Обобщенная модель системы автоматов

И.Б. Бурдонов, А.С. Косачев

Институт системного программирования, Москва, Россия
igor@ispras.ru, kos@ispras.ru

Доклад посвящен моделированию, композиции и детерминизму составных систем. Компонент системы моделируется конечным автоматом с несколькими входами и выходами, а взаимодействие – синхронным обменом сообщениями по симплексным каналам связи, связывающими выходы со входами и называемыми *соединениями*. Переход автомата предъявляет требования к сооообщениям на всех входах и выходах автомата и отдельно указывает часть входов и выходов, по которым сообщения принимаются или посылаются. Синхронность означает, что требования автоматов, связанных соединением, должны быть согласованы.

Задан конечный *алфавит сообщений* M, $\varnothing \notin M$, и $M_\varnothing = M \cup \{\varnothing\}$. *Автомат* в алфавите M – это набор $A=(M,I,J,S,T,s_0)$, где I – множество *входов*, J – множество *выходов*, S – множество *состояний*, $T \subseteq S \times X \times P \times Y \times Q \times S$ – множество *переходов*, где $X=\{x|x:I \rightarrow M_\varnothing\}$ – множество *стимулов*, $Y=\{y|y:J \rightarrow M_\varnothing\}$ – множество *реакций*, $P=2^I$ – для приёма сообщений, $Q=2^J$ – для посылки сообщений, $s_0 \in S$ – *начальное состояние*, и выполнены условия: 1. $I \cap J=\varnothing$; 2. передаются непустые сообщения: $\forall (s,x,p,y,q,t) \in T\ p \subseteq x^{-1}(M)\ \&\ q \subseteq y^{-1}(M)$; 3. автомат всюду определён по стимулам: $\forall s \in S\ \forall x \in X\ \exists p,y,q,t\ (s,x,p,y,q,t) \in T$; 4. приём сообщений выполним независимо от посылки сообщений: $\forall (s,x,p,y,q,t) \in T\ \ \forall q` \subseteq y^{-1}(M)\ \ \exists t`\ (s,x,p,y,q`,t`) \in T$; 5. множества I, J и S конечны. Для $a=(s,x,p,y,q,t)$ обозначим $s_a=s$, $x_a=x$, $p_a=p$, $y_a=y$, $q_a=q$, $t_a=t$. Для $A=(M,I,J,S,T,s_0)$ обозначим $I_A=I$, $J_A=J$, $S_A=S$, $T_A=T$, $s_{0A}=s_0$.

Пусть V – конечное множество автоматов в алфавите M. Входы и выходы автоматов разные: $\forall A,B \in V\ (A \neq B \Rightarrow I_A \cap I_B=\varnothing\ \&\ I_A \cap J_B=\varnothing\ \&\ J_A \cap J_B=\varnothing)$, состояние автомата – множество, и состояния разных автоматов не пересекаются: $\forall A,B \in V\ \ \forall s_A \in S_A\ \ \forall s_B \in S_B\ (A \neq B \Rightarrow s_A \cap s_B=\varnothing)$. *Система автоматов* – это набор $\boldsymbol{R}=(M,V,E)$, где $E:E_{Dom} \rightarrow E_{Im}$ – биекция, определяющая *соединения*, где $E_{Dom} \subseteq J_{\boldsymbol{R}}$, $E_{Im} \subseteq I_{\boldsymbol{R}}$, $J_{\boldsymbol{R}}=\cup\{J_A|A \in V\}$ – множество всех выходов всех автоматов, $I_{\boldsymbol{R}}=\cup\{I_A|A \in V\}$ – множество всех входов всех автоматов. Для $\boldsymbol{R}=(M,V,E)$ обозначим: $V_{\boldsymbol{R}}=V$, $E_{\boldsymbol{R}}=E$. Вход $i \in I_{\boldsymbol{R}} \backslash E_{Im}$ и выход $j \in J_{\boldsymbol{R}} \backslash E_{Dom}$ – *внешние*, связывающие систему с её окружением.

Вводится композиция в духе CCS [1], но с учётом соединений. Обозначим: для функции f и множества N: $f/N = f \setminus \{(z,f(z))\ |\ z \in N \cap \mathbf{Dom}(f)\}$; для отношения эквивалентности $a \sim b = (a\&b) \vee (\neg a \& \neg b)$; для множества переходов T: $States(T) = \{s_a|a \in T\} \cup \{t_a|a \in T\}$. Пусть $A,B \in V$, $(j,i) \in E$, $j \in J_A$, $i \in I_B$.

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

Композиция переходов $a[j,i]b = (s_a \cup s_b,\ x_a \cup x_b/\{i\},\ p_a \cup p_b \backslash \{i\},\ y_a \cup y_b/\{j\},\ q_a \cup q_b \backslash \{j\},\ t_a \cup t_b)$.

Композиция множеств переходов: $G[j,i]H = \{a[j,i]b|a \in G\ \&\ b \in H\ \&\ f(a,j,i,b)\}$.

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

Композиция системы: $\boldsymbol{R}[j,i] = (M,V \backslash \{A,B\} \cup \{A[j,i]B\},E \backslash \{(j,i)\})$.

Композиция ассоциативна. Композиция $\boldsymbol{R}$^ системы $\boldsymbol{R}$ по последовательности всех её соединений не зависит от их порядка, не имеет соединений, и все её автоматы не связаны между собой соединениями, т.е. все их входы и выходы внешние. Для использования системы в качестве компонента другой системы, она докомпоновывается до одного автомата с помощью композиции без соединения (для несвязанных автоматов).

$a[]b = (s_a \cup s_b,x_a \cup x_b,p_a \cup p_b,y_a \cup y_b,q_a \cup q_b,t_a \cup t_b)$, $G[]H = \{a[]b|a \in G\ \&\ b \in H\}$,

$A[]B = (M,I_A \cup I_B,J_A \cup J_B,\{s_{0A} \cup s_{0B}\} \cup States(T_A[]T_B),T_A[]T_B,\ s_{0A} \cup s_{0B})$, $\boldsymbol{R}[A,B] = (M,\ V \backslash \{A,B\} \cup \{A[]B\},\ E)$.

Автомат A *детерминирован*, если 1. состояние s и стимул x однозначно определяют приём сообщений p и реакцию y и 2. одинаково помеченные переходы из одного состояния совпадают. Если в системе $\boldsymbol{R}$ все автоматы детерминированные, разбиты на два класса: класс «вершин» и класс «дуг», каждое соединение связывает автоматы разных классов и в каждом автомате класса «дуг» реакция зависит только от состояния (не зависит от стимула), то композиционный автомат $\boldsymbol{R}$^ детерминирован.

Преложенную модель можно использовать для тестирования детерминированных составных систем. Если ошибки могут быть только в компонентах, а все соединения правильные, тестирование сводится к проверке правильности переходов каждого автомата. Однако автомат тестируется только как часть системы, что похоже на тестирование в контексте [2]. Предполагается, что известно, какими должны быть автоматы (с точностью до изоморфизма), и именно это проверяется. Тест наблюдает как состояния автоматов, так и передаваемые сообщения. Такие предположения оправданы, например, при имитационном тестировании аппаратуры (simulation-based verification) [16].

Литература

1. Milner R. Communication and Concurrency. Prentice-Hall, 1989.
2. Revised Working Draft on “Framework: Formal Methods in Conformance Testing”. *JTC1/SC21/WG1/Project 54/1, ISO Interim Meeting, ITU-T on*. Paris. 1995.
3. A. S. Kamkin, M. M. Chupilko. Survey of modern technologies of simulation-based verification of hardware. *Programming and Computer Software*, 2011, vol. 37 (3), pp. 147–152.