Тестирование системы автоматов

*И.Б. Бурдонов, А.С. Косачев*

Институт системного программирования, Москва, Россия

igor@ispras.ru, kos@ispras.ru

Доклад посвящен тестированию составных систем, компоненты которых моделируются конечными автоматами, а взаимодействие между ними – обменом сообщениями по симплексным каналам связи. Структура связей между компонентами моделируется ориентированным *графом связей*, в вершинах которого находятся автоматы, а дуги соответствуют каналам связи и рассматриваются как очереди сообщений длины 1. Дуга *пуста*, если на ней нет сообщения. *Внутренняя* дуга связывает два автомата, *внешняя выходная* дуга ведет из системы в её окружение, а *внешняя входная* дуга ведет из окружения в систему. Обозначим через *xv#* (*yv#*) частично определённое отображение множества входных (выходных) дуг вершины *v* во множество сообщений, которое для каждой непустой дуги указывает сообщение на ней. Переход *(sv,xv,yv,tv)* автомата в вершине *v* ведёт из *пресостояния* *sv* в *постсостояние* *tv*, при этом указывается *стимул* *xv*, определяющий какие сообщения по каким входным дугам принимаются, и *реакция* *yv*, определяющая какие сообщения по каким выходным дугам посылаются. Переход выполним, если автомат находится в состоянии *sv*, *xv⊆x#v*и ***Dom****(yv)∩****Dom****(y#v)=∅*. Автомат детерминирован, если 1) пресостояние *sv* и стимул *xv* однозначно определяют реакцию *yv* и постсостояние *tv*, 2) для любого *xv#* в каждом состоянии *sv* есть переход не более чем по одному стимулу *xv⊆x#v*.

Композиция (по графу связей) детерминированных автоматов – это автомат *(S,X,Y,T,s0)*, отражающий работу системы в целом. Его состояние – это набор *s = (s1,s2,...,sk,D)*, где *s1,s2,...,sk* – это состояния всех автоматов системы, а *D* – частично-определённое отображение множества дуг графа связей во множество сообщений, которое для каждой непустой дуги указывает сообщение на ней. В начальном состоянии системы все автоматы находятся в своих начальных состояниях, а сообщений на дугах нет. Определение переходов композиции зависит от режима работы. В синхронном режиме за один такт срабатывают все автоматы, которые могут выполнить переход, а в асинхронном – только один такой автомат (вообще говоря, подмножество автоматов), выбираемый недетерминированным образом. Поскольку нас интересуют только детерминированные системы, асинхронный режим не рассматривается. В композиции определяется переход *s⎯?x!y→t*, где *x* (*y*) – частично-определённое отображение, которое для некоторых внешних входных (выходных) дуг указывает сообщения, которые по этим дугам посылает (принимает) окружение, при условии, что ***Dom****(x)∩****Dom****(D)=∅* и *y ⊆ D*, а *t = (t1,t2,...,tk,D^)*. Для *v*-го детерминированного автомата реакция *yv* и постсостояние *tv* однозначно определяются *s* и *D*, а *D^ = (D ∪ x ∪ y1 ∪ ... ∪ yk) \ (y ∪ x1 ∪ ... ∪ xk)*. Такая композиция детерминированных автоматов детерминирована.

Цель тестирования – покрытие всех достижимых переходов автоматов-компонентов. Нам известно, каким должен быть каждый автомат с точностью до изоморфизма, и именно это проверяется. Тест посылает сообщения по некоторым пустым внешним входным дугам и принимает сообщения с некоторых занятых внешних выходных дуг. Кроме того, он «видит» состояния автоматов и сообщения на дугах. Такие предположения оправданы, например, при имитационном тестировании аппаратуры [1]. Само тестирование похоже на тестирование в контексте [2, 3, 4]. Предлагается алгоритм построения набора тестов, который является полным (проверяет все достижимые переходы автоматов-компонентов) при выполнении двух условий: 1) система детерминирована, 2) верна *гипотеза о связях*: в графе связей нет ошибок. Берём алгоритм генерации набора тестов (например, [5]), покрывающего все дуги композиционного автомата системы, и применяем *фильтрацию*, отбрасывая «лишние» тесты, не покрывающие новых переходов по сравнению с уже отобранным тестами.

Для любых чисел состояний автоматов-компонентов *n1,n2,...,nk* существует такая система, что время (в тактах), необходимое для покрытия всех достижимых переходов композиции, равно ***Ω****(n1n2...nk)*, а минимальное время, достаточное для покрытия всех достижимых переходов компонентов, равно ***O****(n1+n2+...+nk)*. Для *n1=n2=...=nk=n*, имеем ***Ω****(nk)* и ***O****(nk)*, т.е. экспоненциальное уменьшение времени тестирования.

Литература

1. *А. Камкин, М. Чупилко.* Обзор современных технологий имитационной верификации аппаратуры // Программирование. – 2011. – №3. – С. 42–49.

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 г.

3. *И.Б.Бурдонов.* Теория конформности (функциональное тестирование программных систем на основе формальных моделей). – LAP Lambert Academic Publishing, 2011. – 428 с.

4. *И.Б.Бурдонов, А.С.Косачев.* Пополнение спецификации для ioco // Программирование. – 2011. – №1. – С. 3–18.

5. *И.Б. Бурдонов, А.С. Косачев, В.В. Кулямин.* Неизбыточные алгоритмы обхода ориентированных графов. Детерминированный случай // Программирование. – 2003. – №5. – С. 59–69.