

ДИСКРЕТНЫЕ ФУНКЦИИ И АВТОМАТЫ

УДК 004

DOI: 10.17223/19988605/37/10

И.Б. Бурдонов, А.С. Косачев

ОБОБЩЕННАЯ МОДЕЛЬ СИСТЕМЫ АВТОМАТОВ

Статья посвящена проблеме моделирования, композиции и детерминизма составных систем. Компоненты системы моделируются конечными автоматами с несколькими входами и выходами, а взаимодействие между ними – обменом сообщениями по симплексным каналам связи. Система описывается ориентированным графом связей, вершина которого соответствует автомату компонента, а дуга – каналу связи, соединяющему выход одного автомата с входом другого. Автоматы системы работают синхронно: на каждом такте каждый автомат выполняет один переход. Определяются ассоциативная композиция автоматов системы по графу связей и условия детерминированности системы.

Ключевые слова: ориентированные графы; взаимодействующие автоматы; композиция автоматов; детерминизм; сети.

Большинство сложных, особенно распределённых, систем представляет собой набор взаимодействующих компонентов. В данной статье компоненты моделируются конечными автоматами, а взаимодействие – обменом сообщениями между автоматами. Автомат может иметь несколько входов для приёма сообщений и может принимать сразу несколько сообщений; также автомат может иметь несколько выходов для отправки сообщений и посылать сразу несколько сообщений. Структура связей между компонентами моделируется ориентированным графом (будем называть его *графом связей*), вершинам которого соответствуют автоматы, а дуги называются *соединениями* и соответствуют симплексным каналам передачи сообщений. Соединение, ведущее из автомата A в автомат B , помечается выходом j автомата A и входом i автомата B . При этом каждый вход (выход) каждого автомата соединён не более чем с одним выходом (входом) другого (или того же самого) автомата. Входы (выходы) автоматов, которые не соединяются с выходами (входами) автоматов, являются *внешними*, через них осуществляется связь системы с её *окружением*. Автоматы системы работают синхронно: на каждом такте каждый автомат выполняет один переход. Также на каждом такте окружение системы посылает сообщения на некоторые внешние входы системы и принимает сообщения с некоторых внешних выходов системы.

Считается, что граф связей статический, т.е. не меняющийся в процессе работы системы. В этом случае система (так же как её компоненты) может моделироваться конечным автоматом, получающимся из автоматов-компонентов с помощью подходящего оператора композиции, учитывающего граф связей.

Такая модель системы является обобщением модели, предложенной в [1]. Там дуга графа связей представляла собой, по сути, очередь сообщений длины 1. Она моделируется автоматом с одним входом и одним выходом. Синхронно взаимодействуют автомат в вершине с автоматом дуги, инцидентной этой вершине. Если в графе связей из [1] каждую дугу разбить на два соединения, вставив в «середину» дуги новую вершину, в которую поместить автомат дуги, то получится система автоматов, являющаяся частным случаем системы, предлагаемой в данной статье. «Частность» этого случая заключается в том, что автомат дуги может теперь моделировать не только очередь сообщений длины 1, но и очередь любой другой (в том числе неограниченной) длины, стек и т.д. Более того, в данной статье мы, по сути, не делаем различия между автоматом вершины и автоматом дуги: автомат дуги тоже может иметь несколько входов и выходов.

Основная особенность предлагаемой модели автомата заключается в том, что переход автомата, предъявляя требования к состоянию всех входов и выходов автомата (указываются сообщения на них),

отдельно указывает ту часть входов и выходов, по которым сообщения принимаются или посылаются. Синхронность взаимодействия автоматов означает, что для каждого соединения требования автоматов, связанных этим соединением, должны быть согласованы. Это даёт возможность описывать более широкий спектр поведений автомата. Например, приоритетный приём сообщений: если на входах автомата имеется несколько сообщений, автомат может принять сообщения с наивысшим приоритетом, не принимая остальные сообщения. Также это позволяет автомату выполнять приём сообщений детерминированно, т.е. независимо от того, удастся или не удастся одновременно послать некоторое сообщение на некоторый выход. Если удастся послать сообщение, автомат выполняет один переход, а если не удастся – то другой. Обычно это приводит к недетерминизму: на один стимул выдаются разные реакции. Однако в нашей модели реакция – это требования к выходам, а не указание той части выходов, по которым сообщения реально передаются.

В статье определяется ассоциативная композиция переходов, автоматов и системы по известному графу связей. Ассоциативность важна для того, чтобы работа системы зависела только от множества её автоматов и соединений и не зависела от какого-либо упорядочивания этих множеств. Дополнительно определяется композиция автоматов, не связанных соединениями, что позволяет докомпоновать систему ровно до одного автомата, который может использоваться как компонент более сложных систем автоматов.

Исследуется проблема детерминизма: даётся определение детерминированного автомата и выясняются условия, при которых композиция детерминированных автоматов является детерминированной.

1. Модель

Пусть задано некоторое конечное множество сообщений M , которое мы будем называть *алфавитом* (сообщений). Пустое множество \emptyset будем называть *пустым сообщением*, считать, что $\emptyset \notin M$, и обозначать $M_\emptyset \triangleq M \cup \{\emptyset\}$.

Автомат в алфавите M – это набор $A = (M, I, J, S, T, s_0)$, где:

- I – конечное множество *входов* автомата;
- J – конечное множество *выходов* автомата;
- S – конечное множество *состояний* автомата;
- $T \subseteq S \times X \times P \times Y \times Q \times S$ – множество *переходов* автомата, где:
- $X = \{x \mid x: I \rightarrow M_\emptyset\}$ – множество *стимулов*;
- $Y = \{y \mid y: I \rightarrow M_\emptyset\}$ – множество *реакций*;
- $P = 2^I$ – семейство подмножеств входов (по ним принимаются сообщения);
- $Q = 2^J$ – семейство подмножеств выходов (по ним посылаются сообщения);
- $s_0 \in S$ – *начальное состояние*;

причём выполнены следующие условия:

- 1) множества входов и выходов не пересекаются: $I \cap J = \emptyset$;
- 2) принимаются и посылаются только непустые сообщения:

$$\forall (s, x, p, y, q, t) \in T (\forall i \in p (x(i) \neq \emptyset) \ \& \ \forall j \in q (y(j) \neq \emptyset)),$$

что эквивалентно $\forall (s, x, p, y, q, t) \in T (p \subseteq x^{-1}(M) \ \& \ q \subseteq y^{-1}(M))$.

Очевидно, что из конечности множеств M, I, J и S следует конечность множеств X, Y, P, Q, T .

Для перехода $a = (s, x, p, y, q, t)$ состояние s будем называть *пресостоянием* перехода, а состояние t – *постсостоянием*, и будем обозначать $s_a = s, x_a = x, p_a = p, y_a = y, q_a = q, t_a = t$. Для автомата $A = (M, I, J, S, T, s_0)$ будем обозначать $I_A = I, J_A = J, S_A = S, T_A = T, s_{0A} = s_0$.

Работу автомата можно описать следующим образом. В текущем состоянии s проверяется состояние входов, т.е. какие сообщения можно принять с входов автомата. Это определяет стимул x . Далее рассматриваются переходы по этому стимулу, т.е. множество $T_{s,x} \subseteq T$ переходов из s по x . Каждый переход из множества $T_{s,x}$ определяет ту или иную реакцию y . Если для данных s и x имеются переходы с разными реакциями y , то недетерминированным образом выбирается реакция y как одна из реакций на переходах из множества $T_{s,x}$. После выбора реакции y проверяется состояние выходов, а именно какие

непустые сообщения, определяемые реакцией y , могут быть переданы по соответствующим выходам, а какие нет. Это определяет параметр q . Тем самым определяется множество $T_{s,x,y,q} \subseteq T_{s,x}$ переходов с данными s, x, y, q . Если это множество содержит более одного перехода, то недетерминированным образом выбирается один из них. Если выбран переход (s, x, p, y, q, t) , то параметр p определяет, какие непустые сообщения будут приняты с входов автомата. Таким образом, принимаются сообщения, определяемые стимулом x , по входам, определяемым параметром p ; посылаются сообщения, определяемые реакцией y , по выходам, определяемым параметром q , после чего автомат переходит в состояние t . Это описание работы автомата неформально. Формальное описание есть, по существу, формальное определение композиции автоматов, которое будет дано в следующем разделе.

Пусть V – конечное множество автоматов в алфавите M . Без ограничения общности можно считать, что входы, выходы и состояния автоматов разные: $\forall A, B \in V (A \neq B \Rightarrow I_A \cap I_B = \emptyset \ \& \ I_A \cap J_B = \emptyset \ \& \ J_A \cap J_B = \emptyset \ \& \ S_A \cap S_B = \emptyset)$. Это всегда можно сделать систематическим переименованием состояний, входов и выходов автоматов, в результате чего получатся автоматы, изоморфные исходным.

Система автоматов – это набор $R = (M, V, E)$, где $E : E_{Dom} \rightarrow E_{Im} \rightarrow$ биекция, определяющая *соединения*, где $E_{Dom} \subseteq J_R$, $E_{Im} \subseteq I_R$, $J_R = \cup \{J_A | A \in V\}$ – множество всех выходов всех автоматов, $I_R = \cup \{I_A | A \in V\}$ – множество всех входов всех автоматов. Для $R = (M, V, E)$ обозначим $V_R = V$, $E_R = E$.

Определим функцию $\phi : (I_R \cup J_R) \rightarrow V$, которая каждому входу или выходу системы R ставит в соответствие автомат, которому этот вход или выход принадлежит: $\forall A \in V \ \forall z \in I_A \cup J_A \ \phi(z) = A$. Будем говорить, что соединение (j, i) связывает выход автомата $\phi(j)$ и вход автомата $\phi(i)$. Вход $i \in I_R$ или выход $j \in J_R$, для которого не определено соединение, т.е. $i \notin E_{Im}$ или $j \notin E_{Dom}$, будем называть *внешним входом* или *внешним выходом* системы соответственно.

Системе $R = (M, V, E)$ соответствует ориентированный *граф связей*, вершинами которого являются автоматы из V , а помеченные дуги – это соединения, причём дуга $A \rightarrow B$ существует и помечена соединением (j, i) тогда и только тогда, когда $(j, i) \in E$, $\phi(j) = A$ и $\phi(i) = B$. Такой граф обозначим через $Og(R)$. Если снять ориентацию соединений, то получившийся неориентированный граф обозначим через $Ng(R)$. Будем говорить, что автоматы A и B *связаны* в системе, если в графе $Ng(R)$ существует (неориентированный) путь из A в B (возможно, пустой, если $A = B$). В противном случае разные автоматы A и B *не связаны*. Очевидно, что отношение связности автоматов является отношением эквивалентности и разбивает множество автоматов V на классы эквивалентности, которые будем называть *классами связности*.

Будем считать, что состояние автомата в системе $R = (M, V, E)$ является множеством и состояния разных автоматов не пересекаются: $\forall A, B \in V \ \forall s_A \in S_A \ \forall s_B \in S_B \ (A \neq B \Rightarrow s_A \cap s_B = \emptyset)$. Если уже задана система, в которой это условие не выполнено, то в каждом автомате каждое состояние s заменим на синглетон $\{s\}$, что даст возможность получить автоматы, изоморфные исходным, а требуемые условия будут выполнены.

Работу системы автоматов можно описать следующим образом. Прежде всего, окружение моделируется автоматом, число выходов (входов) которого равно числу внешних входов (выходов) системы, и добавляются внешние соединения, связывающие выходы (входы) окружения с внешними входами (выходами) системы. Получается замкнутая система без внешних входов и выходов. Все автоматы, включая окружение, срабатывают одновременно, и каждый автомат выполняет ровно один переход (явный или неявный, если используются правила умолчания). Если автоматы A и B связаны соединением (j, i) , где $j \in J_A$ и $i \in I_B$, то для выполняемых переходов $a \in T_A$ и $b \in T_B$ сообщение $y_a(j)$ и условие его передачи $j \in q_a$ на выходе j автомата A должны быть согласованы с сообщением $x_b(i)$ и условием его приёма $i \in p_b$ на входе i автомата B . Это означает, во-первых, что сообщение $y_a(j)$ на выходе j совпадает с сообщением $x_b(i)$ на входе i . И, во-вторых, сообщение либо передаётся из A по выходу j , т.е. $j \in q_a$, и принимается в B по входу i , т.е. $i \in p_b$, либо не передаётся из A по выходу j , т.е. $j \notin q_a$, и не принимается в B по входу i , т.е. $i \notin p_b$.

2. Композиция

Существует большое разнообразие как автоматоподобных моделей, так и операторов их (параллельной) композиции начиная с классической композиции в алгебрах процессов CSP – Communicating Sequential Processes [4] и CCS – Calculus of Communicating Systems [5]. Они определяют две разновидности композиции LTS (Labelled Transition System), в которых переходы помечены символами *действий* из заданного алфавита. Для систем ввода-вывода (IOTS – Input-Output Transition System), в которых действия разбиваются на стимулы (input) и реакции (output), предлагались различные модификации композиции [7–9], в частности для обнаружения тупиковых ситуаций (deadlock) с целью наблюдения *отказов* (refusal) [6–8]. Для систем с приоритетами и моделей *событий* (систем более общего вида) нами были также предложены соответствующие операторы композиции [10,11]. Вводимая в данном разделе композиция учитывает наличие у автомата нескольких входов и выходов и моделирует взаимодействие автоматов через соединения, каждое из которых связывает выход одного автомата с входом другого. Иными словами, это композиция автоматов по графу связей системы автоматов.

Введём способ сокращённой записи для преобразования функции при уменьшении её домена: для функции f и множества N обозначим $f|N \triangleq f \setminus \{(z, f(z)) \mid z \in N \cap \text{Dom}(f)\}$. Для любого множества N' имеем $f|(N \cup N') = (f|N)|N'$, и для любой функции g , такой, что $\text{Dom}(f) \cap \text{Dom}(g) = \emptyset$ и $N \cap \text{Dom}(g) = \emptyset$, имеем $(f \cup g)|N = (f|N) \cup g$. Будем считать, что операция “/” имеет тот же приоритет, что разность “\” и объединение “ \cup ” множеств. В выражениях над множествами, где используются только операции “/”, “\” и “ \cup ”, будем использовать бесскобочную запись и предполагать, что операции выполняются слева направо. Логическую эквивалентность обозначим символом « \sim »: $a \sim b \triangleq (a \& b) \vee (\neg a \& \neg b)$.

Композиция определяется по соединению $(j, i) \in E$. Обозначим через A автомат, которому принадлежит выход j , т.е. $A = \varphi(j)$, а через B – автомат, которому принадлежит вход i , т.е. $B = \varphi(i)$.

Композиция переходов по соединению. Для соединения (j, i) определим условие $f(a, j, i, b)$ композиции двух переходов a и b и результат композиции $a[j, i]b$:

$$f(a, j, i, b) \triangleq a \in T_A \& b \in T_B \& (A = B \Rightarrow a = b) \& y_a(j) = x_b(i) \& (j \in q_a \sim i \in p_b),$$

$$f(a, j, i, b): a[j, i]b \triangleq (s_a \cup s_b, x_a \cup x_b \setminus \{i\}, p_a \cup p_b \setminus \{i\}, y_a \cup y_b \setminus \{j\}, q_a \cup q_b \setminus \{j\}, t_a \cup t_b).$$

Заметим, что если $A = B$, то

$$f(a, j, i, b) \triangleq a \in T_A \& y_a(j) = x_a(i) \& (j \in q_a \sim i \in p_a),$$

$$f(a, j, i, b): a[j, i]b \triangleq (s_a, x_a \setminus \{i\}, p_a \setminus \{i\}, y_a \setminus \{j\}, q_a \setminus \{j\}, t_a).$$

Композиция переходов естественным образом распространяется на композицию множеств переходов, определяемую как множество попарных композиций переходов. Для соединения (j, i) определим композицию множеств переходов G и H : $G[j, i]H \triangleq \{a[j, i]b \mid a \in G \& b \in H \& f(a, j, i, b)\}$.

Композиция автоматов по соединению. В композиции автоматов оставим только начальное состояние композиции и пре- и постсостояния переходов композиции. Это множество включает подмножество достижимых состояний. Для множества переходов T обозначим множество пре- и постсостояний этих переходов: $\text{States}(T) \triangleq \{s_a \mid a \in T\} \cup \{t_a \mid a \in T\}$. Для соединения (j, i) определим композицию автоматов A и B :

$$A[j, i]B \triangleq (M, I_A \cup I_B \setminus \{i\}, J_A \cup J_B \setminus \{j\}, \{s_{0A} \cup s_{0B}\} \cup \text{States}(T_A[j, i]T_B), T_A[j, i]T_B, s_{0A} \cup s_{0B}).$$

Заметим, что если в определении автомата используются правила умолчания, то композиция автоматов существенно использует композицию не только явных, но и неявных переходов.

Теорема 1. Композиция автоматов удовлетворяет условиям, налагаемым на автомат.

Доказательство см. в [2], теорема 1.

Композиция системы по соединению. Для системы R и соединения (j, i) определим

$$R[j, i] \triangleq (M, \bigvee \{A, B\} \cup \{A[j, i]B\}, E \setminus \{(j, i)\}).$$

Теорема 2. Композиция системы удовлетворяет условиям, налагаемым на систему.

Доказательство см. в [2], теорема 2.

Определённая таким образом композиция обладает свойством ассоциативности, которая важна для того, чтобы работа системы зависела только от множеств соединений, автоматов и их переходов и не зависела от какого-либо упорядочивания этих множеств. *Последовательностью всех соединений* для системы $R = (M, V, E)$ назовём последовательность Z соединений из E , в которой каждое соединение встречается ровно один раз, т.е. $Z : [1..|Z|] \rightarrow E$ является биекцией. Определим композицию системы R по последовательности всех соединений Z как $R[Z] \triangleq R[Z(1)][Z(2)]\dots[Z(|Z|)]$, если Z не пусто, и $R[Z] \triangleq R$, если Z пусто.

Теорема 3. Композиция системы R по всем её соединениям не зависит от порядка композиции: для любых двух последовательностей всех соединений Z и Z' имеет место $R[Z] = R[Z']$.

Доказательство см. в [2], теорема 3.

В системе $R[Z]$ нет соединений, т.е. $E_{R[Z]} = \emptyset$; автоматы такой системы не связаны друг с другом и соответствуют классам связности системы R , а все входы и выходы автоматов являются внешними. Если в $R[Z]$ больше одного автомата, то её нельзя использовать как компонент более сложной системы. Для того чтобы обойти это неудобство, определим композицию несвязанных автоматов. Её семантика достаточно прозрачна: если два автомата не связаны, то системы из этих автоматов эквивалентны одному автомату, в котором объединены входы и выходы обоих автоматов, а любые два перехода из разных автоматов объединены в один переход. Определим композицию без соединения формально.

Композиция переходов: $a[]b \triangleq (s_a \cup s_b, x_a \cup x_b, p_a \cup p_b, y_a \cup y_b, q_a \cup q_b, t_a \cup t_b)$.

В этом определении неявно предполагается, что как домены стимулов, так и домены реакций в компонуемых переходах не пересекаются: $Dom(x_a) \cap Dom(x_b) = Dom(y_a) \cap Dom(y_b) = \emptyset$. В противном случае стимул и / или реакция в композиционном переходе могли бы стать многозначными функциями. Мы не записали это требование как условие композиции двух переходов, поскольку такая композиция будет использоваться только для определения композиции несвязанных и, следовательно, разных автоматов, а в этом случае это требование автоматически выполнено.

Композиция множеств переходов: $G[]H \triangleq \{a[]b \mid a \in G \ \& \ b \in H\}$.

Композиция автоматов: $A[]B \triangleq (M, I_A \cup I_B, J_A \cup J_B, \{s_{0A} \cup s_{0B}\} \cup States(T_A[]T_B), T_A[]T_B, s_{0A} \cup s_{0B})$.

Композиция автоматов удовлетворяет условиям, налагаемым на автомат (аналог теоремы 1).

Композиция системы: $R[A, B] \triangleq (M, \bigvee \{A, B\} \cup \{A[]B\}, E)$.

Композиция системы удовлетворяет условиям, налагаемым на систему (аналог теоремы 2). Каждая композиция системы уменьшает на единицу число автоматов в ней. Поэтому через конечное число композиций система $R[Z]$ превратится в систему без соединений, состоящую из одного автомата, который обозначим через R^\wedge .

Нетрудно показать, что композицию несвязанных автоматов можно чередовать с композицией по соединениям. Можно только отметить, что композиция несвязанных автоматов уменьшает на 1 число компонентов связности в графе $Ng(R)$: автомат $A[]B$ оказывается связанным с каждым автоматом (отличным от A и B), с которым был связан автомат A или B .

3. Детерминизм

Автомат $A = (M, I, J, S, T, s_0)$ определен по всем стимулам, если выполнено условие, что в каждом состоянии есть переход по каждому стимулу: $\forall s \in S \ \forall x \in X \ \exists a \in T (s_a = s \ \& \ x_a = x)$.

Автомат $A = (M, I, J, S, T, s_0)$ вполне определен, если он определен по всем стимулам и существуют переходы по всем q : $\forall a \in T \ \forall q' \subseteq y_a^{-1}(M) \ \exists a' \in T (s_{a'} = s_a \ \& \ x_{a'} = x_a \ \& \ p_{a'} = p_a \ \& \ y_{a'} = y_a \ \& \ q_{a'} = q')$.

Вместо этих условий можно использовать *правила умолчания*, если в определении автомата указывать подмножество T^* явных переходов и дополнять его до множества T всех переходов согласно правилам умолчания:

1. Если в T^* нет явного перехода из некоторого состояния по некоторому стимулу, то это означает, что такой переход определён неявно без приёма и выдачи сообщений и без изменения состояния.

2. Если в T^* есть переход (s, x, p, y, q, t) , но для некоторого $q' \subseteq y^{-1}(M)$ нет перехода вида (s, x, p, y, q', t') , то это означает, что такой переход определён неявно без изменения состояния ($t' = t$).

Формально это означает

$$T \triangleq T^* \cup \{(s, x, \emptyset, J \times \{\Lambda\}, \emptyset, s) \mid s \in S \& x \in X \& \forall p, y, q, t (s, x, p, y, q, t) \notin T^*\} \cup \\ \cup \{(s, x, p, y, q', s) \mid \exists q, t (s, x, p, y, q, t) \in T^* \& q' \subseteq y^{-1}(M) \& \forall t' (s, x, p, y, q', t') \notin T^*\}.$$

Автомат A будем называть *детерминированным*, если выполнены следующие условия:

1. Состояние s и стимул x однозначно определяют приём сообщений p и реакцию y :

$$\forall a, b \in T_A (s_a = s_b \& x_a = x_b \Rightarrow p_a = p_b \& y_a = y_b).$$

2. Одинаково помеченные переходы из одного состояния совпадают, т.е. ведут в одно постсостояние:

$$\forall a, b \in T_A (s_a = s_b \& x_a = x_b \& p_a = p_b \& y_a = y_b \& q_a = q_b \Rightarrow t_a = t_b).$$

Проблема в том, что композиция детерминированных автоматов может оказаться недетерминированной. Это может произойти из-за цикла соединений, если в автоматах этого цикла есть две «параллельные», но разные цепочки компонуемых переходов, ведущих из одинаковых состояний и отличающихся в параметрах x и q только по входам и выходам соединений цикла. Для одной цепочки по циклу соединений передаются одни сообщения, а для другой – другие. Более строго, пусть в графе связей есть цикл как чередующаяся последовательность автоматов и неповторяющихся соединений $W = A_1, (j_1, i_2), A_2, (j_2, i_3), \dots, A_n, (j_n, i_{n+1})$, $A_1: \forall k, k'=1..n (j_k, i_{k+1}) \neq (j_{k'}, i_{k'+1})$. Обозначим $i_1 = i_{n+1}$. Пусть также в каждом из автоматов A_k имеются два перехода a_k и a'_k , такие, что если $A_k = A_{k'}$, то $a_k = a_{k'}$. Обозначим $a_{n+1} = a_1$ и $a'_{n+1} = a'_1$. Пусть переходы в соседних в цикле автоматов, т.е. переходы a_k и a_{k+1} , и переходы a'_k и a'_{k+1} компонуются по соединению (j_k, i_{k+1}) , т.е. $f(a_k, j_k, i_{k+1}, a_{k+1}) = f(a'_k, j_k, i_{k+1}, a'_{k+1}) = \text{true}$. Пусть для каждого $k = 1..n$ переходы a_k и a'_k начинаются в одном состоянии и отличаются в x только по входу i_k и в q только по выходу j_k , т.е. если $a_k = (s_k, x_k, p_k, y_k, q_k, t_k)$ и $a'_k = (s'_k, x'_k, p'_k, y'_k, q'_k, t'_k)$, то $s_k = s'_k, x_k \setminus \{i_k\} = x'_k \setminus \{i_k\}, q_k \setminus \{j_k\} = q'_k \setminus \{j_k\}$. Тогда композиция по всем соединениям цикла W содержит два перехода $a = a_1[j_1, i_2]a_2[j_2, i_3]...a_n[j_n, i_{n+1}]a_1$ и $a' = a'_1[j_1, i_2]a'_2[j_2, i_3]...a'_n[j_n, i_{n+1}]a'_1$, в которых $s_a = s_{a'}, x_a = x_{a'}$ и $q_a = q_{a'}$, но, вообще говоря, $p_k \neq p'_k, y_k \neq y'_k$ и $t \neq t'$.

В то же время можно заметить, что если хотя бы в одном автомате цикла, например в первом, по соединению (j_1, i_2) в разных переходах a_1 и a'_1 передаётся одно и то же сообщение $y_1(j_1) = y'_1(j_1)$, то недетерминизм не возникает. Действительно, из условия композиции переходов следует, что в следующем втором автомате $x_2(i_2) = y_1(j_1) = y'_1(j_1) = x'_2(i_2)$, а тогда, поскольку $x_2 \setminus \{i_2\} = x'_2 \setminus \{i_2\}$, имеем $x_2 = x'_2$. Отсюда, поскольку $s_2 = s'_2$, в силу первого условия детерминизма второго автомата имеем $p_2 = p'_2, y_2 = y'_2$, в частности $y_2(j_2) = y'_2(j_2)$. Продолжая аналогичные рассуждения по циклу соединений, получаем $x_k = x'_k, p_k = p'_k, y_k = y'_k$ для каждого k . Тогда для каждого k имеем $i_{k+1} \in p_{k+1} \sim i_{k+1} \in p'_{k+1}$, и по условию композиции $j_k \in q_k \sim j_k \in q'_k$, что вместе с $q_k \setminus \{j_k\} = q'_k \setminus \{j_k\}$ влечёт $q_k = q'_k$. Отсюда по второму условию детерминизма для каждого k имеем $t_k = t'_k$. Тем самым, $a_k = a'_k$ для каждого k , что влечёт $a = a'$. Таким образом, одинаковые сообщения $y_k(j_k) = y'_k(j_k)$ «склеивают» цепочки переходов $a_1 = a'_1, a_2 = a'_2, \dots, a_n = a'_n$, а композиция по такому циклу становится детерминированной.

В [1] композиция детерминированных автоматов детерминированная. В терминах данной статьи это означает, что система детерминированная, если выполнены следующие условия: 1) автоматы разбиты на два класса: класс «вершин» и класс «дуг», 2) каждое соединение связывает автоматы разных классов, 3) «вершина» определена по всем стимулам, а «дуга» вполне определена, 4) все автоматы детерминированные, 5) «дуга» имеет один вход и один выход, 6) «дуга» моделирует очередь сообщений длины 1. Класс «вершин» соответствует вершинам, а класс «дуг» – дугам графа связей из [1]. Условие 5 можно ослабить: достаточно, чтобы «дуга» имела один вход, но она может иметь несколько выходов. Вместо условия 6 достаточно, чтобы в «дуге» реакция зависела только от состояния (т.е. не зависела от стимула): $\forall A \in V_2 \forall a, a' \in T_A (s_a = s_{a'} \Rightarrow y_a = y_{a'})$.

Композиционный переход порождается множеством переходов в автоматах-операндах, которое содержит по одному переходу в каждой «дуге» и либо один переход в «вершине», либо «вершина» «стоит», если какие-то из сообщений, которые она могла бы послать на выходящие «дуги», ими не принимаются.

«Дуга» «склеивает» цепочки в каждом цикле, в который она входит, т.е. композиция по каждому такому циклу становится детерминированной. Для «дуги» в [1] это условие выполнено.

Теорема 4. Если система $R = (M, V, E)$ удовлетворяет указанным выше шести условиям, то после специальной композиции каждой «вершины» со всеми выходящими из неё «дугами» дальнейшая композиция по описанным в разделе 2 правилам дает детерминированный автомат.

Доказательство следует из теорем 1 и 2 в [3].

Заключение

Предложенная в данной статье модель системы автоматов может быть использована как база для тестирования составных систем. Особенно это полезно в том случае, когда в графе связей нет ошибок (*гипотеза о связях*), т.е. граф связей совпадает с заданным. В этом случае тестирование сводится к проверке правильности переходов каждого автомата. Однако проблема в том, что автомат может тестироваться только как часть системы, т.е. тестер не имеет непосредственного доступа к автомату, и вынужден осуществлять тестовые воздействия с помощью сообщений, посылаемых по внешним входным соединениям, которые ведут, быть может, в другие автоматы. Тестирование компонента такой системы похоже на тестирование в контексте [12–15], когда этот компонент рассматривается как тестируемая система, а остальные – как контекст. Существенное отличие, однако, в том, что в таком контексте тоже возможны ошибки, но если верна гипотеза о связях, то только в компонентах, а не в графе связей. С другой стороны, при тестировании проверяется работа сразу нескольких компонентов, через которые проходят сообщения. Поскольку автомат тестируется как часть системы, могут быть проверены не все его переходы, которые проверяются при автономном тестировании с прямым доступом к автомату. Речь идет только о переходах автоматов, *достижимых* при их работе в составе системы.

Тестирование композиционного автомата системы, получающегося композицией автоматов по заданному графу связей, обеспечивает проход по всем его достижимым переходам. При этом, конечно, проверяются все достижимые переходы автоматов-компонентов, но может делаться много «лишней работы». Гипотеза о связях позволяет существенно сократить время тестирования.

Предполагается, что нам известно, каким должен быть каждый автомат (задан граф переходов автомата с точностью до изоморфизма), и именно это проверяется при тестировании. Кроме того, тест может наблюдать как состояния автоматов в вершинах графа связей, так и сообщения на его дугах. Такие предположения могут быть оправданы, например, при имитационном тестировании аппаратуры (*simulation-based verification*) (см. например, [16]).

При этих условиях может быть предложен алгоритм построения набора тестов [3], основанный на фильтрации тестов, генерируемых для композиционного автомата системы: тест отбрасывается, если он покрывает только такие переходы в компонентах системы, которые покрываются уже отобранными тестами. Такой набор тестов будет полным (проверяет все достижимые переходы автоматов-компонентов) при выполнении двух условий: 1) верна гипотеза о связях, 2) система детерминирована. Дополнительно алгоритм может определить недостижимые переходы автоматов.

ЛИТЕРАТУРА

1. Бурдонов И.Б., Косачев А.С. Тестирование системы автоматов // Труды ИСП РАН. 2016. Т. 28 (1).
2. Бурдонов И.Б., Косачев А.С. Система автоматов: композиция по графу связей // Труды ИСП РАН. 2016. Т. 28 (1).
3. Бурдонов И.Б., Косачев А.С. Система автоматов: условия детерминизма и тестирование // Труды ИСП РАН. 2016. Т. 28 (1).
4. Hoare C.A.R. Communicating Sequential Processes. Englewood Cliffs, NJ : Prentice Hall International, 1985.
5. Milner R. Communication and Concurrency. Prentice-Hall, 1989.
6. Langerak R. A testing theory for LOTOS using deadlock detection // Protocol Specification, Testing, and Verification IX / eds. by E. Brinksma, G. Scollo, C.A. Vissers. North-Holland, 1990. P. 87–98.
7. Tretmans J. Test Generation with Inputs, Outputs and Repetitive Quiescence // Software-Concepts and Tools. 1996. Vol. 17, Issue 3.
8. van der Bijl M., Rensink A., Tretmans J. Compositional testing with ioco // Formal Approaches to Software Testing: Third International Workshop, FATES 2003, Montreal, Quebec, Canada, October 6th, 2003 / eds. by Alexandre Petrenko, Andreas Ulrich. LNCS Springer, 2003. V. 2931. P. 86–100.

9. Petrenko A., Yevtushenko N., Huo J.L. Testing Transition Systems with Input and Output Testers. Proc. IFIP TC6/WG6.1 15th Int. Conf. Testing of Communicating Systems, TestCom'2003. Sophia Antipolis, France, 2003. May 26–29. P. 129–145.
10. Бурдонов И.Б., Косачев А.С. Системы с приоритетами: конформность, тестирование, композиция // Программирование. 2009. № 4. С. 24–40.
11. Бурдонов И.Б., Косачев А.С. Согласование конформности и композиции // Программирование. 2013. № 6. С. 3–15.
12. Revised Working Draft on “Framework: Formal Methods in Conformance Testing”. JTC1/SC21/WG1/Project 54/1, ISO Interim Meeting, ITU-T on. Paris, 1995.
13. Bourdonov I.B., Kossatchev A.S., Kuliain V.V. Conformance theory of the systems with Refused Inputs and Forbidden Actions. M.: Nauka, 2008. 412 p. (in Russian)
14. Bourdonov I. Conformance theory (functional testing on formal model base). LAP LAMBERT Academic Publishing. Saarbrücken, Germany, 2011. 428 p. (in Russian)
15. Bourdonov I.B., Kossatchev A.S. Specification Completion for IOCO // Programming and Computer Software. 2011. V. 37 (1). P. 1–14.
16. Kamkin A.S., Chupilko M.M. Survey of modern technologies of simulation-based verification of hardware // Programming and Computer Software. 2011. V. 37 (3). P. 147–152.

Бурдонов Игорь Борисович, д-р физ.-мат. наук. E-mail: igor@ispras.ru

Косачев Александр Сергеевич, канд. физ.-мат. наук. E-mail: kos@ispras.ru

Институт системного программирования РАН (ИСП РАН), Москва

Поступила в редакцию 30 мая 2016 г.

Burdonov Igor B., Kossatchev Alexander S. (Institute for System Programming of the Russian Academy of Sciences, Moscow, Russian Federation).

Generalized model of automata system.

Keywords: directed graph; communicating automata; automata composition; determinism; networks.

DOI: 10.17223/19988605/37/10

The problem of modeling and composing of aggregate systems is considered. The system components are described by finite automata with multiple entries and exits. The communication between automata is described with message passing over simplex communication channels. The system is described with an oriented graph of links. Each node of the graph corresponds to the automaton of a component and an arc corresponds to a communication channel connecting an output of one automaton with input of another automaton. An automaton of the graph node in each state can accept multiple messages from its entries (at most one message from each input) and send multiple messages to its outputs (at most one message to each output). Inputs (outputs) of the automata not connected to outputs (inputs) of other automata are considered to be external and are used for communication between the system and its environment. The automata of the system operate synchronously: on each step each automaton performs one transition. A transition of an automaton imposes requirements on states of all its inputs and outputs (messages in them are specified) and explicitly specifies subset of inputs and outputs through which the messages are received or sent, respectively. Synchronous communication between automata means that for each link the requirements of the automata connected with this link must conform to each other. It makes possible to describe a wider spectrum of automata behavior. For example, a priority of message receiving: if there are multiple message at the automata inputs, it can receive messages with the highest priority and discard the rest of the messages. It also makes possible for the automaton to receive messages regardless of ability to simultaneously send some message to some output. A composition of the automata of the system according to the graph of links is defined and its associativity is proved. In conclusion, the problems of testing of aggregate systems are described.

REFERENCES

1. Burdonov, I.B. & Kossatchev, A.S. (2016) Testing of automata system. *Trudy ISP RAN – The proceeding of ISP RAS*. 28(1). (In Russian).
2. Burdonov, I.B. & Kossatchev, A.S. (2016) Automata system: composition on connection graph. *Trudy ISP RAN – The proceeding of ISP RAS*. 28(1). (In Russian).
3. Burdonov, I.B. & Kossatchev, A.S. (2016) Automata system: conditions of determinism and testing. *Trudy ISP RAN – The proceeding of ISP RAS*. 28(1). (In Russian).
4. Hoare, C.A.R. (1985) *Communicating Sequential Processes*. Englewood Cliffs, New Jersey: Prentice Hall International.
5. Milner, R. (1989) *Communication and Concurrency*. Prentice-Hall.
6. Langerak, R. (1990) A testing theory for LOTOS using deadlock detection. In: Brinksma, E., Scollo, G. & Vissers, C.A. (eds) *Protocol Specification, Testing, and Verification IX*. North-Holland. pp. 87–98.
7. Tretmans, J. (1996) Test Generation with Inputs, Outputs and Repetitive Quiescence. *Software-Concepts and Tools*. 17(3). DOI: 10.1007/3-540-61042-1_42
8. van der Bijl, M., Rensink, A. & Tretmans, J. (2003) Compositional testing with ioco. *Formal Approaches to Software Testing: Third International Workshop, FATES 2003*. Montreal, Quebec, Canada. October 6. 2931. pp. 86–100. DOI: 10.1007/978-3-540-24617-6_7
9. Petrenko, A., Yevtushenko, N. & Huo, J.L. (2003) Testing Transition Systems with Input and Output Testers. *Proc. IFIP TC6/WG6.1 15th Int. Conf. Testing of Communicating Systems, TestCom'2003*. Sophia Antipolis, France. May 26–29. pp. 129–145. DOI: 10.1016/S0140-3664(96)01157-7

10. Bourdonov, I.B. & Kossatchev, A.S. (2009) Systems with Priorities: Conformance, Testing, and Composition. *Programming and Computer Software*. 35(4). pp. 198-211. DOI: 10.1134/S0361768809040045
11. Bourdonov, I.B. & Kossatchev, A.S. (2013) Agreement between Conformance and Composition. *Programming and Computer Software*. 39(6). pp. 269-278. DOI: 10.1134/S0361768813060029
12. Revised Working Draft on “Framework: Formal Methods in Conformance Testing”. (1995) *JTC1/SC21/WG1/Project 54/1, ISO Interim Meeting, ITU-T on*. Paris.
13. Bourdonov, I.B., Kossatchev, A.S. & Kuliainin, V.V. (2008) *Conformance theory of the systems with Refused Inputs and Forbidden Actions*. Moscow: Nauka. (In Russian).
14. Bourdonov, I. (2011) *Conformance theory (functional testing on formal model base)*. Saarbrücken: LAP LAMBERT Academic Publishing.
15. Bourdonov, I.B. & Kossatchev, A.S. (2011) Specification Completion for IOCO. *Programming and Computer Software*. 37(1). pp. 1-14. DOI: 10.1134/S0361768811010014. 1
16. Kamkin, A.S. & Chupilko, M.M. (2011) Survey of modern technologies of simulation-based verification of hardware. *Programming and Computer Software*. 37(3). pp. 147-152. DOI: 10.1134/S0361768811030017