М.Л. ГРОМОВ, А.В. ЛАПУТЕНКО**

ПРОВЕРКА СВОЙСТВ КОНЕЧНЫХ АВТОМАТОВ С ПОМОЩЬЮ НАБОРА ИНСТРУМЕНТОВ MCRL2

В данной работе рассматривается применение набора инструментов mCRL2 и одноименного языка для проверки основных свойств конечных автоматов, таких как детерминированность и полная определенность.

Ключевые слова: *конечный автомат, набор инструментов mCRL2, верификация.*

Введение

На основе автоматных моделей разработано большое количество методов анализа и синтеза дискретных систем. Большинство этих методов предполагает, что заданные модели обладают определёнными свойствами, иначе метод оказывается не применим (не гарантирует решение поставленной задачи). Поэтому, предварительно нужно проверить, выполняется то или иное свойство. В случае, когда проверяемая система представляет собой формальную модель, процесс проверки свойств принято называть верификацией. Существует множество инструментов верификации. Одним из таких инструментов (набором инструментов) является mCRL2. Выбор этого инструмента оправдывается тем, что он использует хорошо задокументированный язык описания процессов и свойств, активно поддерживается разработчиками и находится в свободном доступе.

Определения

Определение 1. LTS (Помеченная система переходов) – *это четверка объектов* $A = (S, Act, \rightarrow, s)$, *где* S *– непустое множество состояний с выделенным начальным состоянием* s. Act *– множество действий. Как обычно* $\tau \notin Act$ *обозначает внутреннее (не наблюдаемое) действие; множество* $Act \cup \{\tau\}$ *кратко записывается как* Act_τ. *Отношение* $\rightarrow \subseteq S \times Act_\tau \times S$ – *отношение переходов.* [1]

Далее приводятся определения, взятые из [2]

Определение 2. Конечным автоматом *(или просто* автоматом*) будем называть пятерку* $A = (S, I, O, \mu, s_0)$, *где* I *и* O *— конечные входное и выходное множества (алфавиты),* S *— конечное, непустое множество состояний с выделенным начальным состоянием* s_0, *и* $\mu \subseteq S \times I \times O \times S$ *— отношение переходов.*

Если четверка $(s, i, o, s') \in \mu$, то будем говорить, что под действием входного сигнала i автомат из состояния s переходит в состояние s' с выдачей выходного сигнала o.

Определение 3. *Состояние* s *автомата* $A = (S, I, O, \mu, s_0)$ *назовём* детерминированным, *если для любого входного символа* $i \in I$ *существует не более одной пары* $(o, s') \in O \times S$, *такой, что* $(s, i, o, s') \in \mu$, *в противном случае состояние* s *назовём* недетерминированным. *Автомат* A *будем называть детерминированным, если всякое его состояние детерминировано, и недетерминированным в противном случае*.

Иными словами, автомат называется детерминированным, если в любом его состоянии любой входной сигнал обрабатывается либо единственным образом, либо переход под действием этого входа неопределен. Очевидно, что детерминированные автоматы являются частным случаем недетерминированных.

Определение 4. *Автомат* $A = (S, I, O, \mu, s_0)$, *будем называть полностью определенным, если в каждом состоянии* $s \in S$, *определен переход по каждому входному символу* $i \in I$, *то есть найдутся такие* $o \in O$ *и* $s' \in S$, *что* $(s, i, o, s') \in \mu$.

Набор инструментов mCRL2

mCRL2 (micro Common Representation Language 2)[3] – это формальный язык спецификации с соответствующим набором инструментов. Набор инструментов можно использовать для моделирования, валидации и верификации параллельных систем и протоколов. Он может быть запущен на Windows, Linux, Apple Mac OS X и FreeBSD

Набор инструментов содержит инструменты для линеаризации, моделирования, генерации и исследования пространства состояний, и инструментов для оптимизации и анализа спецификаций систем. Кроме того, имеется возможность для визуализации, манипуляций и анализа пространства состояний.

Анализ системы часто призван показать, что моделируемая система обладает определенными желательными свойствами (или не обладает нежелательными). Это может быть осуществлено с использованием метода проверки модели (model checking), который является очень мощным методом верификации.

Для осуществления проверки свойств какой-либо системы методом проверки на модели (model checking) с помощью набора инструментов mCRL2, ее описание должно быть представлено на языке mCRL2 в виде текстового файла с расширением .mcrl2. Поскольку данное описание может отражать поведение параллельной системы, для дальнейшей работы данное описание преобразуется к линейному формату (LPS). Большинство инструментов из набора инструментов mCRL2 работают именно с файлами формата LPS. Вторым необходимым условием осуществления проверки любого свойства рассматриваемой системы является наличие формальной записи данного свойства в виде mCRL2 формулы. Данные формулы записываются с использованием модального μ-исчисления.

В mCRL2, проверка на модели обеспечивается за счет использования параметризованных систем булевых уравнений (PBES). Как упоминалось ранее, центральным понятием в mCRL2 является LPS. Проверка модели также начинается с файла LPS, который содержит символическую спецификацию поведения системы. Инструмент lps2pbes по имеющейся LPS и формуле строит PBES, которая хранится в двоичном формате. Решив данную параметризованную систему булевых уравнений, можно получить ответ на главный вопрос процедуры проверки модели «Выполняется ли данная формула для данной LPS?» Основным инструментом, производящим попытку решения PBES является pbes2bool. Он пытается найти решение PBES, и если это возможно, возвращает ответ в виде булевого значения: true либо false.

Следует отметить, что решение PBES в общем случае, неразрешимая задача, поэтому попытка может окончиться неудачей. В этом случае может понадобиться более детальное изучение рассматриваемой системы. Инструмент pbespp производит преобразование pbes в удобочитаемый формат. Статистическая информация может быть получена с помощью pbesinfo и PBES может быть упрощена с помощью pbesrewr. Кроме того, доступны некоторые инструменты для упрощения PBES, например, pbesparelm и pbesconstelm.

Язык описания систем mCRL2

Любая спецификация mCRL2 описывает процесс. Процессы используются для описания поведения некоторой системы или ее компонентов. Примитивной операцией процесса является действие. Действие представляет собой событие любого рода, будь то отправка сообщения, вывод текста на экран, либо рукопожатие. Если речь идет о процессах, содержащих данные в качестве параметров, то действия могут иметь аргументы. Действия могут комбинироваться, для описания более сложного поведения. Главными операторами являются последовательная композиция, или точка ., которая соединяет действия в последовательности, и оператор выбора +, представляющий выбор между двумя действиями.

Пример:

act a, b, c, d, e;

proc P = a.b.c;

Q = d + e;

R = (a + b).c.d.e;

init P;

В примере выше, P – это процесс, в котором действия a,b,c могут совершаться последовательно. В процессе Q есть выбор между между совершением действий d и e. Процесс R состоит из совершения выбора между действиями a и b и дальнейшего выполнения последовательности действий c, d и e.

Ключевое слово **init** означает, что общий процесс начнется с выполнения процесса P.

Формулирование свойств систем на языке модальной логики

Обычно, свойство отражает утверждение о части рассматриваемой системы, и конъюнкция некоторого количества таких свойств может дать представление обо всей системе в целом. Свойства могут быть выражены с помощью большого количества различных логик. Наиболее используемыми для этих средств являются: логика линейного времени(LTL) и логика деревьев вычислений (CTL). Логика CTL* включает в себя как LTL, так и CTL. Наиболее часто, благодаря своей выразительности, для выражения свойств используется модальное μ-исчисление.

Логика Хеннесси-Милнера (HML)

Модальной логикой, лежащей в основе μ-исчисления, является логика Хеннесси-Милнера. Все формулы этой логики выражают свойства состояний системы.

Формула логики Хэннесси-Милнера, определенная на множестве действий *Act*, задается с использованием следующего синтаксиса:

$\phi, \psi ::= true \mid false \mid \neg\phi \mid \phi \wedge \psi \mid \phi \vee \psi \mid \phi \Rightarrow \psi \mid \langle a\rangle\phi \mid [a]\phi$

где $a \in Act$.

Здесь, true – формула, выполняющаяся универсально, в каждом состоянии, fasle – формула, не выполняющаяся ни в одном из состояний. Отрицание, конъюнкция, дизъюнкция, импликация имеют свое обычное значение. $\langle a\rangle\phi$ и $[a]\phi$ модальности, означающие существование a-приемника, в котором ϕ выполняется, и для всех a-приемников ϕ выполняется, соответственно.

Формула HML либо выполняется в состоянии, либо нет. Принято говорить, что LTS удовлетворяет формуле HML, если формула выполняется в начальном состоянии LTS. Более формально, имея помеченную систему переходов $L = (S, Act, \rightarrow, s)$, состояние s, и HML формулу ϕ мы можем говорить, что s удовлетворяет ϕ, что обозначается как $L,s \models \phi$, если

$L,s \models \phi = true$ выполняется во всех состояниях,

$L,s \models \phi = false$ не выполняется ни в одном состоянии

$L,s \models \neg \phi$ если и только если, не верно $L,s \models \phi$

$L,s \models \phi \wedge \psi$ если и только если $L,s \models \phi$ и $L,s \models \psi$

$L,s \models \phi \vee \psi$ если либо $L,s \models \phi$ либо $L,s \models \psi$

$L,s \models \phi \Rightarrow \psi$ если и только если $L,s \models \phi$ имплицирует $L,s \models \psi$

$L,s \models \langle a\rangle \phi$ если и только если найдется $t \in S$, такое, что $(s, a, t) \in \rightarrow$ и $L,t \models \phi$

$L,s \models [a]\,\phi$ если и только если для всех $t \in S$, если $(s, a, t) \in \rightarrow$, *то* $L,t \models \phi$

Первое расширение логики Хэннесси-Милнера[1] производится для возможности использования множества действий вместо одного действия в модальностях. Для обеспечения этого, вводится понятие формулы действия.

Формула действия, определяемая над множеством действий *Act,* задается с помощью следующего синтаксиса:

A, B ::= false | true | a | ¬A | A ∪ B | A ∩ B

где a ∈ Act

Здесь, *false* означает пустое множество действий, *true* обозначает все действия, $a \in Act$ – действие, $\neg A$ означает любое действие из $Act\backslash A$, $A \cup B$ означает действия из A или из B, $A \cap B$ обозначает действия из A и из B.

Используя формулы действий, можно компактно записывать такие свойства, как например:

$\langle true \rangle true$ любое действие возможно

$\langle a \cup b \rangle [a \cup b] false$ после совершения действия a или b мы попадаем в состояние, в котором действия a и b не доступны.

В HML возможно создавать формулы лишь конечной длины. Следующие свойства не могут быть описаны в HML прямо, без привлечения бесконечных формул:

$Inv(\phi) = \phi \wedge [true]\phi \wedge [true][true]\phi \wedge \cdots$

$Pos(\phi) = \phi \vee \langle true \rangle \phi \vee \langle true \rangle \langle true \rangle \phi \vee \cdots$

Для решения подобной проблемы используется расширение HML в котором, внутри модальностей, вместо множества действий мы можем записать $\langle \beta \rangle$ и $[\beta]$, где β – регулярные выражения. Это позволяет более кратко формулировать свойства и выражать свойства систем, способных порождать бесконечные последовательности действий.

Регулярные выражения β определяются с использованием следующего синтаксиса:

$\beta 1, \beta 2 ::= \varepsilon \mid A \mid \beta_1 \cdot \beta_2 \mid \beta_1 + \beta_2 \mid \beta_1{}^*$

где A – формула действия, определенная ранее, ε обозначает пустую последовательность действий. Формула $\beta_1 \cdot \beta_2$ обозначает конкатенацию последовательностей действий из β_1 и β_2. $\beta_1 + \beta_2$ представляет собой выбор между последовательностями действий из β_1 и β_2. β^* обозначает любое, возможно нулевое, количество повторений последовательности действий из β_1.

Используемое в наборе инструментов mCRL2 μ-исчисление– это модальное μ-исчисление первого порядка расширенное процессами с данными и регулярными формулами[3].

Таким образом, синтаксис формул на языке mCRL2 состоит из формул действий, формул состояний, и регулярных формул.

Проверка свойств конечных автоматов

Для осуществления автоматической проверки некоторых свойств конечных автоматов, заданных в формате fsm, т.е. таблицей переходов, были программно реализованы алгоритмы написания формул для проверки детерминированности и полной определенности конечного автомата.

Поскольку набор инструментов mCRL2 в качестве исходного описания системы принимает модель LTS на собственном языке mCRL2, был разработан транслятор форматов fsm->mcrl2, преобразующий автоматное описание системы в описание помеченной системы переходов на языке mCRL2. Поскольку в модели помеченной системы переходов нет разделения символов на входные и выходные, для дальнейшей возможности отличить входные действия автомата от

выходных, мы помечаем все выходные действия помеченной системы переходов символом o, а входные – i.

Формула для проверки детерминированности автомата с 3 выходными символами, будет иметь следующий вид.

[true*](([o1]true =>([o2]false && [o3]false)) && ([o2]true =>([o1]false && [o3]false)) &&([o3]true =>([o1]false && [o2]false)))

Данная формула утверждает, что если в любом достижимом из начального состоянии системы определен хотя бы один выходной символ, то ни один из остальных выходных символов, определенных в этой системе, не определен в этом состоянии.

Формула для проверки полной определенности конечного автомата, например, с тремя входными символами, записывается следующим образом:

[true*](([i0]true =>([i1]true && [i2]true)) && ([i1]true =>([i0]true && [i2]true)) && ([i2]true =>([i0]true && [i1]true)))

Данная формула утверждает, что если в любом достижимом из начального состоянии определен хотя бы один входной символ, то все остальные входные символы, определенные в данном автомате, также определены в этом состоянии.

СПИСОК ЛИТЕРАТУРЫ

1. Jeroen J.A. Keiren. Modal μ-calculus (version 1.1), VU University Amsterdam, 2013
2. Н.В.Евтушенко, А.Ф.Петренко, М.В.Ветрова, Недетерминированные автоматы: анализ и синтез, Ч.1, Отношения и операции, Учебное пособие, Томск, ТГУ, 2006.
3. mCRL2 user manual [Электронный ресурс] : URL: http://mcrl2.org/release/user_manual/user.html
4. Giacomo Lenzi, THE MODAL μ-CALCULUS: A SURVEY, University of Pisa, Department of mathematics, Italy, Pisa, sep.2014

*Национальный исследовательский Томский государственный университет, г. Томск, Россия
E-mail: gromov@sibmail.com , aaandrey@sibmail.com

_____
Громов Максим Леонидович, к.ф.-м.н., доцент;
Лапутенко Андрей Владимирович, магистрант

M.L. GROMOV, A.V. LAPUTENKO

CHECKING FSM PROPERTIES USING MCRL2 TOOLSET

An application of mCRL2 toolset for checking FSM's properties, such as determinism and completeness, is proposed.

Keywords: *finite state machine, mCRL2 toolset, verification.*

REFERENCES

1. Jeroen J.A. Keiren. Modal μ-calculus (version 1.1), VU University Amsterdam, 2013
2. Yevtushenko N.V., Petrenko A.F., Vetrova M.V., *Nedeterminirovannye avtomaty: analiz i sintez, Ch.1, Otnoshenija i operacii* [Nondeterministic finite state machines: analysis and synthesis, Part 1, Relations and operations]. textbook, Tomsk, TSU, 2006.
3. mCRL2 user manual, available at: http://mcrl2.org/release/user_manual/user.html (Accessed 10.04.15)
4. Giacomo Lenzi, THE MODAL μ-CALCULUS: A SURVEY, University of Pisa, Department of mathematics, Italy, Pisa, sep.2014