УДК 681.518.5
ББК 32.965.7

МЕТОД ПОДСТАНОВОК В ФОРМАЛЬНОМ СИНТЕЗЕ ПРОГРАММ ПЛК ПО ДИСКРЕТНО-СОБЫТИЙНЫМ МОДЕЛЯМ

Хадеев А.С.[1], Браништов С.А.[2]
(*Учреждение Российской академии наук Институт проблем управления РАН, Москва*)

В работе рассмотрено развитие метода синтеза алгоритмов управления для ПЛК для составных моделей. Предложенный подход позволяет автоматически формировать программу поведения для ПЛК при изменении задачи управления. В основе метода лежит синтез супервизора, допускающего безопасное функционирование объекта, по модели технологического объекта, спецификации поведения и ограничений. Полученная супервизорная модель преобразуется в алгоритм для ПЛК методом подстановок. В качестве инструмента моделирования использованы сети Петри, программное обеспечение для ПЛК синтезируется на языке релейно-контактных схем.

Ключевые слова: программируемый логический контроллер, релейно-контактная схема, теория супервизорного управления, дискретно-событийная система, сеть Петри.

1. Введение

Решение задачи комплексного управления современными производственными предприятиями в настоящее время предполагает определённую динамичность в формировании алгоритмов технологического управления. Традиционный подход,

[1] *Хадеев Антон Сергеевич, кандидат технических наук* (*khadeev@gmail.com*).
[2] *Браништов Сергей Александрович, кандидат технических наук* (*pochta-na@mail.ru*).

опирающийся на парадигму экстремального программирования, оказывается малоэффективным при достаточно высокой динамике изменений целей управления.

Типичным промышленным устройством управления является программируемый логический контроллер (ПЛК). Существует набор стандартов, описывающих техники создания программ для ПЛК [9], но эти техники относятся скорее к формальному описанию методов программирования, тогда как логика управления программы контроллера формируется программистом в меру его знаний и представлений, в том числе и о работе системы.

В последнее десятилетие ведутся активные исследования особого класса моделей систем, в основном технических, динамика развития в которых происходит по возникновению событий, как внешних, так и внутренних. Данный класс носит название дискретно-событийных систем (Discrete-Event Systems, ДСС) [8]. Широкое распространение ДСС объясняется развитой методологией, включающей методы моделирования и решения задач логического управления. Дискретно-событийные модели хорошо подходят для описания работы и взаимодействия промышленного оборудования. Стремительно развивается теория супервизорного управления, исследующая методы логического управления в ДСС [1-3]. Основу ей положила фундаментальная работа [11].

В части моделирования управляющего компонента применён метод синтеза супервизора для дискретно-событийной системы, смоделированной сетью Петри, предложенный в работах [10, 12]. Метод основан на концепции инварианты позиций в сети Петри и применим к системам, ограничения для которых могут быть выражены логическими формулами, алгебраическими уравнениями или неравенствами. Супервизор состоит из позиций, которые соединены с переходами сети Петри модели таким образом, что это предотвращает переход модели в любое из запрещённых состояний.

Роль управляющего элемента (супервизора) в моделях систем промышленной автоматизации сводится к координации дискретных действий механизмов. Алгоритм подобных взаимодействий представляет из себя программу для ПЛК, написанную

на стандартизованном языке. Традиционно, для программирования ПЛК используют язык релейно-контактных схем (РКС) [4], известный также, как язык лестничной логики или Ladder Diagram (LD).

2. Супервизорная модель

Часто в качестве инструмента для моделирования ДСС используются конечные автоматы, что обусловлено выбором данного инструмента в фундаментальных исследовательских работах теории супервизорного управления [2]. Однако при увеличении размерности системы этот инструмент сталкивается с проблемой «взрыва состояний», вызванной тем, что конечный автомат должен описывать все возможные состояния системы. Использование в качестве инструмента моделирования сетей Петри помогает решить эту проблему. В настоящей работе используется расширенная концепция сетей Петри, подробно и системно изложенная Дж. Питерсоном [5].

Одним из структурных свойств сети Петри, т.е. свойств, которые зависят только от топологической структуры сети Петри, являются инварианты сети. Инвариантой позиций (Place Invariant) называется набор позиций сети Петри, маркировка в которых остаётся всегда неизменной. Данное определение соответствует частному случаю сохраняющей сети Петри, в которой вес фишки в позиции супервизора приравнивается к нулю. Для сохранения концептуального единства терминологии в настоящей работе далее используется термин инварианта позиций.

Формально, инвариантой позиций называют целочисленный вектор $X = \{x_i\}$, $i = 1, \ldots, n$, $x_i = 0$ или 1: ненулевые значения в элементах вектора соответствуют позициям, входящим в инварианту, а нулевые – всем остальным.

(1) $\mu^T X = \mu_0^T X$

где μ_0 – начальная маркировка, μ – любая последующая.

Приведённое уравнение означает, что возможная взвешенная сумма меток в позициях инварианты остаётся постоянной при любой маркировке, и эта сумма определяется начальной

маркировкой сети Петри. Инварианты позиций определены всеми целочисленными векторами, соответствующими следующему уравнению.

(2) $X^T D = 0$

где D – матрица инцидентности сети Петри размерностью $n \times m$, n – число позиций, m – число переходов в сети.

Ограничения, предъявляемые к системе можно представить в виде следующего неравенства.

(3) $\mu_i + \mu_j \leq 1$

где μ_i и μ_j – маркировки позиций i и j соответственно. Данное выражение означает, что обе позиции не могут быть маркированы одновременно.

Неравенство (3) может быть превращено в равенство путём добавления подстановочной переменной μ_s.

(4) $\mu_i + \mu_j + \mu_s = 1$

Подстановочная переменная в данном случае представляет новую позицию p_s, в которую попадают лишние фишки, тем самым обеспечивая выполнение равенства – сумма фишек в позициях μ_i и μ_j будет равной 1. Эта позиция принадлежит к сети супервизора, дополняющей модель системы в сети Петри. Таких позиций будет столько же, сколько ограничений предъявлено системе, тем самым и размер супервизора пропорционален количеству ограничений.

Т.к. к сети добавлена новая позиция, матрица инцидентности D модели всей управляемой системы будет состоять из матрицы модели системы D_p размерностью $n \times m$, увеличенной на то количество строк, сколько подстановочных переменных введено в ограничения. Эти новые строки принадлежат матрице инцидентности сети супервизора D_c. Дуги, соединяющие позицию супервизора с оригинальной сетью Петри системы будут вычислены с помощью выражения (2). Все выражения типа (3) можно выразить в матричной форме следующим образом:

(5) $L \cdot \mu_p \leq b$

где μ_p – вектор маркировки сети Петри, моделирующей процесс; L – матрица размерности $n_c \times n$; b – вектор размерности $n_c \times 1$; n_c – число ограничений типа (3).

Схожим образом выражения с подставленными переменными (4) могут быть сгруппированы в матричной форме:

(6) $L \cdot \mu_p + \mu_c = b$

где μ_c – вектор размерности $n_c \times 1$, представляющий маркировку позиций супервизора.

Инварианта позиций, определённая в формуле (4), должна соответствовать формуле (2). Следующее матричное уравнение представляет собой выражение инварианты позиций для всех инвариант, определённых формулой (6).

(7) $X^T D = [LI] \times \begin{bmatrix} D_p \\ D_c \end{bmatrix} = 0 \Leftrightarrow LD_p + D_c = 0 \Leftrightarrow D_c = -LD_p$

где I – единичная матрица размерности $n_c \times n_c$, D_c – матрица, содержащая дуги, соединяющие супервизор с переходами сети Петри моделирующей систему.

Таким образом, с заданной моделью системы в сети Петри (D_p) и ограничениями, которым система должна соответствовать (n_c, l и b), супервизор в сети Петри (D_c) вычисляется формулой (7).

Необходимо также вычислить начальную маркировку сети Петри супервизора μ_{c0}, которая должна быть такой, чтобы выполнялась формула (6). Отталкиваясь от выражения (1) формула (6) может быть преобразована следующим образом для начальной маркировки

(8) $L \cdot \mu_{p0} + \mu_{c0} = b \Leftrightarrow \mu_{c0} = b - L \cdot \mu_{p0}$

Продемонстрируем применение метода синтеза супервизора на примере железнодорожного стрелочного перевода (ж.д. стрелки). Стрелка, как показано на рис. 1, имеет два конечных положения: на первом и на втором путях, а также два промежуточных, в которые попадает во время перестановки между путями. Смена состояния происходит после получения соответствующей команды извне. Помимо этого, перевод оснащён датчиком, распознающим нахождение на стрелке вагона.

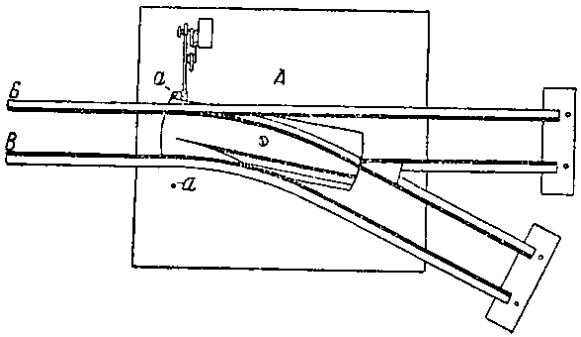

Рис. 1. Ж.д. стрелочный перевод

Задача управления в данном случае заключается в том, чтобы предотвратить переключение стрелочного перевода в момент нахождения на нем вагона. Дискретно-событийная модель стрелки в виде сети Петри представлена на рис. 2.

Приведём описание позиций и переходов в сети Петри: p_1 – стрелка указывает на путь 1, p_2 – стрелка переводится на путь 2, p_3 – стрелка указывает на путь 2, p_4 – стрелка переводится на путь 1, p_5 – стрелка свободна, p_6 – на стрелке находится вагон, t_1 – команда на перевод стрелки на путь 2, t_2 – стрелка переведена на путь 2, t_3 – команда на перевод стрелки на путь 1, t_4 – стрелка переведена на путь 1, t_5 – датчик сигнализирует о нахождении вагона на стрелке, t_6 – вагон покинул стрелку.

Матрица инцидентности D_p для данной сети Петри:

$$(9)\quad D_p = \begin{bmatrix} -1 & 0 & 0 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}$$

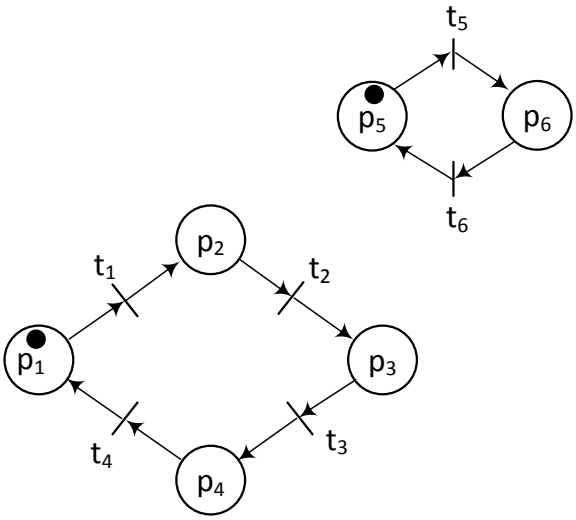

Рис. 2. Дискретно-событийная модель стрелки

Начальная маркировка сети Петри:

$$(10)\ \mu_{p0} = \begin{bmatrix} \mu_1 & \mu_2 & \mu_3 & \mu_4 & \mu_5 & \mu_6 \end{bmatrix}^T = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}^T$$

Для осуществления поставленной задачи управления стрелочный перевод не должен выполнять переключение между путями (позиции p_2 и p_4), когда на стрелке находится вагон (позиция p_6). Согласно заданным условиям, ограничения для данной сети Петри состоят в том, что позиции p_2 и p_4 достижимы только при отсутствии маркировки в позиции p_6, т.е. фишка может находиться только в одной из этих трёх позиций одновременно. Зададим ограничения в виде неравенства, как показано в формуле (3): $\mu_2 + \mu_4 + \mu_6 \leq 1$. В матричном виде данное неравенство будет выглядеть следующим образом: L = [0 1 0 1 0 1], b = 1. После добавления подстановочной переменной ограничения примут вид: $\mu_2 + \mu_4 + \mu_6 + \mu_s = 1$. Подстановочная переменная μ_s является маркировкой позиции p_s, принадлежащей сети супервизора. Матрица инцидентности супервизора вычисляется по формуле (7). В данном примере

введено одно ограничение, поэтому матрица состоит из одной строки:

(11) $$D_c = -LD_p = \begin{bmatrix} -1 & 1 & -1 & 1 & -1 & 1 \end{bmatrix}$$

Начальная маркировка позиции супервизора μ_{c0} вычисляется по формуле (8): $\mu_{c0} = 1 - L\mu_{p0} = 1$:

(13) $$\mu_0 = \begin{bmatrix} \mu_{p0} \\ \mu_{c0} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}^T$$

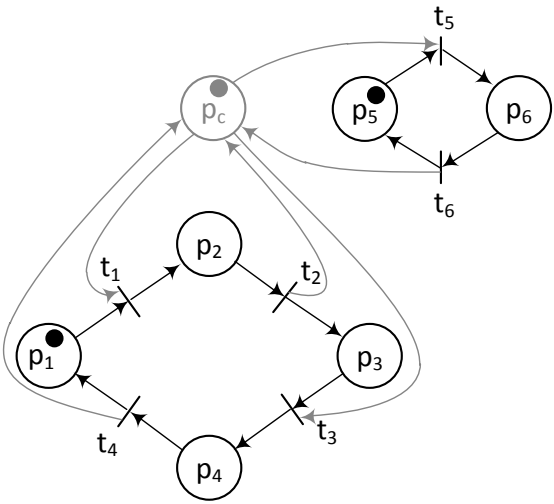

Рис. 3. Супервизорная модель стрелки

Синтезированный супервизор в сети Петри представляет собой дополнительные позиции, которых будет столько же, сколько ограничений предъявлено модели. Для исследуемого примера сеть Петри с синтезированным супервизором приведена на рис. 3.

3. Метод преобразования модели в программу

Сеть Петри с супервизором представляет модель объекта, обеспечивающую решение поставленной задачи управления. Практическую реализацию управления в промышленных объек-

тах выполняет ПЛК. Необходимо преобразовать имеющуюся модель в язык программирования для ПЛК, такой как язык лестничной логики. Метод преобразования сети Петри приведён далее в виде пошагового алгоритма.

На первом этапе нужно указать точки соприкосновения модели и ПЛК, для чего сопоставить его входы и выходы событиям и условиям модели. Составление требует участия человека, и выполняется на этапе моделирования системы. Формально, обозначим сопоставление по следующему принципу:

$$x(t_i), t_i \in PLC_I,\ y(p_k), p_k \in PLC_O \tag{14}$$

где PLC_I – множество входных сигналов на ПЛК, PLC_O – множество сигналов телеуправления, подаваемых с ПЛК.

Данные выражения говорят о том, что условиям для срабатывания переходов в модели сопоставляются сигналы, подаваемые на ПЛК, а позициям – выходные команды, которые с ПЛК поступают на исполнительные механизмы. Далее приведён пошаговый алгоритм преобразования модели сети Петри с сопоставленными позициями и переходами в язык лестничной логики.

Шаг 1. Сопоставление

$$\forall t_i, t_i \in I(p_k), p_k \in PLC_O \tag{15}$$

Переход t_i учитывается в программе на языке лестничной логики в виде ступеньки, если одна из выходных позиций для него p_k отмечена как выход ПЛК. Далее, для каждой такой позиции составляется отдельная ступенька.

Шаг 2. Логические условия

$$\chi(t_i) = x(t_i) \wedge \bigwedge\nolimits_{j=1}^{l} x(t_v), t_v \in I(p_j), t_i \in O(p_j) \tag{16}$$

Для перехода t_i формулируется логическое условие $\chi(t_i)$, необходимое для подачи команды управления в p_k. Логическое условие формулируется по принципу конъюнкции условий всех входных переходов для каждой из позиций p_i, от которой идёт дуга к переходу t_i, включая и условия для самого перехода t_i. l – количество всех входных дуг в t_i.

Шаг 3. Исключение лишних условий

(17) $x(t_j), t_j \in O(p_k)$

Из $\chi(t_i)$ исключаются условия, принадлежащие переходам, к которым идут дуги из p_k.

Шаг 4. Преобразование логического условия

Число ступенек в программе будет равно числу позиций p_k, сопоставленных выходам ПЛК. Для каждой такой позиции составляется отдельная ступенька, условия срабатывания которой заданы в $\chi(t_i)$, $t_i \in I(p_k)$. Схема сопоставления приведена в таблице 1.

Таблица 1. Сопоставление условий элементам языка лестничной логики

Элемент логического условия	Элемент языка лестничной логики
$x(t_j)$	
$\overline{x(t_j)}$	
$y(p_k)$	

Формально, срабатывание логического условия можно определить как $\chi(t_i) => y(p_k)$, то есть результат вычисления условия $\chi(t_i)$ записывается в сопоставленный p_k выход ПЛК.

Покажем применение метода преобразования сети Петри в язык лестничной логики на ранее рассмотренном примере ж.д. стрелочного перевода.

Шаг 1. Сопоставление

Сопоставление входов и выходов ПЛК модели объекта в сети Петри приведено в таблицах 2-3. Отдельно остановимся на принципе сопоставления условий для переходов t_5 и t_6. Эти переходы соответствуют событиям, генерируемым датчиком нахождения вагона на стрелочном переводе. Данный датчик работает по схеме «сухой контакт», что предполагает у него два состояния: замкнут и разомкнут. В силу этого одно из событий, генерируемых данным устройством, будет инверсным.

Таблица 2. Сопоставление входов ПЛК событиям

$\chi(t_i)$	t_i	Описание
I1.1	t_1	Команда на перевод стрелки на путь 2
I1.2	t_2	Стрелка переведена на путь 2
I1.3	t_3	Команда на перевод стрелки на путь 1
I1.4	t_4	Стрелка переведена на путь 1
I1.5	t_5	Вагон находится на стрелке
$\overline{I1.5}$	t_6	Вагон на стрелке не находится

Таблица 3. Сопоставление выходов ПЛК состояниям для примера с ПУ

$y(p_k)$	p_k	Описание
O2.1	p_2	Подача команды на перевод стрелки на путь 1
O2.2	p_6	Подача команды на перевод стрелки на путь 2

Шаг 2. Логические условия

Для примера ж.д. стрелки условия будут следующими: $\chi(t_1) = \chi(t_1) \wedge \chi(t_2) \wedge \chi(t_4) \wedge \chi(t_6)$ и $\chi(t_3) = \chi(t_3) \wedge \chi(t_2) \wedge \chi(t_4) \wedge \chi(t_6)$.

Шаг 3. Исключение лишних условий

Из логического условия $\chi(t_1)$ будет исключён элемент $\chi(t_2)$. Из $\chi(t_3)$ будет исключён $\chi(t_4)$. В окончательном виде логические условия примут вид:

$$\chi(t_1) = x(t_1) \wedge x(t_4) \wedge x(t_6) = I1.1 \wedge I1.4 \wedge \overline{I1.5} \quad (18)$$

$$\chi(t_3) = x(t_2) \wedge x(t_3) \wedge x(t_6) = I1.2 \wedge I1.3 \wedge \overline{I1.5}$$

Шаг 4. Преобразование логического условия

Условия на языке лестничной логики примут вид, как показано на рис. 4.

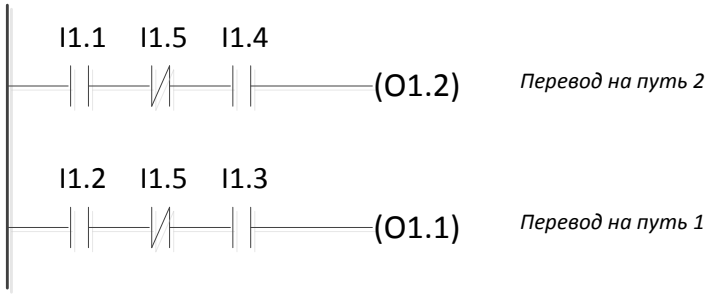

Рис. 4. Алгоритм переключения стрелки на языке лестничной логики

4. Метод подстановок

Моделирование сложных объектов можно существенно упростить, если выполнить моделирование составных компонент этих объектов (модульный подход), а далее объединить их. Т.е. части из общего множества событий и условий в системе структурно выделить в отдельные устройства и описать частными моделями. Представив каждую из таких частных моделей как «чёрный ящик», можно выделить для неё входные и выходные сигналы, являющиеся соответственно выходами и входами для других моделей: $x \leftarrow \chi$, $x \in M_1$, $\chi \in M_2$.

Метод подстановок описывает форму того, как цепочки логических условий, необходимых для формирования выходного воздействия в подстановочной модели (M_2), подставляются в логические условия подставляемой модели (M_1). В том случае, если несколько моделей определяют условие, возможны две формы их взаимодействия, описанные ниже.

Форма конкурирующих моделей. Для возникновения события в подстановочной системе достаточно того, чтобы одна из подставляемых моделей перешла в соответствующее выходное состояние. В этом случае, форма взаимодействия определяется по формуле: $x \leftarrow \chi_1 \vee \chi_2$, $x \in M_1$, $\in M_2$, $\chi \in M_3$, где M_1 – подстановочная модель, M_2, M_3 – подставляемые модели. В программе на языке лестничной логики конкурирующие модели представляются

отдельными ступеньками, соединяющимися с основной ступенькой в местах подстановки, как показано на рисунке 5.

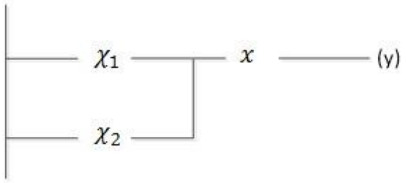

Рис. 5. Формирование программы для конкурирующих моделей

Форма совместных моделей. Для возникновения события в подстановочной системе необходимо, чтобы все подставляемые модели перешли в соответствующее выходное состояние. В этом случае, форма взаимодействия определяется по формуле: $x \leftarrow \chi_1 \wedge \chi_2, x \in M_1, \in M_2, \chi \in M_3$. В программе совместные модели объединяются в основной ступеньке (рис. 6).

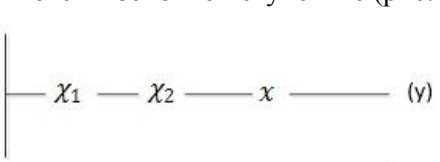

Рис. 6. Формирование программы для совместных моделей

В качестве примера исследуем полный алгоритм переключения стрелочного перевода, исследованный в работе [6]. Алгоритм представляет из себя последовательность проверок на возможность выполнения перевода стрелки. Стрелочный перевод нельзя переключать, если установлена аварийная блокировка в системе или если на стрелочном переводе находится вагон. Подача команды переключения возможна либо с пульта управления стрелкой (ПУ), либо с встроенного блока управления стрелкой (БУ), причём данные пульты могут быть заблокированы, что проверяется соответствующим условием.

Покажем, как можно использовать модульный подход с подстановками в сетях Петри. Выделим отдельный компонент алгоритма, осуществляющий непосредственное управление

стрелкой, он представлен на рис. 7. Данный компонент уже был смоделирован ранее, результат применения для него метода подстановок показан на рис. 4. Входом данной частной модели является команда на перевод стрелки от оператора, а выходом – выдача управляющего воздействия на стрелочный перевод.

Следующими частными моделями будут блок подачи команды с пульта управления, и схожий блок подачи команд с БУ стрелки. В силу их одинаковости рассмотрим лишь один из них – блок подачи команды с ПУ. Данный фрагмент порождает два события: команда с пульта управления подана или не подана. В том случае, если команда подана, нужно убедиться, что есть разрешение на эту команду, и только после этого переходить к фазе управления. В силу того, что события, ведущие к терминальному положению «Конец» фактически никаких действий не выполняют, рассматриваем только события успешной проверки.

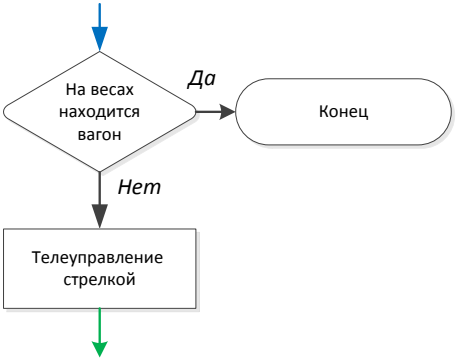

Рис. 7. Фрагмент алгоритма переключения стрелки, блок управления стрелкой

Обе блока управления имеют по две кнопки: одну для перевода на первый путь и другую для перевода на второй. Супервизорная модель для алгоритма по нажатию кнопок в сетях Петри для данного блока показана на рис. 8, где p_{21}/p_{22} – кнопка на ПУ для перевода на путь 1 находится в нажатом/отжатом состоянии;

p_{23}/p_{24} – стрелку переводить разрешено/запрещено; p_{25}/p_{26} – кнопка на ПУ для перевода на путь 2 находится в нажатом/отжатом состоянии; t_{21}/t_{22} – нажатие/отжатие кнопки на ПУ для перевода на путь 1; t_{23}/t_{24} – разрешение/запрет на перевод стрелки; t_{25}/t_{26} – нажатие/отжатие кнопки на ПУ для перевода на путь 2.

Применив метод синтеза супервизора и сопоставление ко всему алгоритму, получим сеть Петри, приведённую на рис. 8. Формулы сопоставленных условий и логические условия после применения метода преобразования и метода подстановок примут вид (19). Программа на языке лестничной логики для алгоритма железнодорожного стрелочного перевода, полученная в результате применения предложенной методики, приведена на рис. 10.

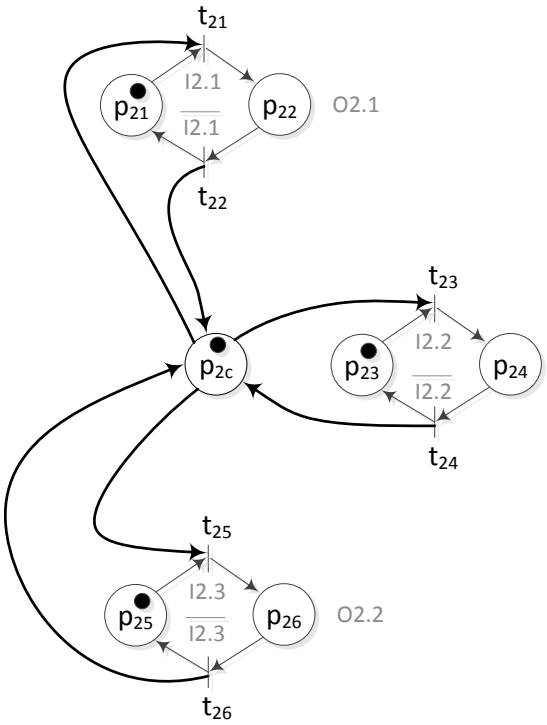

Рис. 8. Супервизорная модель блока подачи команды от ПУ

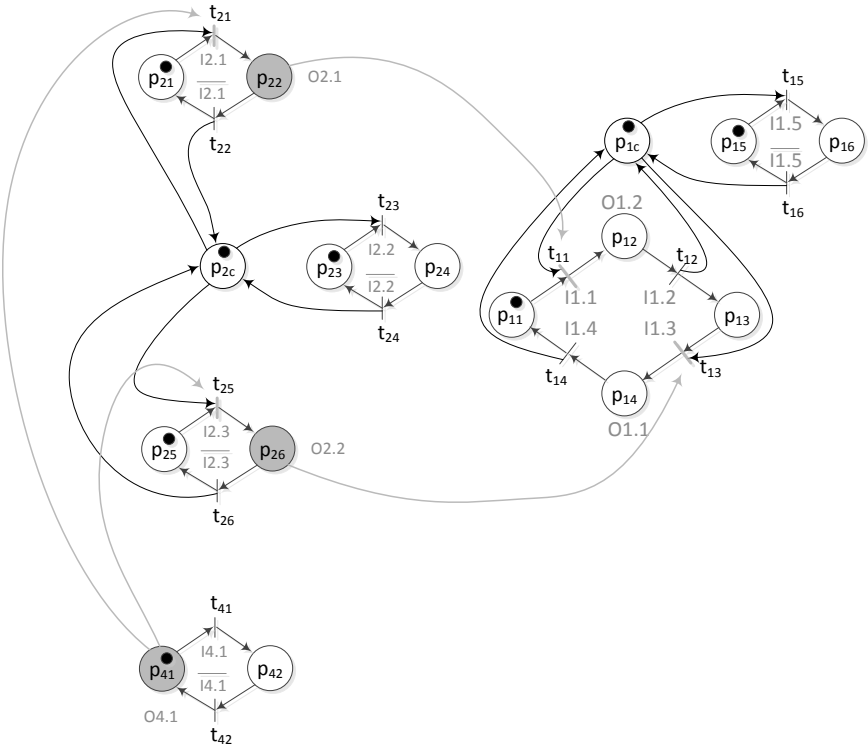

Рис. 9. Сеть Петри с управлением для алгоритма стрелочного перевода

(19) $x(t_{11}) = \chi(t_{21}) \vee \chi(t_{31})$; $x(t_{11}) \in M_1, \chi(t_{21}) \in M_2, \chi(t_{31}) \in M_3$

$x(t_{13}) = \chi(t_{25}) \vee \chi(t_{35})$; $x(t_{13}) \in M_1, \chi(t_{25}) \in M_2, \chi(t_{31}) \in M_3$

$$\chi(t_{11}) = ((I2.1 \wedge \overline{I2.2} \wedge \overline{I2.3}) \vee (I3.1 \wedge \overline{I3.2} \wedge \overline{I3.3})) \wedge I1.4 \wedge \overline{I1.5}$$

$$\chi(t_{13}) = ((I2.3 \wedge \overline{I2.2} \wedge \overline{I2.1}) \vee (I3.3 \wedge \overline{I3.2} \wedge \overline{I3.1})) \wedge I1.3 \wedge \overline{I1.5}$$

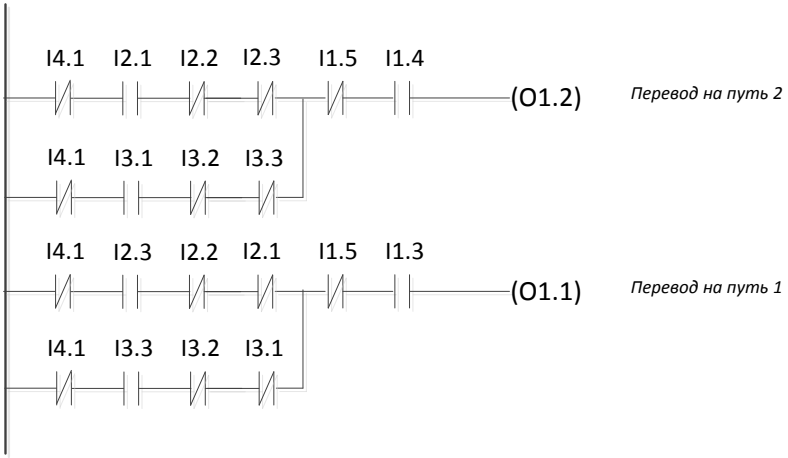

Рис. 10. Программа для алгоритма стрелочного перевода

5. Заключение

В работе подробно изложен хорошо алгоритмизированный метод, который включает формирование логики управления технологическим объектом и синтез программного обеспечения для ПЛК. Метод предусматривает участие человека на этапе моделирования и сопоставления моделей входам и выходам ПЛК, тем не менее, он позволяет автоматизировать основные действия разработчика по проектированию алгоритма и программированию, что снижает вероятность человеческой ошибки. Используя подход дискретно-событийного моделирования можно применить богатый набор методов анализа алгоритма и свойств объекта управления, что делает данный подход очень гибким с точки зрения модификации и поиска ошибок.

Метод подстановок, изложенный во второй части работы, позволяет применить принципы синтеза программы ПЛК. Для сложных моделей, также возможно применить эти принципы, путём их декомпозиции на множество простых и синтеза программы для каждой из них в отдельности. Показано как после этого формально построить общую модель объекта и программу управления.

Литература

1. АМБАРЦУМЯН А.А. *Супервизорное управление структурированными динамическими дискретно-событийными системами* // Автоматика и телемеханика. – 2009. – №. 8. – С. 156–176.
2. АМБАРЦУМЯН А.А. *Моделирование и синтез супервизорного управления на сетях Петри для рассредоточенных объектов. II. Метод синтеза супервизора по множеству последовательностей общего вида* // Автоматика и телемеханика. – 2011. – №. 9. – С. 173–189.
3. АМБАРЦУМЯН А.А. Моделирование и синтез супервизорного управления на сетях Петри для рассредоточенных объектов. I. Механизм взаимодействия и базовый метод // Автоматика и телемеханика. 2011. Vol. 8. P. 151–169.
4. ДЕНИСЕНКО В.В. *Компьютерное управление технологическим процессом, экспериментом, оборудованием.* – М.: Горячая линия-Телеком, 2013. – 606 с.
5. ПИТЕРСОН Д. *Теория сетей Петри и моделирование систем*: Пер. с англ. – М.: Мир, 1984. – 264 с.
6. ХАДЕЕВ А.С. *Обзор средств автоматического управления процессом взвешивания на железнодорожной эстакаде налива* // Автоматизация IT в нефтегазовой области. – 2012. – №. 3. – С. 67–69.
7. ХАДЕЕВ А.С., БРАНИШТОВ С.А. *Метод синтеза LD-программ для ПЛК на основе концепции супервизорного управления дискретно-событийными системами, представленными в сетях Петри* // Автоматизация, телемеханизация и связь в нефтяной промышленности. – 2013. – №. 1. – С. 42–48.
8. CASSANDRAS C., LAFORTUNE G. *Introduction to Discrete Event Systems*. 2-nd ed. – Springer, 2008. – 771 p.
9. *International Standard IEC 61131-3. Programmable Controllers – Part 3: Programming Languages*. 2-nd Ed. – International Electrotechnical Commission, 2003. – 143 p.
10. MOODY J.O., ANTSAKLIS P.J. *Petri Net Supervisors for DES with Uncontrollable and Unobservable Transitions* // IEEE Trans. Automat. Contr. – 2000. – Vol. 45, № 3. – P. 462–476.

11. WONHAM W., RAMADGE P. *The Control of Discrete Event Systems* // Proceedings of the IEEE. – 1989. – Vol. 1, № 77. – P. 81-98.
12. YAMALIDOU K. et al. *Feedback Control of Petri Nets Based on Place Invariants* // Automatica. – 1996. – Vol. 32, № 1. – P. 15–28.

PLC ALGORYTHMS SYNTHESIS BASED ON THE SUPERVISORY CONTROL THEORY

Anton Khadeev, Institute of Control Sciences of RAS, Moscow, Cand. of Science (khadeev@gmail.com).
Branishtov Sergey, Institute of Control Sciences of RAS, Moscow, Cand. of Science (branishtov@mail.ru).

Abstract: *Progress in enchacement of PLC programming concentrated in the field of programming technics and language notations development, though less attention given to control synthesis methods. In this work authors propose the method to synthesize PLC programming control algorithms. The method allows to develop the control program to PLC in automatic mode when a control aim is changed. The method based on the Supervisory Control Theory which allows to build a new model of control object with behavior constraints included, and this new model grant only required behavior. Then the model transforms into the algorithm for PLC. As an modeling instrument Petri Nets are used, PLC programs are synthesized in Ladder Diagram language.*

Keywords: PLC, Ladder Diagrams, Supervisory Control Theory, Discrete Event Systems, Petri Nets.