Слайд 1. Название доклада.

Тестирование системы автоматов с буферизацией сообщений.

Слайд 2. Постановка задачи: тестирование составных систем

Большинство сложных, особенно распределённых, систем представляет собой набор взаимодействующих компонентов.

В этом докладе компоненты моделируются конечными автоматами, а взаимодействие – обменом сообщениями между автоматами. Рассматривается задача тестирования таких систем.

Слайд 3. Граф связей

Мы будем предполагать, что структура связей между компонентами моделируется ориентированным графом (будем называть его *графом связей*), в вершинах которого находятся автоматы, а дуги соответствуют симплексным каналам передачи сообщений.

Кроме *внутренних* дуг, то есть дуг, соединяющих автоматы между собой, существуют также *внешние* дуги, связывающие систему с её *окружением*.

Если такая дуга ведёт из окружения в некоторый автомат, то будем называть её *внешней входящей* дугой, а если дуга ведёт из автомата в окружение, то – *внешней выходящей* дугой.

Окружение взаимодействует с системой, выдавая сообщения на внешние входящие дуги системы и принимая сообщения с внешних выходящих дуг системы.

Мы будем считать, что граф связей статический, то есть не меняющийся в процессе работы системы. В этом случае система (также как её компоненты) может моделироваться конечным автоматом, получающимся из автоматов-компонентов с помощью подходящего оператора композиции, учитывающего граф связей.

Более того, мы будем предполагать, что граф связей системы «правильный» и совпадает с графом связей заданным спецификацией с точностью до изоморфизма. Это мы назвали *гипотезой о связях*.

Слайд 4. Три составляющих тестирования

Любое тестирование опирается на имеющиеся возможности по воздействию на систему и наблюдению её поведения, а также на критерий правильности поведения системы, который задаётся с помощью спецификации.

Что касается управления, то будем считать, что тест подменяет собой окружение. Это означает, что при тестировании управление сводится к выдаче сообщений по внешним входящим дугам и приёму сообщений с внешних выходящих дуг. Никаким другим способом мы на систему воздействовать не можем.

Что касается наблюдения, то будем считать, что в процессе тестирования мы можем наблюдать как состояния автоматов, так и сообщения, передаваемые по дугам графа связей.

Что касается критерия правильности, то будем считать, что спецификация однозначно определяет, каким должен быть автомат каждого компонента, т.е. задаёт граф переходов автомата. При тестировании проверяется изоморфизм реализационного и спецификационного автоматов.

Прежде всего, следует отметить, что такие предположения могут быть оправданы на практике. Примером может служить имитационное тестирование аппаратуры (simulation-based verification).

В этом месте нужно сказать несколько слов о конформности. Мы специально не изучали этот вопрос, но можно предположить, что тот подход к тестированию, о котором я буду рассказывать, применим, быть может, с некоторыми модификациями, не только для проверки изоморфизма автоматов, но и других конформностей.

Может быть, здесь важно только то, что при том же самом графе связей конформность компонентов эквивалентна конформности системы.

Это верно не для всех конформностей. Существует, так называемая, проблема сохранения конформности при композиции, когда композиция конформных реализаций компонентов оказывается неконформной спецификации системы. Возможная несогласованность спецификации системы со спецификациями

компонентов называют ещё проблемой декомпозиции системных требований.

Но я не буду дальше на этом останавливаться, и ещё раз повторю: при тестировании проверяется изоморфизм автомата реализации и автомата спецификации для каждого компонента системы.

Слайд 5. Цель тестирования и гипотеза о связях

Как следствие этого, если гипотеза о связях верна, то целью тестирования становится покрытие переходов автоматов компонентов.

Здесь следует сразу оговориться, что все переходы всех автоматов мы проверить, скорее всего, не сможем. Проблема в том, что каждый автомат может тестироваться только как часть системы. Это означает, что тест не имеет непосредственного доступа к автомату, и вынужден осуществлять тестовые воздействия с помощью сообщений, выдаваемых по внешним входящим дугам, которые ведут, быть может, в другие автоматы. Поэтому речь должна идти только о *достижимых* переходах автоматов, то есть таких переходах, которые могут быть выполнены при работе автомата как части данной системы.

Тестирование компонента такой системы похоже на тестирование в контексте, когда этот компонент рассматривается как тестируемая система, а остальные – как контекст. Существенное отличие, однако, в том, что в таком контексте тоже могут быть ошибки, хотя, если верна гипотеза о связях, то только в компонентах, а не в структуре связей между ними. С другой стороны, наше тестирование может проверять работу сразу нескольких компонентов, через которые проходят сообщения.

Тестирование автомата системы, который получается композицией автоматов компонентов для заданного графа связей, обеспечивает проход по всем достижимым переходам этого автомата системы. При этом, конечно, проверяются все достижимые переходы автоматов компонентов, но, вообще говоря, делается много «лишней работы».

Основная идея доклада в том, что гипотеза о связях позволяет получить существенный выигрыш во времени тестирования.

Дело в том, что число состояний системы не меньше произведения числа состояний компонентов, тогда как для проверки всех переходов всех компонентов может оказаться достаточно суммы этих чисел. При равенстве этих чисел и фиксированном количестве компонентов мы получаем экспоненциальное уменьшение времени тестирования. Пример этого я покажу попозже.

В докладе будет предложен алгоритм тестирования детерминированных систем.

Но сначала давайте формально определим систему автоматов, определим детерминизм системы и выясним условия детерминизма.

Слайд 6. Граф связей

Сначала определим формально граф связей.

Вершины мы перенумеруем, начиная с 0.

Номер 0 будет соответствовать окружению системы.

Дуга задаётся начальной и конечной вершиной, а для различения кратных дуг будем помечать дугу символом из некоторого алфавита пометок.

Внешняя входящая дуга – это дуга, ведущая из вершины 0, т.е. из окружения.

Внешняя выходящая дуга – это дуга, ведущая в вершину 0, т.е. в окружение.

Будем предполагать, что в вершине 0, соответствующей окружению, нет петель.

Вот здесь на слайде ещё введены обозначения, которые нам потом понадобятся. Это множество входящих дуг вершины и множество выходящих дуг вершины.

Слайд 7. Дуга

В этом докладе мы будем считать, что дуга устроена очень просто – это очередь сообщений длины 1.

Если дуга пуста, т.е. на ней нет сообщения, с неё нельзя принять сообщение, но по ней можно выдать сообщение.

Если дуга не пуста, т.е. на ней есть сообщение, с неё можно принять это сообщение, но по ней нельзя выдать сообщение.

Слайд 8. Автомат

Под автоматом будем понимать набор из множества стимулов, т.е. входных символов, множества реакций, т.е. выходных символов, конечного множества состояний, конечного множества переходов и выделенного начального состояния автомата.

Переход ведет из пресостояния в постсостояние и помечен стимулом и реакцией.

Слайд 9. Автомат в вершине

Для того чтобы такой автомат мог работать в вершине графа связей, принимая и выдавая сообщения по дугам, нужно, чтобы стимул автомата описывал принимаемые сообщения, а реакция описывала выдаваемые сообщения.

Пусть задано конечное множество сообщений.

Стимулом у нас будет частичное отображение множества входящих дуг во множество сообщений. Если входящая дуга не принадлежит домену стимула, это значит, что автомат не принимает сообщение с этой дуги. Если же дуга *i* принадлежит домену стимула *x*, то *x(i)* – это то сообщение, которое автомат должен принять с этой дуги.

Реакцией у нас будет частичное отображение множества выходящих дуг во множество сообщений. Если выходящая дуга не принадлежит домену реакции, это значит, что автомат ничего не выдает по этой дуге. Если же дуга *j* принадлежит домену реакции *y*, то *y(j)* – это то сообщение, которое автомат должен выдать по этой дуге.

Слайд 10. Условие выполнения перехода

Для того чтобы определить формально условие выполнения перехода автомата, определим состояния входящих и выходящих дуг вершины, в которой находится автомат.

Они тоже задаются частичными отображениями множества входящих или выходящих дуг во множество сообщений.

x с решёткой каждой непустой входящей дуге ставит в соответствие сообщение на этой дуге, а *y* с решёткой каждой непустой выходящей дуге ставит в соответствие сообщение на этой дуге.

Состояния входящих и выходящих дуг однозначно определяют множества потенциальных стимулов и реакций, т.е. таких стимулов и реакций, которыми может быть помечен выполнимый переход. Они обозначены как *X* большое с решёткой и *Y* большое с решёткой.

Соответственно, условием выполнения перехода является принадлежность стимула множеству потенциальных стимулов, а реакции – множеству потенциальных реакций.

Говоря попросту, переход может быть выполнен, если он помечен такими стимулом и реакцией, что согласно стимулу принимаются с входящих дуг только такие сообщения, которые находятся на этих дугах, а согласно реакции выдаются сообщения только на такие выходящие дуги, которые пустые.

Здесь нужно сделать важное замечание. В условие выполнимости перехода входит пустота тех выходящих дуг, по которым согласно реакции на переходе должны выдаваться сообщения. Пустота или непустота тех или иных выходящих дуг описывается доменом отображения *y* маленькое с решёткой. Так что это, по сути, является расширением входного символа автомата. Это нужно учитывать для понимания того, как работает автомат.

Слайд 11.Детерминированный автомат

Для того чтобы система автоматов была детерминирована, потребуем детерминированности каждого из этих автоматов.

Прежде всего, в детерминированном автомате пресостояние и стимул должны однозначно определять реакцию и постсостояние перехода.

Во-вторых, состояние входящих дуг должно определять не более одного выполнимого перехода в каждом состоянии автомата. Иначе выбор между этими переходами придётся делать недетерминированно. Поскольку состояние входящих дуг x маленькое с решёткой однозначно определяет множество потенциальных стимулов X большое с решёткой, в состоянии автомата должно быть определено не более одного перехода по стимулам из этого множества X большое с решёткой.

Формально это описывается так. Будем говорить, что два стимула *совместимы*, если при некоторых сообщениях на входящих дугах, т.е. при некотором отображении x маленькое с решёткой, могут оказаться выполнимы переходы как с одним, так и с другим стимулом. Это означает, что стимулы совместимы, если они совпадают на пересечении их доменов.

Поэтому второе требование детерминизма автомата запрещает наличие переходов из одного пресостояния по разным совместимым стимулам.

Если выполнены оба требования детерминизма, то состояние автомата и отображения x маленькое с решёткой и y маленькое с решёткой однозначно определяют, выполняет ли автомат какой-либо переход, и, если выполняет, то сам переход, т.е. однозначно определяют принимаемый стимул, выдаваемую реакцию и постсостояние.

Слайд 12. Композиция системы (1)

Определим композицию детерминированных автоматов с данным графом связей. Результатом композиции будет автомат, отражающий работу системы в целом, включая все автоматы-компоненты и все дуги.

Состояние системы – это набор состояний всех автоматов, а также описание расположения сообщений на дугах. Такое расположение задаётся частично-определённым отображением D, которое для каждой непустой дуги указывает находящееся на ней сообщение.

Начальным состоянием системы будем считать состояние, в котором каждый автомат находится в своём начальном состоянии, а сообщений на дугах нет.

Переходы композиции определяются в зависимости от того, предполагается ли синхронный или асинхронный режим работы автоматов. В каждом состоянии системы имеется множество автоматов, которые могут выполнять переходы.

В синхронном режиме за один такт срабатывают все автоматы из этого множества, а в асинхронном – только один автомат (вообще говоря, некоторое подмножество), выбираемый недетерминированным образом.

Поскольку нас интересуют только детерминированные системы, асинхронный режим мы не рассматриваем.

Слайд 13. Композиция системы (2)

Определим переходы композиции формально. В текущем состоянии системы окружение может выдать в систему сообщение по любой пустой внешней входящей дуге, а также принять сообщение с любой занятой внешней выходящей дуги. Отображение D как часть состояния системы определяет допустимые стимулы и реакции.

В частности, всегда допустимы пустой стимул, когда окружение никакие сообщения не помещает на внешние входящие дуги, и пустая реакция, когда никакие сообщения не принимаются окружением с внешних выходящих дуг.

Если стимул и реакция допустимы, то в композиции определяется переход по ним. Определим его постсостояние, т.е. постсостояние каждого автомата и изменённое расположение сообщений на дугах.

Сначала рассмотрим автомат, находящийся в некоторой вершине v. Для заданного пресостояния системы состояния входящих и выходящих дуг вершины v однозначно определяются как сужения отображения D на множество, соответственно, входящих и выходящих дуг. Тем самым, вместе с пресостоянием детерминированного автомата в вершине v, которое является частью пресостояния системы, это однозначно определяет переход в этом автомате.

Тем самым, мы знаем постсостояние каждого автомата, а также знаем, с каких дуг сообщения принимаются автоматами, и на какие дуги сообщения выдаются автоматами. Это однозначно определяет постсостояние системы.

Слайд 14. Композиция системы (3)

Итак, мы построили композиционный автомат системы, в котором для данного пресостояния системы каждая пара из допустимого стимула и допустимой реакции однозначно определяют переход, т.е. постсостояние системы.

Вот такую систему мы и будем называть детерминированной.

Здесь нужно сделать три замечания.

Первое замечание. Наш автомат системы отличается от автоматов в вершинах. Дело в том, что автомат вершины привязан к входящим и выходящим дугам, которые не являются частью самого автомата. В отличие от этого в системе внешние входящие и выходящие дуги «погружены» внутрь получившегося автомата системы.

Для автомата в вершине стимул и реакция – это наборы сообщений, которые принимаются с части входящих дуг и выдаются на часть выходящих дуг. А для автомата системы приём сообщений с внешних входящих дуг и выдача сообщений на внешние выходящие дуги – это часть «внутренней» работы системы. Для системы стимул – это набор сообщений, которые окружение помещает на пустые внешние входящие дуги, а реакция – это набор сообщений, которые окружение принимает с занятых внешних выходящих дуг.

Поэтому автомат системы не может рассматриваться как автомат в вершине. Это означает, что определённая нам композиция системы обладает недостатком: при такой композиции система автоматов не может использоваться как компонент более сложной системы. Другое дело, что мы пока и не ставили себе такой задачи. Но на будущее это не очень хорошо.

Второе замечание. Получившееся определение детерминизма системы – это классическое определение наблюдаемого недетерминизма, когда пресостояние и стимул могут неоднозначно определять реакцию, но вместе с реакцией однозначно определяют постсостояние.

Для наших целей тестирования этого вполне достаточно. Дело в том, что выбор той или иной допустимой реакции осуществляется не системой автоматов, а окружением. Это окружение решает, с

каких занятых внешних выходящих дуг принимать сообщения, а с каких не принимать.

При тестировании, когда тест подменяет собой окружение, такой выбор делает тест. И, если тест ведёт себя детерминированно, то и выбор будет детерминирован. Иными словами, композиция детерминированной (в указанном смысле) системы и детерминированного теста будет детерминирована.

Вот сейчас мы об этом и поговорим.

Но сначала ещё одно третье замечание. Я уже сказал, что для системы реакция – это набор сообщений, которые окружение или тест принимает с занятых внешних выходящих дуг. При этом с какой-то занятой внешней выходящей дуги сообщение принимается тестом, и дуга становится пустой, а с какой-то – не принимается, и она остаётся занятой.

Это, естественно, влияет на поведение системы, поскольку автоматы могут выдавать сообщения только на пустые выходящие дуги, в том числе на внешние дуги. Поэтому приём тестом реакции от системы можно рассматривать как дополнительное тестовое воздействие на систему.

В общем, здесь та же ситуация, что с автоматом компонента системы, для которого признаки пустоты или непустоты выходящих дуг, по сути, расширяют входной символ автомата.

Если учитывать это дополнительное тестовое воздействие на систему, то наблюдаемый недетерминизм системы становится кажущимся. На самом деле поведение системы вполне детерминировано стимулом и реакцией, понимаемой как такое дополнительное тестовое воздействие.

Слайд 15.Тест

При тестировании на каждом такте тест выдает в тестируемую систему сообщения по пустым внешним входящим дугам (не обязательно всем) и принимает от системы с занятых внешних выходящих дуга (не обязательно со всех) имеющиеся на них сообщения.

Определим композицию системы и теста. Состояние композиции – это пара состояний системы и теста. Переход композиции соответствует паре из допустимого стимула и допустимой реакции, что определяет возможные постсостояния системы и теста, т.е. возможные постсостояния композиции. Сам переход композиции является внутренним, т.е. ничем не помечен, поскольку композиция системы и теста замкнута и ни с чем не взаимодействует.

Формально в композиции системы и теста переходы определяются обычным правилом композиции, изображённом на слайде.

Детерминированный тест – это тест, который в каждом состоянии осуществляет только одно тестовое воздействие из множества допустимых. Поэтому тест выбирает только один допустимый стимул, так это обычно и делается. Но, кроме этого, выбирается только одна допустимая реакция, что несколько необычно. Это объясняется тем, что выбор реакции является дополнительным тестовым воздействием.

Если тест и система оба детерминированы, то их композиция тоже будет детерминирована в следующем смысле: в каждом её состоянии определено не более одного перехода.

Тестовая последовательность – это конечная последовательность пар из допустимых стимула и реакции. В тестируемой системе и в тесте этой последовательности соответствуют маршруты – цепочки смежных переходов. Для такой тестовой последовательности детерминированной тест состоит только из одного маршрута.

Слайд 16. Тестовые проверки

Какие проверки делает тест и как выносится вердикт?

На каждом такте проверяется, в какое состояние перешла система.

Если система детерминирована, то это состояние однозначно определяется тестовой последовательностью.

Поскольку в качестве конформности мы выбрали изоморфизм, мы проверяем, совпадает ли состояние системы с тем, которое определяет спецификация.

Для каждого автомата в системе проверяется:

- правильно ли изменилось его состояние,
- правильно ли он выполнил приём стимула – с тех или не тех входящих дуг принял сообщения,
- и правильно ли он выполнил выдачу реакции – на те или не на те выходящие дуги выдал сообщения, и правильные ли эти сообщения.

Слайд 17. Покрытие переходов. Прогон теста. Полнота.

Прогон теста покрывает некоторое множество переходов автоматов компонентов системы.

Поскольку система детерминирована, это множество одно и то же при разных прогонах данного теста (с рестартом между прогонами), поэтому тест достаточно прогонять один раз.

Мы будем рассматривать только конечные наборы тестов.

После завершения прогона одного теста выполняется рестарт системы и прогоняется следующий тест из набора.

Набор тестов покрывает множество переходов компонентов, которое является объединением множеств переходов компонентов, покрываемых тестами из этого набора.

Набор тестов будем называть *полным*, если он покрывает все переходы всех компонентов, достижимые при работе этих компонентов в системе.

Ставится задача генерации полного набора тестов.

Слайд 18. Алгоритм фильтрации.

Для решения этой задачи предлагается использовать любой алгоритм генерации полного набора тестов для одного автомата. Таких алгоритмов предложено довольно много, по сути, они сводятся к построению набора маршрутов, покрывающих граф переходов автомата, достижимых из его начального состояния.

В качестве такого автомата для наших целей выбирается автомат системы, получаемый с помощью композиции, о которой я рассказывал.

Здесь важно отметить, что композиционный автомат системы вполне определен по допустимым стимулам и реакциям, то есть в каждом достижимом состоянии определён переход по каждой паре допустимых стимула и реакции.

Поэтому ошибка – это такой переход по допустимому стимулу и допустимой реакции, при котором неверным оказывается постсостояние системы.

Понятно, что покрывая все достижимые переходы композиционного автомата системы, мы покрываем все достижимые переходы автоматов компонентов. Однако такой набор тестов может оказаться сильно избыточным для решения нашей задачи: покрытие всех достижимых переходов автоматов компонентов не обязательно требует покрытия всех достижимых переходов композиционного автомата системы.

Поэтому предлагается в процессе генерации полного набора тестов для композиционного автомата системы применять *процедуру фильтрации*, которая будет отбрасывать «лишние» тесты. Эта процедура работает следующим образом.

С самого начала создаётся пустое множество T генерируемого набора тестов и множество P непокрытых переходов автоматов компонентов, которое сначала равно множеству всех переходов всех автоматов компонентов.

Когда для композиционного автомата системы генерируется очередной i-ый тест T_i, вычисляется множество P_i переходов автоматов компонентов, покрываемое этим тестом. Тесту соответствует маршрут в композиционном автомате. Каждому переходу этого маршрута соответствует не более одного перехода

каждого из автоматов компонентов; множество таких переходов и есть множество P_i.

Далее алгоритм фильтрации проверяет, покрывает ли i-ый тест какой-либо новый, ещё не покрытый переход какого-либо автомата компонента. Если $P_i \cap P = \varnothing$, то никаких новых переходов i-ый тест не покрывает, и он отбрасывается. В противном случае тест добавляется к набору тестов $T := T \cup \{T_i\}$, а из множества непокрытых переходов удаляются новые переходы $P := P \setminus P_i$.

После того как все тесты сгенерированы и отфильтрованы, получившееся множество T является полным набором тестов для системы автоматов при выполнении гипотезы о связях, а множество P – множеством недостижимых переходов автоматов компонентов. Это дополнительный результат алгоритма.

Слайд 19.Пример

На этом слайде показан пример, который иллюстрирует ту выгоду, которую даёт при тестировании гипотеза о связях. Иными словами, он показывает, как может уменьшиться время тестирования, если нам достаточно проверить каждый переход каждого компонента, а не все переходы композиционного автомата системы.

В этом примере имеется только один тип сообщения – это число 1.

У каждой вершины i две входящие дуги a_i и b_i, и одна выходящая дуга a_{i+1}.

i-ый автомат имеет n_i состояний. Принимая «1» по дуге a_i, он реализует функцию прибавления 1 к своему состоянию в системе модульной арифметики по модулю n_i. При переполнении, когда автомат прибавляет «1» к своему максимальному состоянию n_i-1, он возвращается в начальное нулевое состояние, а по выходящей дуге тоже выдается «1».

Набор состояний автоматов можно понимать как число, записанное в системе счисления с переменным основанием: в i-ом разряде основание равно n_i. Число максимально, когда каждый автомат находится в своём максимальном состоянии n_i-1. Если к нему прибавить «1», все автоматы системы возвратятся в свои начальные нулевые состояния.

Подавая «1» на внешнюю входящую дугу a_1, мы заставляем систему выполнять прибавление 1 к своему состоянию в такой системе счисления.

Понятно, что число состояний системы не меньше произведения чисел состояний компонентов. Поскольку все эти состояния достижимы и в каждом из них имеется переход, время тестирования не меньше этого произведения.

В то же время, когда i-ый автомат принимает «1» по входящей дуге b_i, он не меняет своего состояния, но выдает «1» дальше по дуге a_{i+1}. Это даёт возможность покрыть все переходы автоматов за время порядка суммы чисел их состояний.

Если все автоматы имеют одинаковое число состояний n, то при фиксированном количестве автоматов k, мы получаем экспоненциальный выигрыш во времени тестирования.

Слайд 20. Недостатки и нерешённые проблемы.

В заключение я перечислю основные недостатки предложенной модели системы автоматов и генерации тестов, а также нерешённые проблемы.

1. Композиционный автомат системы нельзя использовать как компонент более сложной системы. Я об этом уже говорил.

2. В предложенной модели дуга – это очередь длины 1. Нужно рассмотреть возможность обобщения модели так, чтобы дуга могла реализовывать очередь другой длины, очередь с приоритетами, стек и более общие модели буферизации. В общем, дуга – это некоторый автомат-посредник между автоматами, находящимися в вершинах.

3. Предложенная автоматная модель компонента не позволяет моделировать многие важные виды поведения компонентов. В целом это можно назвать условным приёмом сообщений и условной выдачей сообщений.

   Примером условного приёма может служить компонент, который принимает только те сообщения, которые удовлетворяют некоторому критерию. Например, сообщение – это число, и выбирается максимальное число. Для того чтобы так работать, компоненту нужно сначала «узнать», какие сообщения можно принимать со входящих дуг, а уже потом решать, какие из них принимать, а какие нет.

   Условная выдача сообщений означает, что выдаются на выходящие дуги те сообщения, которые эти дуги готовы принять. Например, если нужно выдать два сообщения на две выходящие дуги как очереди длины 1, а только одна из них пуста, то выдается сообщение на эту дугу, а на другую не выдается. Может быть и другое поведение компонента: если выполняется переход, то одно сообщение должно выдаться, а другое – как получится. В предложенной модели автомат работает по принципу «всё или ничего»: если он детерминирован, то либо передаются оба сообщения, либо не выполняется никакого перехода. Для того чтобы реализовать условную выдачу, компоненту нужно сначала

«узнать», какие выходящие дуги какие сообщения могут принять, а уже потом решать, на какие из них выдавать сообщения.

4. Алгоритм фильтрация, к сожалению, не обязательно даёт оптимальный полный набор тестов. Даже в том примере, который я приводил, фильтрация вовсе не обязательно даст тот набор тестов, который гарантирует время тестирования порядка суммы чисел состояний компонентов.

   Оптимальность естественно понимать как минимальное время тестирования, то есть как минимальную суммарную длину тестовых последовательностей. Соответственно, возникает задача оптимизации, т.е. поиска оптимального набора, которая, вообще говоря, сводима к задаче о поиске минимального покрытия.

5. К нерешённым проблемам относится также тестирование системы недетерминированных автоматов.

6. Ещё было бы интересно посмотреть, как можно применять предложенный или аналогичный подход к тестированию системы автоматов для других конформностей. Например, для эквивалентности автоматов, симуляции, редукции, конформностей с отказами и других.

Первые три из этих недостатков мы постараемся исправить во втором докладе.

Ну, а последние три относятся к классу пока нерешённых проблем.