Использование логики X-CTL для формальной верификации X-машин

М.С. Соболев

Аннотация. Приведен обзор логики CTL и ее применения для формальной верификации. Предложена логика X-CTL, позволяющая осуществлять формальную верификацию X-машин. Приведен пример модели-счетчика.

Ключевые слова: формальная верификация, проверка моделей, X-машина, сложные системы.

Введение

Формальная верификация[1] модели – это процесс использования формальных методов для проверки соответствия модели выбранным критериям. Эти критерии формируются исходя из свойств моделируемой системы и поставленной задачи. Таким образом, формальная верификация позволяет установить адекватность модели системе, которая моделируется. Формальная верификация X-машин является важной задачей, так как позволяет протестировать X-машины, входящие в сложную систему, что является необходимым критерием обеспечения правильной работы системы в целом [2].

В данной работе предложен новый способ формальной верификации X-машин при помощи так называемой логики X-CTL.

Логика CTL

Наиболее часто используемые виды формальной верификации – это дедуктивный анализ и проверка моделей.

Дедуктивный анализ [3] – это процесс доказательств желаемых свойств системы с использованием внутренних аксиом этой системы. При этом подходе и система и ее желаемые свойства задаются формулами некоторой логики, которая, в свою очередь, определяется набором аксиом и правил вывода. Существуют полуавтоматические системы для дедуктивного анализа, однако они не могут быть использованы для анализа широкого класса систем и требуют вмешательства человека в процессе своей работы. Таким образом, процесс дедуктивного анализа занимает много времени и подвержен ошибкам. К достоинствам метода можно отнести возможность работы с системами с бесконечным числом состояний. Недостаток общности данного метода не позволяет использовать его для верификации X-машин.

Проверка моделей [1] – подход к формальной верификации, использующий темпоральную логику. В рамках этого подхода система представляется в виде графа переходов между состояниями, а желаемые свойства в виде формулы темпоральной логики. Для того чтобы проверка моделей была возможной, удобно представить систему в виде машины состояний. Таким образом, как будет далее показано, метод проверки моделей можно применить к системам, смоделированным при помощи X-машин.

Логика предикатов не учитывает течение времени: верное утверждение всегда верно, а ложное утверждение всегда ложно. Эту особенность можно обойти, добавив к каждому элементарному выражению временной параметр, однако это приведет к неоправданному усложнению логических выражений. Кроме того выражения вида "в будущем возможно

событие A" или "свойство A будет верно до тех пор, пока верно свойство B" уже не будут элементарными.

Темпоральные логики [4] представляют собой обобщение логики предикатов операторами, использующими понятие времени (модальные операторы). С помощью этих операторов удобно конструировать логические выражения, которые верны в некий определенный момент времени. Время считается дискретным и определенным на сколь угодно большом промежутке как в прошлом, так и в будущем. В выражениях темпоральных логик используется не время как таковое, а временные отношения.

Используя модальные операторы, можно выписать выражение, означающее, что свойство верно во всей модели или начиная с некоторого состояния.

В работах [4] и [5] была предложена и описана темпоральная логика CTL (Computational Tree Logic), которая позволяет описывать системы с ветвлением вычислений. Выражения CTL позволяют представлять знания естественным образом, особенно в случае ветвящихся временных выражений.

В логике CTL вводятся пять временных операторов, определенных на свойствах, которые принимают некоторые значения вдоль пути вычислений (свойствами могут быть как атомарные высказывания, так и другие CTL-выражения):

- **Xp**: свойство p должно быть истинным в состоянии, непосредственно следующем за начальным;
- **Fp**: свойство p должно стать истинным хотя бы в одном состоянии в будущем или в текущем состоянии;
- **Gp**: свойство p должно быть истинным во всех будущих состояниях и в текущем состоянии;
- **pUq**: свойство q должно выполниться в некотором состоянии в будущем (или, возможно, в текущем состоянии), свойство p должно выполняться во всех состояниях до этого состояния (не включительно);
- **pRq**: свойство q должно быть истинным во всех состояниях, предшествующих состоянию, в котором станет истинным свойство p, и в самом этом состоянии; в этом случае говорят, что p «освобождает» q; если p ложно во всех состояниях, то оператор эквивалентен Gq.

Кроме того, существуют два префикса для описания ветвлений в дереве вычислений:

- **A** f : выражение f верно для всех ветвей вычисления;
- **E** f : выражение f верно хотя бы для одной ветви вычислений.

Выражения в CTL делятся на два типа: *выражения состояний* (ВС) и *выражения пути вычислений* (ВПВ). ВС верны для некоторого состояния; ВПВ верны для некоторого пути вычислений, начинающегося от данного состояния. При этом:

- Если p – атомарное свойство (элементарное высказывание), то p является ВС;
- Если f и g являются ВС, то выражения $\neg f, f \vee g, f \wedge g$ являются ВС;
- Если f является ВПВ, то **A** f и **E** f являются ВС;
- Если f является ВС, то f также является ВПВ;
- Если f и g являются ВПВ, то выражения $\neg f, f \vee g, f \wedge g$, $\mathrm{X} f$, $\mathrm{F} f$, $\mathrm{G} f$, $f \mathrm{U} g$, $f \mathrm{R} g$ являются ВПВ.

Для проверки конкретной модели ее требуется представить в виде машины состояний. Обычно вместе с логикой CTL используется *структура Крипке [5]*. Приведем несколько определений:

Определение 1. *Структурой Крипке называется тройка* $K(Q,R,L)$ *где*

- Q *– непустой набор состояний;*
- $R \subseteq Q \times Q$ *– бинарное отношение, показывающее, какие из состояний связаны друг с другом;*
- $L: Q \to 2^{AP}$ *– функция означивания состояний атомарными свойствами; свойство* p *верно в состоянии* s *тогда и только тогда, когда* $p \in L(s)$.

Определение 2. *Путь вычислений* π *в структуре Крипке* K *– это последовательность состояний* $\pi = q_0, q_1, \ldots$ *такая, что для любого* $i \geq 0$ *выполняется* $(q_i, q_{i+1}) \in R$. *Путь* π^i *- отрезок пути* π, *начинающийся с i-го состояния.*

Определение 3. *Запись* $K,q \models f$ *означает, что* f *выполняется в данной модели с данным начальным состоянием.* $K,q \models f$ *определяется индуктивно (будем считать, что* f_1, f_2 *-ВС, а* g_1, g_2 *- ВПВ):*

- $K,q \models p \Leftrightarrow p \in L(q)$, p – *атомарное свойство.*
- $K,q \models \neg f_1 \Leftrightarrow K,q \not\models f_1$,
- $K,q \models f_1 \vee f_2 \Leftrightarrow (K,q \models f_1) \vee (K,q \models f_2)$,
- $K,q \models f_1 \wedge f_2 \Leftrightarrow (K,q \models f_1) \wedge (K,q \models f_2)$,
- $K,q \models \mathrm{E}g_1$ $\Leftrightarrow$ *из q существует путь* π *такой, что* $K,\pi \models g_1$,
- $K,q \models \mathrm{A}g_1$ $\Leftrightarrow$ *для любого пути* π *из q верно* $K,\pi \models g_1$,
- $K,\pi \models f_1$ $\Leftrightarrow$ *q – первое состояние пути* π *и* $K,q \models f_1$,
- $K,\pi \models \neg g_1 \Leftrightarrow K,\pi \not\models g_1$,
- $K,\pi \models g_1 \vee g_2 \Leftrightarrow (K,\pi \models g_1) \vee (K,\pi \models g_2)$,
- $K,\pi \models g_1 \wedge g_2 \Leftrightarrow (K,\pi \models g_1) \wedge (K,\pi \models g_2)$,
- $K,\pi \models \mathbf{X}g_1 \Leftrightarrow K,\pi^1 \models g_1$,
- $K,\pi \models \mathbf{F}g_1 \Leftrightarrow \exists k \geq 0 : K,\pi^k \models g_1$,
- $K,\pi \models \mathbf{G}g_1 \Leftrightarrow \forall i \geq 0 : K,\pi^i \models g_1$,
- $K,\pi \models g_1 \mathbf{U} g_2 \Leftrightarrow \exists k \geq 0$, *такое что* $K,\pi^k \models g_2$ *и для всех* $0 \leq j < k$ $K,\pi^j \models g_2$,
- $K,\pi \models g_1 \mathbf{R} g_2 \Leftrightarrow \forall j \geq 0$, $i < j$: $K,\pi^i \not\models g_1 \Rightarrow K,\pi^j \models g_2$.

Корректные выражения в CTL могут быть следующими:

- атомарные свойства;
- выражения вида **A**(ВПВ);
- выражения вида **E**(ВПВ);
- стандартные булевы операторы с корректными CTL-выражениями в качестве операндов.

Под ВПВ здесь понимаются выражения, полученные применением темпоральных операторов **X,F,G,U,R** к корректным CTL-выражением. Легко заметить, что в CTL темпоральные операторы всегда используются вместе с одним из префиксов. Поэтому можно говорить о CTL-операторах: **AG, EG, AF, EF, AX, EX, AU, EU, AR, ER.**

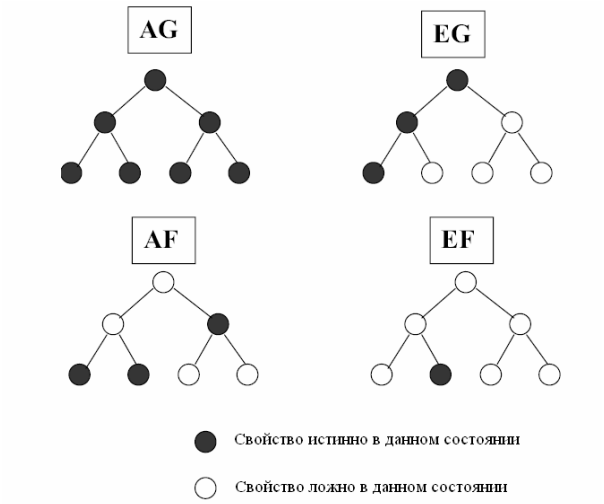

Рис 1. Иллюстрация действия основных CTL-операторов

Самыми часто используемыми из них являются операторы **AG, EG, AF, EF** (Рис 1)**.**

Приведем формальное определение задачи проверки модели при помощи CTL.

Определение 4. *Входными данными является структура Крипке* $K(Q,R,L)$, *описывающая некоторую систему с конечным числом состояний, и спецификация, представляющая собой CTL-выражение* f. *Определим набор состояний* Q^*, *удовлетворяющих* f $\{\forall q^* \in Q^* : K,q^* \models f\}$. *Система будет считаться прошедшей проверку, если все её начальные состояния содержатся в* Q^*.

Эта задача может быть решена с помощью пошагового алгоритма

- Поставим каждому состоянию $q \in Q$ в соответствие набор верных выражений $l(q)$. Начальное содержание набора – атомарные свойства из L(q).
- Обозначим через F_i набор всех подвыражений f со степенью вложенности операторов *i.* Таким образом, F_0 - набор всех атомарных свойств, составляющих *f*. Пусть *j* - уровень вложенности самого выражения *f*, то есть $F_j = \{f\}$.
- Для каждого *i*: $0 \leq i \leq j$ обойдем все состояния из *Q,* добавляя в $l(q)$ те выражения из F_i, которые являются истинными в данном состоянии. Поскольку перебор *i* идет в порядке возрастания вложенности, то значения простых выра-

жений для данного состояния уже известны к моменту вычисления составных выражений.

- После завершения работы алгоритма $f \in l(q) \Rightarrow K, q \models f$, таким образом, набор состояний Q^* легко определить.

X-машины

Определение X-машины расширяет определение конечного автомата, добавляя память [6;2].

Определение 5. *X-машиной называется девятка* $M = (Q, \Sigma, \Gamma, M, \Phi, F, q_0, M_0, T)$*:*

- Q *– конечный набор состояний*;
- Σ, Γ *– входной и выходной алфавиты соответственно*;
- M *– память, конечный набор переменных*;
- Φ *– конечный набор функций* ϕ*, отображающих входной символ и содержание памяти в выходной символ и новое содержание памяти* $\phi: \Sigma \times M \to \Gamma \times M$;
- F *– функция, отображающая состояние и функцию из* Φ *в новое состояние* $f: Q \times \Phi \to Q$;
- q_0, M_0 *– начальное состояние и начальное содержимое памяти соответственно*;
- T *– множество заключительных состояний.*

Работа X-машины заключается в выполнении некоторого количества шагов (тактов), на каждом из которых происходит чтение символа из входной цепочки. На основе текущего состояния X-машины выбирается функция из Ф, которая генерирует выходной символ и, возможно, неким образом модифицирует память. Переходы между состояниями задаются функцией F.

Для того чтобы верификация X-машины при помощи логики X-CTL была возможна, требуется установить дополнительное ограничение на значения памяти: множество всех возможных значений каждой переменной в памяти должно быть конечно. Учитывая эти ограничения, введем следующие обозначения:

- $M(0)$, $M(1)$, ... - переменные в памяти;
- M^q – множество всех возможных значений памяти в состоянии q; такое множество конечно, и его можно представить в виде $M^q = \{m^0, ..., m^i, ..., m^k\}$, $0 \le i \le k$, где $m^i = (M^i(0), M^i(1), ...)$ – i-е возможное состояние памяти;
- запись qm_j^i означает, что X-машина находится в состоянии $q_j \in Q$ и при этом память имеет значение $m^i \in M^{q_j}$, состояния $q_j \in Q$, таким образом, расщепляются на *метасостояния* qm_j^i.

Логика X-CTL

Логика CTL имеет ограниченную применимость при верификации X-машин, так как ее операторы не учитывают содержимое памяти. Для того чтобы верификация стала возможной, в данной работе предлагается д ополнить логику CTL.Назовем новую расширенную логику X-CTL.

Обозначим через *MProp* множество всех возможных предикатов, состоящих из переменных памяти и атомарных выражений, например, $M(1) = 0$ или $M(3) \ne 2$.

В логике X-CTL введем два новых оператора:

- $M_x(p \in MProp)$ – выражение p верно при любом состоянии памяти;
- $m_x(p \in MProp)$ – существует состояние памяти, в котором выражение p верно.

Введем понятие выражения состояния (ВС), выражения пути вычислений (ВПВ) и выражения в памяти (ВП). Эти типы выражений определяются рекурсивно.

- Если $p \in MProp$, то p является *выражением в памяти* (ВП).
- Если a и b являются ВП, то $\neg a$, $a \vee b$ и $a \wedge b$ также являются ВП.
- Если a является ВП, то $M_x a$ и $m_x a$ являются ВС.
- Если f и g являются ВС, то выражения $\neg f$, $f \vee g$, $f \wedge g$ являются ВС.
- Если f и g являются ВС, то выражения $\mathrm{X} f$, $\mathrm{F} f$, $\mathrm{G} f$, $f \,\mathrm{U}\, g$, $f \,\mathrm{R}\, g$ являются ВПВ.
- Если f является ВПВ, то $\mathbf{A} f$ и $\mathbf{E} f$ являются ВС.

Корректные выражения в X-CTL являются ВС. Для того чтобы дать формальное определение верификации Х-машин при помощи X-CTL, понадобятся определения означивающей функции, пути вычислений и отношения $M, q \models f$.

Определение 6. *$L_M : QM \to 2^{MProp}$ – функция означивания метасостояний предикатами из MProp. $p \in MProp$ верно в метасостоянии qm_j^i тогда и только тогда, когда $p \in L_M(qm_j^i)$. Так же определим $L_M(q_j) = \bigcup_{i=0}^{k} L_M(qm_j^i)$.*

Определение 7. *Путь вычислений π в Х-машине М – это последовательность состояний $\pi = q_0, q_1, \ldots$ такая, что для любого $i \geq 0$, $\exists \phi \in \Phi : (q_i, \phi, q_{i+1}) \in F$. Будем обозначать π^i – отрезок пути π, начинающийся с i-го состояния. Будем обозначать first(π) – первое состояние в пути π.*

Определение 8. *Запись $M, q \models f$ означает, что выражение X-CTL f выполняется в состоянии q в Х-машине М. В случае, если p является ВП, $M, qm^i \models p$ означает, что p выполняется в метасостоянии qm^i. Если g является ВПВ, то запись $M, \pi \models g$ означает, что g выполняется вдоль некоторого пути π в Х-машине М. Для прочих случаев отношение $\models$ определяется по индукции (будем считать, что a, b – ВП, f, f_1, f_2 – ВС, а g_1, g_2 – ВПВ)*:

- $M, qm_j^i \models p \Leftrightarrow p \in L_M(qm_j^i)$,

где $p \in MProp$;

- $M, qm_j^i \models \neg a \Leftrightarrow M, qm_j^i \not\models a$;
- $M, qm_j^i \models a \wedge b \Leftrightarrow M, qm_j^i \models a$

и $M, qm_j^i \models b$;

- $M, qm_j^i \models a \vee b \Leftrightarrow M, qm_j^i \models a$

или $M, qm_j^i \models b$;

- $M, q_j \models M_x a \Leftrightarrow \forall\, i \geq 0 : M, qm_j^i \models a$;
- $M, q_j \models m_x a \Leftrightarrow \exists\, i \geq 0 : M, qm_j^i \models a$;
- $M, q \models \neg f \Leftrightarrow M, q \not\models f$;
- $M, q \models f_1 \wedge f_2 \Leftrightarrow M, q \models f_1$ и $M, q \models f_2$;
- $M, q \models f_1 \vee f_2 \Leftrightarrow M, q \models f_1$

или $M, q \models f_2$;

- $M, \pi \models \mathbf{X}f \Leftrightarrow M, first(\pi^1) \models f$;
- $M, \pi \models \mathbf{F}f \Leftrightarrow \exists i \geq 0 : M, first(\pi^i) \models f$;
- $M, \pi \models \mathbf{G}f \Leftrightarrow \forall i \geq 0 : M, first(\pi^i) \models f$;
- $M, \pi \models f_1 \mathbf{U} f_2 \Leftrightarrow \exists k \geq 0 : M, first(\pi^k) \models f_2$, $0 < j < k$, $M, first(\pi^j) \models f_1$;
- $M, \pi \models f_1 \mathbf{R} f_2 \Leftrightarrow$ для $\forall j \geq 0$, если для $\forall i \leq j$ $M, first(\pi^i) \not\models f_1$, то $M, first(\pi^j) \models f_2$;
- $M, q \models \mathbf{A}g \Leftrightarrow M, \pi \models g$ для всех таких путей π, что $first(\pi) = q$;
- $M, q \models \mathbf{E}g \Leftrightarrow$ существует такой путь π, что $first(\pi) = q$ и $M, \pi \models g$.

Теперь можно формализовать задачу верификации Х-машины при помощи CTL:

Определение 9. *Пусть дана Х-машина $M = (Q, \Sigma, \Gamma, M, \Phi, F, q_0, M_0, T)$ и X-CTL выражение f. Тогда проверка модели осуществляется путем определения множества состояний, удовлетворяющих f ($\{q^* \in Q^* \mid M, q^* \models f\}$). Х-машина считается прошедшей проверку, если ее начальное состояние входит во множество Q^*.*

Пример

Рассмотрим простейший пример – Х-машину, моделирующую счетчик [7]. Счетчик получает на вход количество тактов n, которое нужно пропустить, и синхросигналы TICK, обозначающие завершение каждого такта. По прошествии нужного количества тактов в выходную строку записывается символ EVENT. Ниже будет дано описание каждого элемента Х-машины в соответствии с её определением. При этом функции из F удобно изображать графиком переходов между состояниями (Рис. 2).

Набор состояний: $Q = \{Q_1, Q_2\}$.

Входной и выходной алфавиты: $\Sigma = \{\varepsilon, n \leq N_{\max}, SYNC\}, \Gamma = \{\varepsilon, EVENT\}$.

Переменные в памяти:
$M = (M(0), M(1), M(2), M(3))$.

$M(0) = \{ОЖИДАНИЕ, ОТСЧЕТ\}$ состояние счетчика.

$M(1) = N_{\max}$ – максимальное число тиков для счетчика.

$M(2) = n_1$ – требуемое число тиков при отсчете.

$M(3) = n_2$ – прошедшее с начала отсчета число тиков.

Начальное состояние и начальное состояние памяти: $q_0 = Q_1, m_0 = (ОЖИДАНИЕ, N_{\max}, 0, 0)$.

Функции из Ф:

$заданий\_нет(SYNC; M) = (\varepsilon; M)$,

$новое\_задание(n; M(1) \geq n) = (\varepsilon; M(0) =$

$= ОТСЧЕТ, M(2) = n)$,

$продолжить(SYNC, M(2) \neq M(3)) =$

$= (\varepsilon; M'(3) = M(3) + 1)$,

$отсчет\_закончен(SYNC, M(2) = M(3)) =$

$= (EVENT; M(0) = ОЖИДАНИЕ; M(3) = 0)$.

Заключительные состояния: $T = \{Q_1\}$.

Проведем проверку такой модели счетчика с помощью X-CTL выражений. Приведем некоторые из них:

- В любом состоянии $n_2 \leq n_1$ и $n_1, n_2 \leq N_{\max}$

$AG[M_x(M(2) \leq M(1) \wedge M(3) \leq M(1) \wedge M(3)$

$\leq M(2))]$;

- Возможен переход в состояние ОТСЧЕТ:

$E(M_x(M(0) = ОЖИДАНИЕ)UM_x(M(0) =$

$= ОТСЧЕТ))$;

- Если в состоянии ОТСЧЕТ достигнуто нужное количество тактов, то следующим состоянием будет ОЖИДАНИЕ:

$EF(m_x(M(0) = ОТСЧЕТ \wedge M(2) =$

$= M(3)) \wedge AXM_x(M(0) = ОЖИДАНИЕ))$.

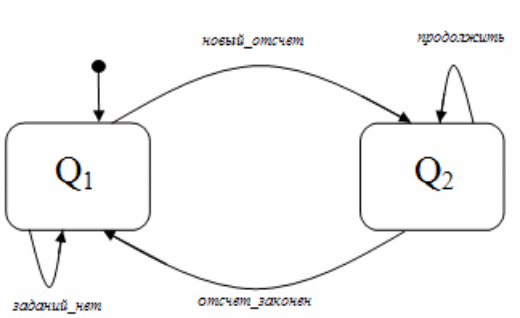

Рис. 2. График перехода между состояниями

Заключение

Хотя логика CTL и представляет собой мощный инструмент для формальной верификации моделей, существовавшие версии логики CTL не могли быть использованы для верификации X-машин. В работе предложена и описана новая логика X-CTL, расширяющая логику CTL так, что это позволяет определять критерии верификации для X-машин.

Литература

1. Кларк Э. М., Грамберг О., Пелед Д. Верификация моделей программ: Model Checking. – М. МЦНМО, – 2002. – 416 с.
2.Соболев М. С. Описание систем при помощи X-машин //Информационные технологии и вычислительные системы, 2009, №4, с. 22-27
3. C.A. R. Hoare. An axiomatic basis for computer programming //Communications of the ACM, 1969
4. Emerson, E.; Clarke, E Characterizing correctness properties of parallel programs using fixpoints //Automata, Languages and Programming, 1986
5. Clarke, E. M.; Emerson, E; Sistla, A. Automatic verification of finite-state concurrent systems using temporal logic specifications //ACM Transactions on Programming Languages and Systems, 1986
6. Eilenberg, S. Automata, Languages and Machines, Vol. A //Academic Press, 1974
7. Соболев М.С. Автоматные модели вычислений //Моделирование и обработка информации. – М.: МФТИ, 2009. – С 76-85.

Соболев Михаил Сергеевич. Аспирант Московского физико-технического института. Окончил магистратуру МФТИ в 2007 году. Имеет 4 публикации. Область научных интересов: компьютерная лингвистика, теория автоматов. E-mail: sabelogic@gmail.com.