22. Метод Model checking. Модель Крипке. Линейная темпоральная логика: синтаксис, семантика, примеры формул. Автомат Бюхи. Верификация LTL при помощи автоматов Бюхи.

http://is.ifmo.ru/verification/velder_verification_posobie.pdf

Модель Крипке(стр. 30).

Атомарные предложения – это базовые предложения, которые могут быть сделаны. Множество атомарных предложений обозначается AP. Примерами атомарных предложений являются предложения «x больше 0» или «x равно 1» для некоторой переменной x. Другими примерами таких предложений являются «идет дождь» или «в магазине нет покупателей». В принципе, атомарные предложения определяются над множеством переменных $x, y, \ldots$, констант $0, 1, 2, \ldots$, функций $max, gcd, \ldots$ и предикатов $x = 2$, $x \ mod \ 2 = 0, \ldots$; допускаются предложения вида $max(x, y) \leq 3$ или $x = y$.

Моделью Крипке (структурой Крипке) над множеством атомарных предложений AP называется тройка $(S, R, Label)$, где

- S – непустое множество состояний;
- $R \subset S \times S$ – тотальное отношение переходов на S, которое сопоставляет элементу $s \in S$ его возможных потомков;
- $Label : S \to 2^{AP}$ сопоставляет каждому состоянию $s \in S$ атомарные предложения $Label(s)$, которые верны в s.

Отношение $R \subset S \times S$ называется тотальным, если оно ставит в соответствие каждому состоянию $s \in S$ как минимум одного потомка $(\forall s \in S : \exists s' \in S : (s, s') \in R)$. Иногда еще требуют, чтобы для модели Крипке был задан набор начальных состояний $S_0 \subset S$. Путь в модели Крипке из состояния s_0 – это бесконечная последовательность состояний $\pi = s_0 s_1 s_2 \ldots$ такая, что для всех $i \geq 0$ выполняется $R(s_i, s_{i+1})$.

Синтаксис LTL(стр. 35).

Пусть AP – множество атомарных предложений. Тогда:

1. p является формулой для всех $p \in AP$.
2. Если ϕ – формула, то $\neg\phi$ – формула.
3. Если ϕ и ψ – формулы, то $\phi \vee \psi$ – формула.

4. Если ϕ – формула, то $X\ \phi$ – формула.

5. Если ϕ и ψ – формулы, то $\phi\ U\ \psi$ – формула.

Множество формул, построенных в соответствии с этими правилами, называется формулами LTL.

Заметим, что множество формул, полученных на основе первых трех пунктов, определяет множество всех формул пропозициональной логики. Пропозициональная логика является, таким образом, собственным подмножеством LTL. Темпоральными операторами являются только X (neXt) и U (Until).

Еще формулы:

- $\phi \wedge \psi = \neg(\neg\phi \vee \neg\psi)$;
- $\phi \rightarrow \psi = \neg\phi \vee \psi$;
- $\phi \leftrightarrow \psi = (\phi \rightarrow \psi) \wedge (\psi \rightarrow \psi)$;
- $true = \phi \vee \neg\phi$;
- $false = \neg true$;
- $F\ \phi = true\ U\ \phi$;
- $G\ \phi = \neg F\ \neg\phi$.

F – Future, G – Globally.

Итак, можно утверждать, что формула без темпорального оператора (X, F, G, U) на «верхнем уровне» относится к текущему состоянию, формула $X\ \phi$ – к следующему состоянию, $G\ \phi$ – ко всем будущим состояниям, $F\ \phi$ – к некоторому будущему состоянию, а U – ко всем будущим состояниям до тех пор, пока определенное условие не станет верным.

Семантика LTL(стр. 37).

LTL-модель – это тройка $M = (S, R, Label)$, в которой:

- S – непустое конечное множество состояний;
- $R : S \rightarrow S$ сопоставляет элементу $s \in S$ единственный следующий за ним элемент $R(S)$;
- $Label : S \rightarrow 2^{AP}$ сопоставляет каждому состоянию $s \in S$ атомарные предложения $Label(s)$, которые верны в s.

Для состояния $s \in S$ состояние $R(S)$ – единственное состояние, следующее за s. Важной характеристикой функции R является то, что она работает как генератор бесконечных последовательностей $s, R(S), R(R(S)), R(R(R(S))), \ldots$ Последовательности состояний для семантики LTL являются краеугольным камнем. Можно с тем же успехом определить LTL-модель как структуру $(S, \sigma, Label)$, где σ – бесконечная последовательность состояний, а S и $Label$ определены, как это сделано выше.

Функция $Label$ указывает, какие атомарные предложения верны для заданного состояния M. Если для состояния s имеем $Label(s) = \varnothing$, то это означает, что ни одно атомарное предложение не верно в состоянии s. Состояние s, в котором предложение p верно ($p \in Label(s)$), иногда называется p-состоянием.

Смысл формул в логике определяется в терминах отношения выполняемости (обозначаемого $\vDash$) между моделью M, одним из ее состояний s и формулой ϕ.

$(M, s, \phi) \in \vDash$ обозначается в инфиксной нотации: $M, s \vDash \phi$. Идея заключается в том, что $M, s \vDash \phi$ тогда и только тогда, когда ϕ верно в состоянии s модели M. Когда модель M ясна из контекста, будем опускать модель и писать $s \vDash \phi$ вместо $M, s \vDash \phi$. Семантика LTL определяется следующим образом. Пусть $p \in AP$ – атомарное предложение, $M = (S, R, Label)$ – LTL-модель, $s \in S$ и ϕ, ψ – LTL-формулы. Отношение выполняемости $\vDash$ задается таким способом:

$$
\begin{aligned}
&s \vDash p && \Leftrightarrow p \in Label(s);\\
&s \vDash \neg\phi && \Leftrightarrow \neg(s \vDash \phi);\\
&s \vDash (\phi \vee \psi) && \Leftrightarrow (s \vDash \phi) \vee (s \vDash \psi);\\
&s \vDash X\phi && \Leftrightarrow R(s) \vDash \psi;\\
&s \vDash (\phi U \psi) && \Leftrightarrow \exists j \geq 0 : R^j(s) \vDash \psi \wedge (\forall 0 \leq k < j : R^k(s) \vDash \phi).
\end{aligned}
$$

Здесь $R^0(s) = s$ и $R^{n+1}(s) = R(R^n(s))$ для любого $n \geq 0$. Если $R(s) = s'$, то состояние s' называется прямым потомком s. Если $R^n(s) = s'$ для $n \geq 1$, то состояние s' называется потомком s. Если $M, s \vDash \phi$, то говорят, что модель M удовлетворяет формуле ϕ в состоянии s. Иначе говоря, формула ϕ выполняется в состоянии s модели M.

Примеры на стр. 40-41.

Автомат Бюхи(стр. 62).

Пусть AP – множество атомарных предложений. Автоматом Бюхи над алфавитом 2^{AP} называется четверка $A = (Q, q_0, \delta, F)$, где

- Q – конечное множество состояний;

- q_0 – начальное состояние;
- $\delta \subset Q \times 2^{AP} \times Q$ – тотальное отношение переходов;
- $F \subset Q$ – множество допускающих состояний.

Опишем алгоритм Герта, Пеледа, Варди и Волпера для построения автомата Бюхи по LTL-формуле. Введем новый темпоральный оператор R (Release), который определяется следующим образом:

$$\phi R \psi = \neg(\neg\phi \ U \ \neg\psi).$$

Для него верно, например, аналогичное тождество расширения:

$$\phi R \psi \equiv \psi \wedge (\phi \vee X(\phi \ R \ \psi)).$$

Для работы алгоритма требуется, чтобы LTL-формула была приведена в негативную нормальную форму – отрицание должно применяться только к атомарным предложениям. Опишем метод приведения LTL-формулы к негативной нормальной форме.

1. Заменим все подформулы вида $F\phi$ на $true \ U \ \phi$.
2. Заменим все подформулы вида $G\phi$ на $false \ R \ \phi$.
3. Используя булевы тождества, оставим в формуле только три логические операции: $\neg, \vee, \wedge$.
4. Используя тождества LTL
   - $\neg(\phi \ U \ \psi) \equiv \neg\phi \ R \ \neg\psi$;
   - $\neg(\phi \ R \ \psi) \equiv \neg\phi \ U \ \neg\psi$;
   - $\neg X \ \phi \equiv X \ \neg\phi$,

   погружаем отрицания внутрь темпоральных операторов

Для алгоритма потребуются следующие структуры данных:

- UID – уникальный идентификатор;
- Formula – LTL-формула;
- Node – вершина графа переходов автомата Бюхи.

Для алгоритма неважно, в каком виде будут представлены уникальные идентификаторы и формулы, поэтому опишем только структуру Node.

```
struct Node
{
    UID id;
    list<NodeID> incoming;
    list<Formula> old;
    list<Formula> new;
    list<Formula> next;
};
```

Здесь *incoming* – список вершин-предшественников (вершин, из которых идет дуга в текущую вершину). В полях *old*, *new* и *next* содержатся списки подформул исходной формулы.

Функция *CreateAutomaton* строит граф переходов автомата Бюхи по формуле f.

```
list<Node> CreateAutomaton (Formula f)
{
    Node n;
    n.incoming = {init};
    n.old = ∅;
    n.new = {f};
    n.next = ∅;
    return Expand(n, ∅);
}
```

Добавление новой вершины выполняется функцией *Expand*. Код на стр. 63-65.

Верификация LTL при помощи автоматов Бюхи(стр. 67).

Пусть даны модель Крипке и LTL-формула, выполнение которой на модели требуется проверить. Общая идея алгоритма следующая:

- Из отрицания LTL-формулы строится эквивалентный ей автомат Бюхи.
- Модель Крипке также преобразуется в автомат Бюхи.
- Строится третий автомат Бюхи как пересечение первых двух. Такой автомат будет допускать пути исходной модели, которые не удовлетворяют LTL-формуле спецификации.
- Если язык, допускаемый построенным автоматом-пересечением, пуст, то верификация успешна. Если нет, то путь, допускаемый автоматом-пересечением, является контрпримером.