Опубликовано в материалах XV Международной научно-методической конференции «Высокие интеллектуальные технологии и инновации в образовании и науке». СПбГ ПУ. 2008, с. 293–296.

Е. А. Курбацкий, А. А. Шалыто

ВЕРИФИКАЦИЯ ПРОГРАММ, ПОСТРОЕННЫХ НА ОСНОВЕ АВТОМАТНОГО ПОДХОДА

В последнее время активно начинает применяться метод верификации программных систем, основанный на моделях (*Model checking*). При использовании этого подхода строится модель с конечным числом состояний. Требования к модели выражаются на языке темпоральной логики. Модель и требования подаются на вход верификатора. Верификатор автоматически проверяет требования к модели, и в случае не выполнения требования выдает контрпример, который указывает последовательность состояний модели, при которой нарушается проверяемое требование.

При традиционном подходе [1] построение модели, задание требований к ней и разбор контрпримера осуществляются вручную. Подобный подход имеет недостатки: выполнение операций вручную требует больших затрат времени и при построении модели могут быть допущены ошибки. Автоматизация хотя бы некоторых этапов верификации позволяет повысить ее эффективность. Однако для программ общего вида это не сделать.

В работе [2] рассматривается автоматный подход применительно к построению реактивных систем. Этот подход с точки зрения построения модели имеет ряд преимуществ по сравнению с традиционным подходом. При построении модели для программы, написанной с применением традиционного подхода, возникает проблема, будет ли модель адекватна исходной программе. При автоматном программировании такая проблема исключена, так как набор взаимодействующих автоматов уже является моделью, которая изоморфна поведенческой части автоматной программы. Эта модель имеет конечное число

управляющих состояний (при любом числе вычислительных состояний) [3], что является необходимым условием для её верификации.

В работе [4] описан способ проверки одного автомата. Однако, при построении реальных систем обычно используется большее их число. В работе [5] рассматривается иерархическая модель автоматов, но её построение осуществляется вручную. Поэтому возникает задача автоматизированной проверки требований к системе автоматов.

В настоящей работе для систем автоматов автоматизируются построение модели и преобразование контрпримера, построенного для модели, в контрпример для программы. При этом верификация сводится к следующим подзадачам: преобразовать систему автоматов в модель; преобразовать требования к системе в требования к модели; проверить модель; контрпример к модели (при его наличии) преобразовать в контрпример в системе автоматов.

Для верификации модели в настоящей работе используется одно из открытых и бесплатных средств – символьный верификатор моделей *NuSMV* [6, 7] (аббревиатура «*New Symbolic Model Verifier*»). В качестве языка для описания модели используется язык *SMV*. Это инструментальное средство предназначено для проверки того, что система переходов с конечным числом состояний удовлетворяет требованиям, заданным на языке темпоральной логики *CTL*. В нем применяется описанный в работе [1] символьный алгоритм верификации моделей, основанный на упорядоченных двоичных диаграммах решений (*OBDD*).

Модель является системой переходов, каждое состояние которой задается набором переменных, в которых записываются состояния автоматов, их действия, а также события, которые им посылаются.

Проверка системы автоматов выполняется следующим образом.

1. Формируются требования к системе автоматов. Вручную выполняется их преобразование в требования к модели, записываемые формулами логики *CTL*.

2. Система автоматов и требования к модели записываются в файле в *XML*-формате. Для этого используется разработанная авторами программа-редактор, которая позволяет задать систему автоматов в виде графов переходов и требования к ней в виде формул темпоральной логики *CTL*.

3. Запускается разработанная авторами программа верификации системы автоматов. Ей на вход подается файл с системой автоматов и требованиями, полученный на предыдущем шаге. Она генерирует модель на языке *SMV*.

4. Указанная выше программа запускает верификатор *SMV,* которому на вход подаются модель, полученная на предыдущем шаге, и требования на указанных языках, а на выходе формируется информация о выполненных требованиях и контпримерах (при их наличии).

5. В случае, если верификатор выдал контрпример к модели, то написанная авторами программа преобразует его в контрпример к системе автоматов.

Предлагаемый метод проверяет систему автоматов со следующими свойствами. Каждый автомат является автоматом Мили. Автоматы могут взаимодействовать, получая информацию о состояниях других автоматов и вызывая другие автоматы, передавая им управление за счет формирования событий.

Переход автомата может помечаться событием и условием, при которых он происходит, а также последовательностью действий.

При описании требований метод позволяет проверять следующие свойства состояний системы: в каком состоянии будет находиться автомат или несколько автоматов, на каком переходе будет находиться вызываемый автомат и какие значения принимают входные переменные при выполнении перехода.

Как показали эксперименты, предложенный подход позволяет верифицировать системы, содержащие 10 – 15 автоматов по три–четыре состояния в каждом. Следовательно, метод может применяться для достаточно большого круга задач, построенных на основе автоматного подхода.

При проверке систем с большим числом автоматов время работы быстро возрастает.

ИСТОЧНИКИ

1. *Кларк Э., Грамберг О., Пелед Д.* Верификация моделей программ: *Model Checking*. М.: МЦНМО. 2002. http://is.ifmo.ru/verification/_klark_gamberg_pered_verification.djvu
2. *Шалыто А. А.* SWITCH-технология. Алгоритмизация и программирование задач логического управления. СПб.: Наука. 1998. http://is.ifmo.ru/books/switch/1
3. *Туккель Н. И., Шалыто А. А.* От тьюрингова программирования к автоматному //Мир ПК. 2002. № 2, с. 144–149. http://is.ifmo.ru/works/turing/
4. *Вельдер С. Э., Шалыто А.А.* Введение в верификацию автоматных программ на основе метода Model checking. СПбГУ ИТМО. 2006. http://is.ifmo.ru/verification/modelchecking/
5. *Кузьмин Е. В.* Иерархическая модель автоматных программ // Моделирование и анализ информационных систем. 2006. № 1, с. 27–34. http://is.ifmo.ru/verification/_hamp.pdf
6. *Symbolic Model Verifier*. http://www.cs.cmu.edu/~modelcheck/smv.html
7. *New Symbolic Model Verifier*. http://nusmv.irst.itc.it/