н}$ = 0 цикл останавливается.

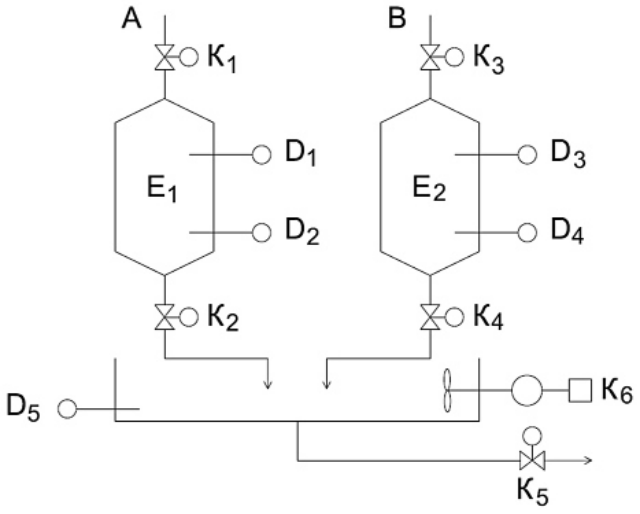

Рис. 5. Принципиальная схема объекта управления

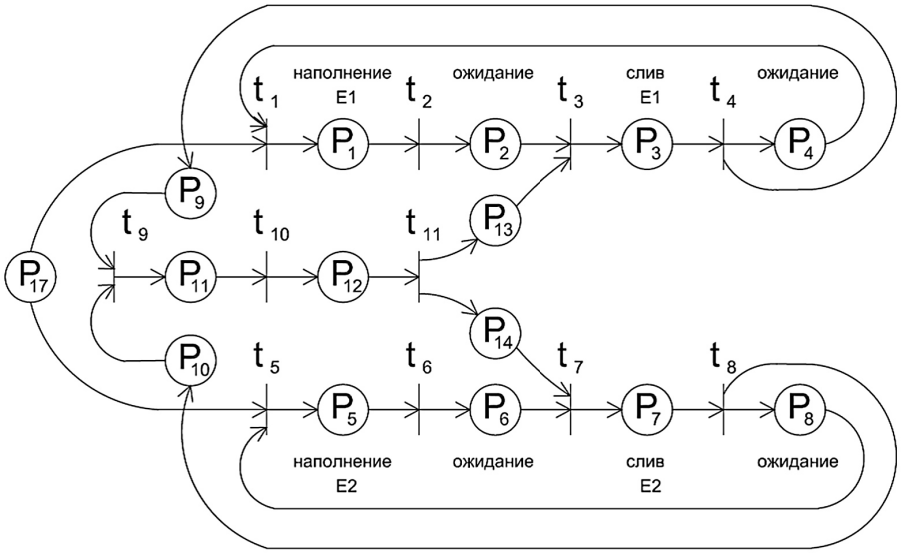

Рис. 6. Модель АТК в виде сети Петри без синхронизации процессов слива из E_1 и E_2

Требуется обеспечить цикл, включающий операции заполнения емкостей E_1 и E_2 от уровней $Д_2$ и $Д_4$ до уровней $Д_1$ и $Д_3$ соответственно, опустошение емкостей E_1, E_2 в ёмкость E_3, перемешивание продуктов A и B в E_3 в течение времени τ и слив смеси до уровня $Д_5$.

В качестве модели работы данного автоматизированного технологического комплекса (АТК) используем сеть Петри, которая приведена на рисунке 6 [8].

Анализ свойств сети показывает, что она «правильная», т. е. безопасная и живая. Это означает, что автомат, управляющий процессом, будет работоспособным.

Входные переменные:

$K_н$ – логическая переменная, соответствующая состоянию тумблера;

$Д_1$, $Д_2$, $Д_3$, $Д_4$, $Д_5$ – логические переменные сигналов соответствующих датчиков; единичные значения сигналов соответствуют достижению уровней расположения датчиков;

$T_м$ – сигнал с таймера; $T_м$ = 1 соответствует включенному таймеру.

Всего 7 входных переменных.

Выходные переменные (соответствуют командам на ИУ):

K_1, K_2, K_3, K_4, K_5 – выходные логические переменные, соответствующие положению клапанов; единичные значения переменных соответствуют открытому положению клапанов;

K_6 = 1 – сигнал на включение таймера; одновременно с запуском таймера включается мешалка (M – сигнал на включение мешалки), которая отключается при завершении работы таймера.

Имена позиций имеют смысл предикатов (условий) с исходами: «0» – не выполняется, «1» – выполняется.

$P_1 = \overline{D}_1$ – условие заполнения E_1;

$P_2 = D_1$ – емкость E_1 заполнена, режим ожидания разрешения на слив;

$P_3 = \overline{D}_2$ – условие окончания слива из E_1;

$P_4 = \overline{D}_2$ – слив из E_1 окончен, режим ожидания начала нового цикла;

$P_5 = \overline{D}_3$ – условие заполнения E_2;

$P_6 = D_3$ – емкость E_2 заполнена, режим ожидания разрешения на слив;

$P_7 = \overline{D}_4$ – условие окончания слива из E_2;

$P_8 = \overline{D}_4$ – слив из E_2 окончен, режим ожидания начала нового цикла;

$P_9 = \overline{D}_2$ – одно из условий $P_9 \wedge P_{10}$ на начало перемешивания в E_3 (включение мешалки и таймера) со стороны емкости E_1;

$P_{10} = \overline{D}_4$ – одно из условий $P_9 \wedge P_{10}$ на начало перемешивания в E_3 (включение мешалки и таймера) со стороны емкости E_2;

$P_{11} = T_м$ – условие окончания перемешивания: таймер отработал время τ (в течение этого времени $T_м$ = 1) и отключился;

$P_{12} = \overline{D}_5$ – условие окончания слива из E_3;

$P_{13} = P_{14} = \overline{D}_5$ – условия, разрешающие слив из E_1, E_2; выполняют роль координации работы трех частей схемы управления;

P_{17} – положение тумблера, разрешающего ($P_{17} = K_н$ = «1») или запрещающего ($\overline{P}_{17} = \overline{K}_н$ = «0») работу автомата.

Активизация переходов вызывает изменение состояния ИУ. Связь переходов и значений команд для ИУ представлена в матрице инициализаций (табл. 2).

Синтез конечно-автоматной модели на основе таблицы автомата Мура для модели такого размера встречает большие технические сложности. Количество

Таблица 2

Матрица инициализаций

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}
K_1	1	0	0	0	*	*	*	*	*	*	*
K_2	0	0	1	0	*	*	*	*	*	*	*
K_3	*	*	*	*	1	0	0	0	*	*	*
K_4	*	*	*	*	0	0	1	0	*	*	*
K_5	*	*	*	*	*	*	*	*	0	1	0
K_6	*	*	*	*	*	*	*	*	1	0	0

Примечание: знак «*» означает, что состояние ИУ не изменяется.

входных логических переменных равно 7. Общее число состояний, даже без учета комбинаций состояний отдельных субавтоматов, составляет 11. Размер таблицы состояний автомата Мура будет $2^7 * 11 = 128 * 11$, что делает процедуру синтеза технически невозможной.

Поэтому синтез может проводиться только путем декомпозиции схемы управления на подсети, каждой из которых соответствует субавтомат.

Декомпозиция проводится с использованием двух рекомендаций:

- общее число состояний субавтоматов (пар «позиция – переход») не должно быть больше 5–6;
- подсети обычно выделяются по принципу сильной связности элементов подсети по сравнению с элементами других подсетей.

Можно показать, что синтез логических последовательностей для декомпозированных частей автомата на основе модели автомата Мура (см. п. 2 раздела 1, вариант 2.1) остается проблематичным. Поэтому рассматриваются варианты 2.2 и 2.3, т. е. синтез по УТС.

Модели субавтоматов нижнего уровня, координируемые «флагами», приведены на рисунке 7.

В этом случае координация работы подсетей проводится через «флаги», *формируемые при активизации отдельных переходов подсетей*.

Так при активизации t_4 формируется «флаг» B_1, т. е. исполнительные действия $K_1 = 0$; $K_2 = 0$ для этого перехода дополняются командой $B_1 = 1$. Аналогично для t_8 исполнительные действия дополняются командой $B_2 = 1$. Для t_{11} – дополняется командой $B_3 = 1$. Для t_9 – дополняется командами $B_1 = B_2 = B_3 = 0$.

Матрица инициализаций примет вид (см. табл. 3).

Таблица соответствия кодирует состояния через промежуточные переменные:

$R_i (i = 1,2,3)$ для подсети на рисунке 7 а;
$Q_i (i = 1,2,3)$ для подсети на рисунке 7 б;
$S_i (i = 1,2)$ для подсети на рисунке 7 в.

В соответствии с изложенными выше процедурами получим логические последовательности:

$$R_1 = R_1 \wedge R_3 \vee R_1 \wedge \overline{R}_2 \wedge \overline{R}_3 \vee P_2 \wedge B_3 \wedge \wedge R_2 \wedge \overline{R}_1 \wedge R_3,$$

$$R_2 = R_1 \wedge \overline{R}_3 \vee \overline{R}_1 \wedge R_2 \wedge R_3 \vee (P_{17} \wedge P_4 \vee P_1) \vee \vee P_2 \wedge B_3 \wedge R_1 \wedge R_2 \wedge R_3,$$

$$R_3 = R_2 \wedge R_3 \vee \overline{R}_1 \wedge R_3 \vee P_{17} \wedge P_4 \vee \overline{R}_1 \wedge \overline{R}_2 \wedge \wedge \overline{R}_3 \vee R_1 \wedge \overline{R}_2 \wedge R_3,$$

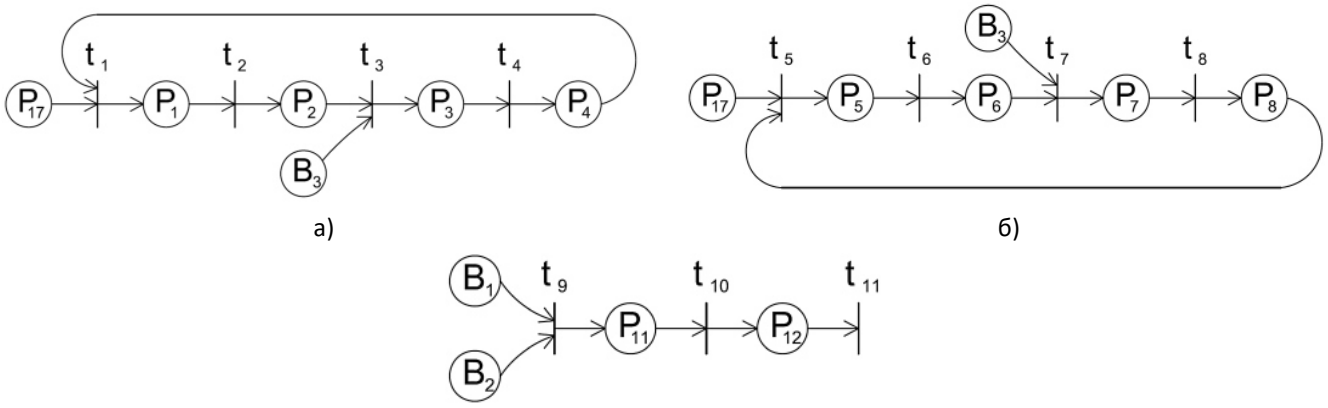

Рис. 7. Модели субавтоматов для сети на рисунке 6

$$t_1 = \overline{R}_1 \wedge \overline{R}_2 \wedge R_3,$$
$$t_2 = \overline{R}_1 \wedge R_2 \wedge R_3,$$
$$t_3 = R_1 \wedge R_2 \wedge R_3,$$
$$t_4 = R_1 \wedge \overline{R}_2 \wedge R_3.$$

$$Q_1 = Q_1 \wedge Q_3 \vee Q_1 \wedge \overline{Q}_2 \wedge \overline{Q}_3 \vee P_6 \wedge B_3 \wedge \wedge Q_2 \wedge \overline{Q}_1 \wedge Q_3,$$

$$Q_2 = Q_1 \wedge \overline{Q}_3 \vee \overline{Q}_1 \wedge \overline{Q}_2 \wedge Q_3 \vee (P_{17} \wedge P_8 \vee P_5) \vee \vee P_6 \wedge B_3 \wedge Q_1 \wedge Q_2 \wedge Q_3,$$

$$Q_3 = Q_2 \wedge Q_3 \wedge \overline{Q}_1 \wedge Q_3 \vee P_{17} \wedge P_8 \wedge \overline{Q}_1 \wedge \overline{Q}_2 \wedge \wedge \overline{Q}_3 \vee Q_1 \wedge \overline{Q}_2 \wedge Q_3,$$

$$t_5 = \overline{Q}_1 \wedge \overline{Q}_2 \wedge Q_3,$$
$$t_6 = \overline{Q}_1 \wedge Q_2 \wedge Q_3,$$
$$t_7 = Q_1 \wedge Q_2 \wedge Q_3,$$
$$t_8 = Q_1 \wedge \overline{Q}_2 \wedge Q_3.$$

$$S_1 = (P_{11} \vee P_{12}) \wedge S_1 \vee P_{11} \wedge S_2 \vee \vee (B_1 \wedge B_2 \vee P_{11} \vee P_{12}),$$

$$S_2 = (B_1 \wedge B_2 \vee P_{11} \vee P_{12}) \wedge \overline{S}_1 \wedge S_2 \vee \vee (B_1 \wedge B_2 \vee P_{11}) \wedge S_2 \vee (B_1 \wedge B_2) \wedge S_1,$$

$$t_9 = \overline{S}_1 \wedge S_2,$$
$$t_{10} = S_1 \wedge S_2,$$
$$t_{11} = S_1 \wedge \overline{S}_2.$$

Состояние ИУ K_1, K_2, K_3, K_4, K_5, K_6 и значения «флагов» при активизации переходов (см. табл. 3).

Для координации работы субавтоматов «флаги» можно либо формировать при активизации выходных (терминальных) переходов подсетей, либо сформировать координирующий автомат. Первый вариант используется для относительно простых сетей с количеством субавтоматов не более 4–5, второй – обладает общностью методики и позволяет синтезировать сложные автоматы.

Координирующий автомат строится на основе идентификации состояний по логическим выражениям переходов, имена которых закодированы через промежуточные переменные.

При этом входными логическими переменными для координирующих автоматов будут имена переходов автоматов нижнего уровня, а выходами – «флаги», используемые для запуска-останова автоматов нижнего уровня.

В рассматриваемом примере координирующая сеть будет иметь вид (рис. 8).

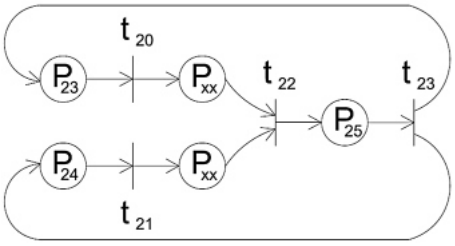

Рис. 8. Подсеть координирующего автомата с использованием позиций, выраженных через имена переходов подсетей нижнего уровня

Позиции P_{xx}, постоянно имеющие значения логической единицы, введены в рассмотрение для соблюдения формализма сетей Петри (обеспечение двудольности графа):

$$P_{23} = t_4 = R_1 \wedge \overline{R}_2 \wedge R_3,\ t_4 \to B_1 = B_2 = 0,$$

Таблица 3

Матрица инициализаций с учетом формирования «флагов»

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}	t_{11}
K_1	1	0	0	0	*	*	*	*	*	*	*
K_2	0	0	1	0	*	*	*	*	*	*	*
K_3	*	*	*	*	1	0	0	0	*	*	*
K_4	*	*	*	*	0	0	1	0	*	*	*
K_5	*	*	*	*	*	*	*	*	0	1	0
K_6	*	*	*	*	*	*	*	*	1	0	0
B_1	*	*	*	1	*	*	*	*	0	*	*
B_2	*	*	*	*	*	*	*	1	0	*	*
B_3	*	*	*	*	*	*	*	*	0	*	1

Примечание: значок «*» означает, что состояние ИУ не изменяется.

$$P_{24} = t_8 = Q_1 \wedge \overline{Q}_2 \wedge Q_3,\ t_{21} \to B_2 = 1,$$
$$P_{xx} \wedge P_{xx} = 1,\ t_{22} \to B_1 = B_2 = 0,$$
$$P_{25} = t_{11} = S_1 \wedge \overline{S}_2,\ t_{23} \to B_3 = 1.$$

Данная сеть «правильная», т. е. живая и безопасная.

С учетом того, что позиции P_{xx} введены для соблюдения формализмов сетей Петри и в синтезе не участвуют, УТС для координирующего субавтомата с учетом коррекции для устранения состязаний будет иметь вид (табл. 4).

Таблица соответствия, кодирующая состояния через промежуточные переменные $Z_i\ (i = 1,2,3)$ (см. рис. 9).

Z_2 \ Z_3	1	0
1	2	1
0	4	3
1	5	7
0	6	8

Z_1

Рис. 9. Таблица соответствия для промежуточных переменных Z_i

Проведем синтез логических выражений для промежуточных функций $Z_i\ (i = 1,2,3)$.

$$Z_1 = \overline{Z}_3 \wedge \left(Z_1 \vee Z_2 \wedge \overline{Z}_1\right) \wedge P_{25},$$
$$Z_2 = \overline{Z}_1 \wedge \left[Z_2 \wedge \left(\overline{Z}_3 \vee P_{25}\right) \vee P_{24}\right],$$
$$Z_3 = Z_1 \wedge Z_2 \wedge \overline{Z}_3 \vee \left[\overline{Z}_1 \wedge P_{24} \wedge \overline{P}_{23}\right] \vee Z_3 \wedge \overline{Z}_2 \wedge$$
$$\wedge \overline{Z}_1 \wedge \left[P_{23} \wedge P_{24} \wedge P_{25}\right] \vee \overline{Z}_1 \wedge Z_3 \wedge P_{25}.$$

Выражения для переходов:

$$t_{20} = \overline{Z}_1 \wedge \overline{Z}_2 \wedge \overline{Z}_3,$$
$$t_{21} = \overline{Z}_1 \wedge Z_2 \wedge Z_3,$$
$$t_{22} = \overline{Z}_1 \wedge Z_2 \wedge \overline{Z}_3,$$
$$t_{23} = Z_1 \wedge Z_2 \wedge \overline{Z}_3.$$

Если в схеме на рисунке 8 отказаться от обеспечения формального свойства «безопасность», которое фактически обеспечивается логическим условием для позиции P_{25}, сеть Петри координирующего автомата будет еще меньше (рис. 10):

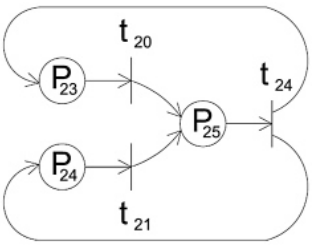

Рис. 10. Упрощенная подсеть координирующего автомата с использованием позиций, выраженных через имена переходов подсетей нижнего уровня

На рисунке 10 переход t_{24} соответствует командам $B_1 = B_2 = 0, B_3 = 1$.

Проведем синтез логических выражений для этого варианта сети, для чего составим УТС для подсети на рисунке 10 (табл. 5).

Таблица соответствия, кодирующая состояния через переменные $H_i\ (i = 1,2)$ будет иметь вид рис. 11.

H_1 \ H_2	1	0
1	1	2
0	4	3

Рис. 11. Таблица соответствия для промежуточных переменных H_1, H_2

Таблица 4

Укрупненная таблица состояний координирующего автомата

№ состояния	$P_{23} \wedge \overline{P}_{24}$	$\overline{P}_{23} \wedge P_{24}$	$P_{23} \wedge P_{24}$	P_{25}	Активизируемые переходы
1	2	3	1	1	
2	2	3	4	2	t_{20}
3	2	3	4	3	t_{21}
4	4	4	4	5	t_{22}
5	6	6	6	5	t_{23}
6	8	8	8	8	
7					
8	1	1	1	1	

Таблица 5

Укрупненная таблица состояний для упрощенной сети Петри

№ состояния	$P_{23} \wedge \overline{P}_{24}$	$\overline{P}_{23} \wedge P_{24}$	P_{25}	Активизируемые переходы
1	2	3	4	
2	2	3	4	t_{20}
3	2	3	4	t_{21}
4	1	4	4	t_{24}

Таблица 6

Укрупненная таблица состояний с учетом промежуточной дополнительной переменной

№ состояния	$P_{23} \wedge \overline{P}_{24}$	$\overline{P}_{23} \wedge P_{24}$	P_{25}	Активизируемые переходы
1	2	8	4	
2	2	3	7	t_{20}
3	2	3	4	t_{21}
4	1	4	4	t_{24}
5			4	
6		3		
7		6	8	
8		7	5	

Видно, что отсутствие состязаний не обеспечивается (есть «опасные» переходы 1 ⟶ 3, 2 ⟶ 4), поэтому количество промежуточных функций увеличивается до 3-х и корректируется УТС (рис. 12, табл. 6).

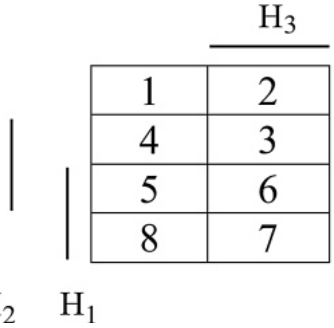

Рис. 12. Таблица соответствия с учетом дополнительной промежуточной переменной, $H_i\,(i=1,2)$

$$H_1 = \overline{H}_2 \wedge \left[P_{23} \wedge \overline{P}_{24} \wedge \left(\overline{H}_3 \vee H_1 \wedge H_3\right) \vee \vee P_{25} \wedge \left(H_1 \vee \overline{H}_1 \wedge H_3\right)\right],$$

$$H_2 = \overline{P}_{24} \wedge P_{23} \wedge \left(H_3 \vee H_2 \wedge \overline{H}_3\right) \vee P_{25} \wedge \wedge \left(\overline{H}_2 \wedge \overline{H}_3 \vee H_2 \wedge \overline{H}_1\right),$$

$$H_3 = P_{23} \wedge \overline{P}_{24} \wedge \overline{H}_2 \vee \left(H_1 \wedge H_2 \wedge \overline{H}_3\right) \vee \vee P_{24} \wedge P_{23} \wedge H_1 \vee H_3 \vee H_2 \wedge \overline{H}_1,$$

$$P_{23} = t_4 = R_1 \wedge \overline{R}_2 \wedge R_3,$$

$$P_{24} = t_8 = Q_1 \wedge \overline{Q}_2 \wedge Q_3,$$

$$P_{25} = t_{11} = S_1 \wedge \overline{S}_2,$$

$$t_{20} = \overline{H}_1 \wedge \overline{H}_2 \wedge H_3,$$

$$t_{21} = \overline{H}_1 \wedge H_2 \wedge H_3,$$

$$t_{24} = \overline{H}_1 \wedge H_2 \wedge \overline{H}_3.$$

Алгоритм работы для иерархических автоматов:

При запуске автоматов формируют начальные значения «флагов». Для рассматриваемого примера $B_1 = B_2 = B_3 = 1$.

Таблица 7

Матрица инициализаций для координирующего автомата (рис. 10)

	t_{20}	t_{21}	t_{24}
B_1	1	*	0
B_2	*	1	0
B_3	*	*	1

1. Последовательно записываются логические выражения для вычисления позиций через входные логические переменные $(P_1,...,P_{17},P_{19},...,P_{22})$;

2. Последовательно записываются логические выражения для вычисления промежуточных функций автоматов нижнего уровня (для сети на рисунке 6)

$R_i(i=1,2,3)$ для подсети на рисунке 7 а;

$Q_i(i=1,2,3)$ для подсети на рисунке 7 б;

$S_i(i=1,2)$ для подсети на рисунке 7 в.

3. Последовательно записываются логические выражения для вычисления функций активизации переходов автоматов нижнего уровня (для сети на рисунке 6 – это $t_1,\ldots, t_{11}$);

4. Последовательно для активных переходов через матрицу соответствия (табл. 3) формируются сигналы на ИУ;

5. Последовательно записываются логические выражения для вычисления позиций координирующего автомата (для схемы на рисунке 10 – это P_{23}, P_{24}, P_{25}, которые формируются через значения переходов автоматов нижнего уровня;

6. Последовательно записываются логические выражения для вычисления функций активизации переходов координирующего автомата (для схемы на рисунке 10 – это t_{20}; t_{21}; t_{24});

7. Последовательно для активных переходов через матрицу инициализаций (табл. 3) либо через условные операторы формируются значения «флагов».

Далее алгоритм выполняется с п. 1.

Заключение

На основе рассмотренных подходов можно сделать следующие выводы:

1. Использование процедур декомпозиции конечно-автоматных моделей является безальтернативным методом для сложных систем.

2. Наиболее перспективным является метод синтеза на основе использования УТС. При этом для конечно-автоматных моделей средней сложности проще использовать метод с «флагами», формируемыми при активизации переходов. Для очень сложных конечно-автоматных моделей лучше использовать метод с «флагами», формируемыми вспомогательным координирующим автоматом.

СПИСОК ЛИТЕРАТУРЫ

1. Горбатов В.А. Логическое управление технологическими процессами. М. : Энергия, 1978. 272 с.

2. Мозгалевский А.В., Калявин В.П. Системы диагностирования судового оборудования. Л. : Судостроение, 1987. 224 с.

3. Фридман А., Менон П. Теория и проектирование переключательных схем. М. : Мир, 1978. 680 с.

4. Смирнов И.Н. Синтез систем управления на логических элементах. Л. : Изд-во ЛГУ, 1975. 78 с.

5. Поспелов Д.А. Логические методы анализа и синтеза схем. М. : Энергия, 1974. 368 с.

6. Лазарев В.Г., Пийль Е.И., Турута Е.Н. Построение программируемых управляющих устройств. М. : Энергоатомиздат, 1984. 192 с.

7. Веревкин А.П., Динкель В. Г. Технические средства автоматизации химико-технологических процессов. Уфа : Изд-во УНИ, 1989. 72 с.

8. Веревкин А.П., Кирюшин О.В. Автоматизация технологических процессов и производств в нефтепереработке и нефтехимии. Уфа : Изд-во УГНТУ, 2005. 171 с.

9. Веревкин А.П., Кирюшин О.В., Павлова З.Х. Метод синтеза сложных логических управляющих устройств с переменными булевой и нечеткой логик // Датчики и системы. 2017. № 12. С. 10–15.

10. Балакирев В.С., Софиев А.Э. Применение средств пневмо- и гидроавтоматики в химических производствах. М. : Химия, 1984. 192 с.

11. Технология, экономика и автоматизация процессов переработки нефти и газа: учеб. пособие / С.А. Ахметов, М.Х. Ишмияров, А.П. Веревкин, Е.С. Докучаев, Ю.М. Малышев. М. : Химия, 2005. 736 с.

12. Юдицкий С.А., Магерут В.З. Логическое управление дискретными процессами. М. : Машиностроение, 1987. 176 с.

13. Иерархические автоматы. URL: https://alphapedia.ru/w/Communicating_finite-state_machine (дата обращения: 22.09.2022).

14. Крон Кеннет, Джон Родс Алгебраическая теория машин. I. Теорема разложения на простые числа для конечных полугрупп и машин // Труды Американского математического общества. 1965. № 116. С. 450–464.

15. Лукин М.А. Верификация параллельных автоматных программ // Научно-технический вестник информационных технологий, механики и оптики. 2014. № 1. С. 60–66.

16. Поликарпова Н.И., Шалыто А.А. Автоматное программирование : учебное пособие. СПБ : ИТМО, 2008. 167 с.

17. Шалыто А.А. Алгоритмизация и программирование для систем логического управления и «реактивных» систем // Автоматика и телемеханика. 2001. № 1. С. 3–39.

18. Harel D., Pnueli A. On the development of reactive systems. Trans. of NATO Advanced Study Institute on Logic and Models for Verification and Specification of Concurrent Systems. Logic and Models of Concurrent Systems. Springer Verlag, 1985. pp. 477–498.

19. Питерсон Дж. Теория сетей Петри и моделирование систем. М. : Мир, 1984. 264 с.

20. Котов В.Е. Сети Петри. М. : Наука. Главная редакция физ.-матем. лит-ры, 1984. 160 с.

REFERENCES

1. Gorbatov V.A. *Logicheskoe upravlenie tekhnologicheskimi protsessami* [Logical Manufacturing Process Control]. Moscow, Energiia Publ., 1978. 272 p.

2. Mozgalevskii A.V., Kaliavin V.P. *Sistemy diagnostirovaniia sudovogo oborudovaniia* [Diagnostic-Troubleshooting Systems of Marine Equipment]. Leningrad, Sudostroenie Publ., 1987. 224 p.

3. Fridman A., Menon P. *Teoriia i proektirovanie perekliuchatelnykh skhem* [The Theory and Design of Logic Switches]. Moscow, Mir Publ., 1978. 680 p.

4. Smirnov I.N. *Sintez sistem upravleniia na logicheskikh elementakh* [Synthesis of Control Systems on Logical Components]. Leningrad, LGU Publ., 1975. 78 p.

5. Pospelov D.A. *Logicheskie metody analiza i sinteza skhem* [Logical Methods of Circuit Analysis and Schematic Synthesis]. Moscow, Energiia Publ., 1974. 368 p.

6. Lazarev V.G., Piyl E.I., Turuta E.N. *Postroenie programmiruemykh upravliaiushchikh ustroistv* [Designing of the Programmable Controlling Systems]. Moscow, Energoatomizdat Publ., 1984. 192 p.

7. Verevkin A.P., Dinkel V. G. *Tekhnicheskie sredstva avtomatizatsii khimiko-tekhnologicheskikh protsessov* [Technological Facilities for Chemical-Engineering Process Automation]. Ufa, UNI Publ., 1989. 72 p.

8. Verevkin A.P., Kiriushin O.V. *Avtomatizatsiia tekhnologicheskikh protsessov i proizvodstv v neftepererabotke i neftekhimii* [Automation of Manufacturing Procedures and Production Processes in the field of Oil Refining and Petroleum Chemistry]. Ufa, UGNTU Publ., 2005. 171 p.

9. Verevkin A.P., Kiriushin O.V., Pavlova Z.Kh. Metod sinteza slozhnykh logicheskikh upravliaiushchikh ustroistv s peremennymi bulevoi i nechetkoi logic [Method for the Synthesis of Complex Logical Controllers with Variables of Boolean and Fuzzy Logics]. *Datchiki i sistemy* [Sensors and Systems], 2017, no. 12, pp. 10–15.

10. Balakirev V.S., Sofiev A.E. *Primenenie sredstv pnevmo- i gidroavtomatiki v khimicheskikh proizvodstvakh* [Application of Hydro-and-Pneumatic Control System in Chemical Industry]. Moscow, Khimiia Publ., 1984. 192 p.

11. Akhmetov S.A., Ishmiiarov M.Kh., Verevkin A.P., Dokuchaev E.S., Malyshev Iu.M. *Tekhnologiia, ekonomika i avtomatizatsiia protsessov pererabotki nefti i gaza.* Ucheb. posobie [Technology, Economics and Automation of Petroleum-and-Gas Processing. Study Guide]. Moscow, Khimiia Publ., 2005. 736 p.

12. Yuditskii S.A., Magerut V.Z. *Logicheskoe upravlenie diskretnymi protsessami* [Logical Control of Discrete Processes]. Moscow, Mashinostroenie Publ., 1987. 176 p.

13. *Ierarkhicheskie avtomaty* [Hierarchical Automata]. Available at https://alphapedia.ru/w/Communicating_finite-state_machine (accessed 22.09.2022).

14. Krohn Kenneth, John Rhodes. Algebraicheskaia teoriia mashin. I. Teorema razlozheniia na prostye chisla dlia konechnykh polugrupp i mashin [Algebraic Theory of Machines. I. Prime Decomposition Theorem for Finite Semigroups and Machines]. *Trudy Amerikanskogo matematicheskogo obshchestva* [Transactions of the American Mathematical Society], 1965, no. 116, pp. 450–464.

15. Lukin M.A. Verifikatsiia parallelnykh avtomatnykh program [Verification of Parallel Automata-Based Programs]. *Nauchno-tekhnicheskii vestnik informatsionnykh tekhnologii, mekhaniki i optiki* [Scientific and Technical Journal of Information Technologies, Mechanics and Optics], 2014, no. 1, pp. 60–66.

16. Polikarpova N.I., Shalyto A.A. *Avtomatnoe programmirovanie.* Uchebnoe posobie [Computed Programming. Handbook]. St. Petersburg, ITMO Publ., 2008. 167 p.

17. Shalyto A.A. Algoritmizatsiia i programmirovanie dlia sistem logicheskogo upravleniia i “reaktivnykh” system [Logic Control and “Reactive” Systems: Algorithmization and Programming]. *Avtomatika i telemekhanika* [Automation and Remote Control], 2001, no. 1, pp. 3–39.

18. Harel D., Pnueli A. On the Development of Reactive Systems. *Trans. of NATO Advanced Study Institute on Logic and Models for Verification and Specification of Concurrent Systems. Logic and Models of Concurrent Systems.* Springer Verlag, 1985, pp. 477–498.

19. Peterson J. *Teoriia setei Petri i modelirovanie system* [Petri Nets Theory and the Modeling of Systems]. Moscow, Mir Publ., 1984. 264 p.

20. Kotov V.E. *Seti Petri* [Petri Nets]. Moscow, Nauka Publ., 1984. 160 p.