# Символьная верификация моделей

Ю. Лифшиц*

7 января 2006 г.

## План лекции

1. Двоичные разрешающие диаграммы
2. Вычисление неподвижной точки
3. Символьный алгоритм верификации CTL

## 1 Двоичные разрешающие диаграммы

### 1.1 Определения и свойства

Рассмотрим булеву функцию вида

$$F : \{0,1\}^k \to \{0,1\}.$$

(Заметим, что таких функций — $2^{2^k}$.) Рассмотрим ориентированное корневое дерево, такое что:

1. исходящая степень любой внутренней вершины равна двум;
2. каждая внутренняя вершина помечена како-либо переменной;
3. одно исходящее ребро любой внутренней вершины помечено 0, другое — 1;
4. на листьях записаны значения функции.

Такое дерево является одним из способов задания булевой функции $F$ и называется **двоичным разрешающим деревом**. Внутренние вершины такого дерева называются **нетерминалами**, а листья — **терминалами** (см. рис. 1).

Легко убедиться в том, что любой булевой функции соответсвует какое-то ДРД и по любому ДРД можно построить функцию. Действительно, если

*Законспектировал С. Вишняков.

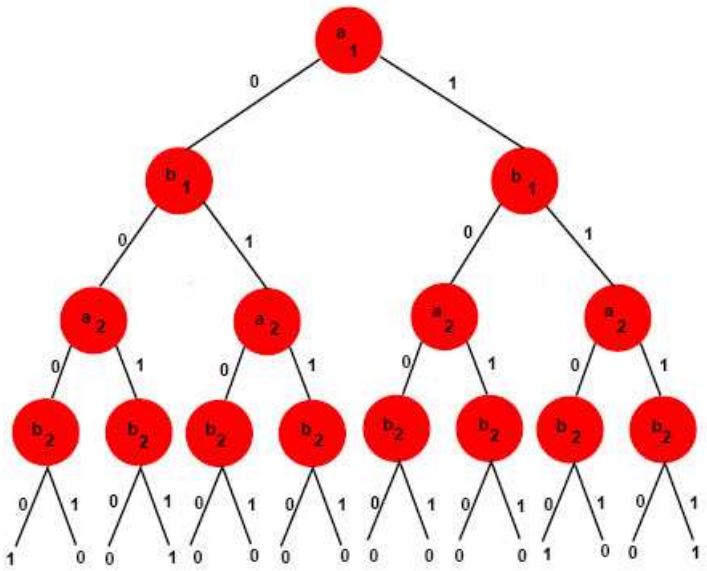


Рис. 1: Пример ДРД. Функция $(a1 = b1) \wedge (a2 = b2)$

задать какой-нибудь порядок переменных, то любая последовательность нулей и единиц при спуске по дереву (вариантов спуска $2^k$) будет взаимно однозначно соотвествать строке из таблицы истинности функции.

Теперь рассмотрим ациклический ориентированный граф, удовлетворяющий тем же условиям, что и ДРД. Такой граф называется **двоичной разрешающей диаграммой**. Если переменные упорядочены каким-либо образом и на любом пути из корня к листьям переменные встречаются именно в таком порядке (не обязательно все), то такая диаграмма называется **упорядоченной двоичной разрешающей диаграммой**.

Далее будем использовать аббревиатуру OBDD (Ordered Bynary Decision Diagram).

Для каждого порядка переменных существует минимальная OBDD. Для разных порядков размеры OBDD могут отличаться. К примеру, для функции $(a_1 \oplus b_1)\& \ldots \&(a_n \oplus b_n)$ при разных порядках размер меняется от $3n+2$ до $3 \cdot 2^n - 1$. (Порядки $a_1, b_1, \cdots, a_n, b_n$ и $a_1, \cdots, a_n, b_1, \cdots, b_n$, соответсвенно). Есть функции, при любом порядке дающие OBDD экспоненциального размера. Найти оптимальный порядок — NP-трудная задача. Существуют эвристические алгоритмы, подбирающие по формуле порядок близкий к оптимальному (идея, примерно, следующая: переменные, "взаимодействующие" между собой в формуле, ставить близко в OBDD).

**Вопрос:** OBDD, по сути, — детерминированный конечный автомат (стартовая вершина — самая верхняя, входное слово — набор значений переменных, допускающие состояния — терминалы со значением 1). Можно

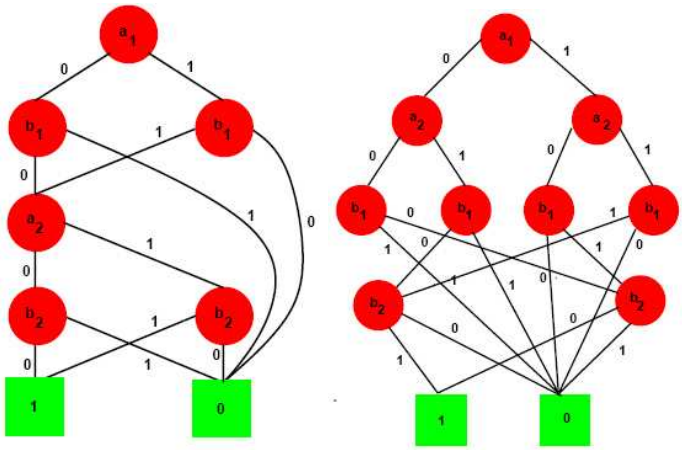


Рис. 2: Два варианта OBDD для той же функции. Стрелки можно не рисовать — порядок задается высотой узлов.

ли добиться минимализации OBDD, применив алгоритм минимализации ДКА?

**Ответ:** OBDD можно отождествить с ДКА только зафиксировав предварительно порядок переменных. Применив алгоритм для ДКА, мы не найдем оптимальный порядок, а всего лишь получим минимальную OBDD только для определенного порядка переменных.

## 1.2 Операции над OBDD

Пусть есть OBDD для функций $f$ и $f'$. Построим OBDD для

1. $\neg f$;
2. $f \vee f'$;
3. $f \wedge f'$;
4. любой бинарной булевой функции от $f$ и $f'$.

OBDD для функции $\neg f$ строится очень просто: заменить значения во всех терминальных вершинах на их отрицание.

Для построения OBDD бинарной функции от двух булевых функций используем **алгоритм Бриана**.

**Идея алгоритма.**

Пусть даны функции $f$ и $f'$ и бинарная операция $*$. Для каждой вершины $a$ OBDD для $f$ рассмотрим OBDD "висящую" на ней. Эта "висящая" на $a$ OBDD задает функцию $f_a$. Для каждой пары $(a; a')$ будем строить диаграмму для $f_a * f'_{a'}$, где $a$ — вершина OBDD для $f$, а $a'$ — вершина OBDD

для $f'$. То есть придется подсчитать $|OBDD(f)| * |OBDD(f')|$ диаграмм: сначала диграммы для нижних вершин, потом используя уже посчитанные диаграммы — для более высоких, пока не доберемся до $f_s * f'_{s'}$, где $s$ и $s'$ — корни диаграмм для $f$ и $f'$, соответсвенно.

**Формулы Шеннона.**

Эти формулы вдохновили Бриана на создание своего алгоритма.

$$f * f' = (x \wedge (f|_{x\to 1} * f')) \vee (\neg x \wedge (f|_{x\to 0} * f'))$$

$$f * f' = (x \wedge (f|_{x\to 1} * f'|_{x\to 1})) \vee (\neg x \wedge (f|_{x\to 0} * f'|_{x\to 0}))$$

**Алгоритм Бриана**

**База.** Вычислить OBDD для пар: терминал из $f$-OBDD, терминал из $f'$-OBDD. Таким образом, после этого шага у нас есть все комбинации вида $f_t * f'_{t'}$, где $t$ и $t'$ — терминальные вершины диаграмм для $f$ и $f'$ соответсвенно.

**Переход.** Пусть посчитаны все OBDD функций $f_a * f'_{a'}$ для всех комбинаций $a$ и $a'$, высота которых (считая от терминальных вершин) меньше или равна $h$ и $h'$ соответсвенно. Заметим, что эта высота определяется порядком, заданным на вершинах. Будем считать, что способ определения высоты мы зафиксировали. Теперь можно посчитать OBDD функций $f_a * f'_{a'}$, где высота $a$ равна $h+1$, а высота $a'$ — меньше или равна $h'$ (или наоборот).

**Случай 1.** Вершины $a$ и $a'$ помечены одинаковой переменной $x$. Пусть $a_0$, $a_1$, $a'_0$, $a'_1$ — их дети (индекс соответвует значению, которым помечено ребро). Тогда рисуем вершину, помеченную $x$, слева (переход по 0) "подвешиваем" OBDD для $f_{a_0} * f'_{a'_0}$, справа (переход по 1) — для $f_{a_1} * f'_{a'_1}$. Эти OBDD посчитаны по индукционному предположению.

**Случай 2.** Вершины $a$ и $a'$ помечены разными переменными $x$ и $y$ (соответсвенно). Тогда OBDD будет выглядеть следующим образом: рисуем вершину $(x)$, слева — OBDD для $f_{a_0} * f'_{a'}$, справа — для $f_{a_1} * f'_{a'}$. Эти OBDD также посчитаны по индукционному предположению.

Каждую вновь полученную OBDD упрощаем.

На рисунках проиллюстрирован пример работы алгоритма Бриана.

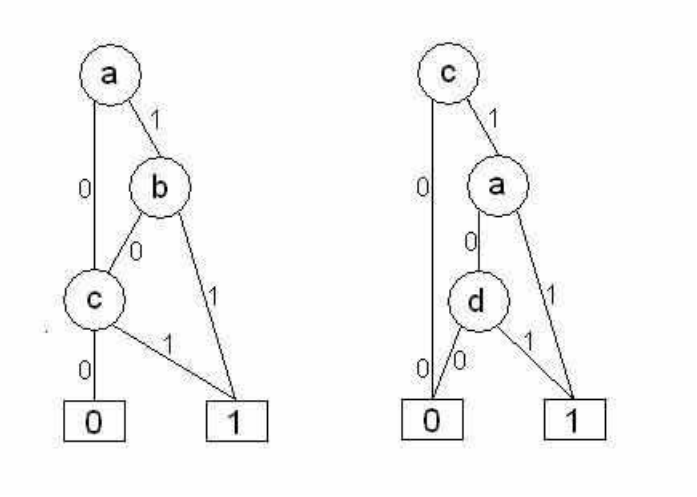


Рис. 3: Рассмотрим функции $f = (a \wedge b) \vee c$ (слева) и $f' = (a \vee d) \wedge c$ (справа). Продемонстрируем начало постороения $f \wedge f'$

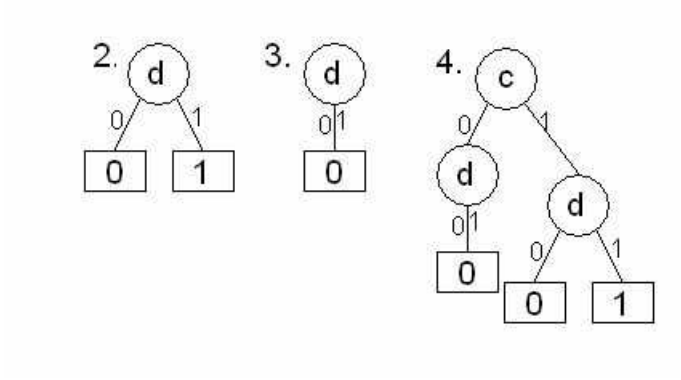


Рис. 4: Первый шаг алгоритма (обработка терминальных вершин) очевиден. следующие шаги: 2. $a = 0$, $a' = d$; 3. $a = 1$, $a' = d$; 4. $a = c$, $a' = d$. Легко проследить, как на шаге 4 мы использовали результаты, полученные на шагах 2 и 3. Для лучшего понимания алгоритма рекомендуется проделать пару следующих шагов самостоятельно.

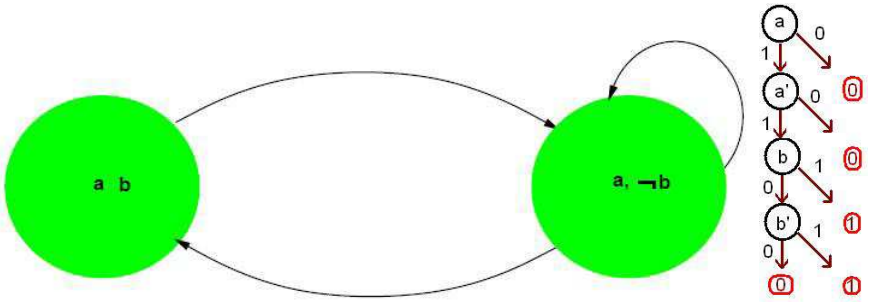


Рис. 5: Пример описания модели Крипке при помощи OBDD: $S = (a \wedge b') \vee (a \wedge \neg b)$; $R = (a \wedge b \wedge a' \wedge \neg b') \vee (a \wedge \neg b \wedge a' \wedge b') \vee (a \wedge \neg b \wedge a' \wedge \neg b')$. Справа — диаграмма для $R$. В качестве упражнения читателю предлагается самостоятельно построить диаграмму для $S$.

### 1.3 OBDD и модель Крипке

Напомним, что **модель Крипке** — один из способов описания программ. Пусть $AP$ — множество атомарных высказываний. Моделью Крипке над $AP$ называется четверка $M = (S, S_0, R, L)$, где:

1. $S$ — конечное множество состояний;
2. $S_0 \subseteq S$ — множество начальных состояний;
3. $R \subseteq S \times S$ — функция переходов;
4. $L : S \to 2^{AP}$ — функция истинности (задает, какие атомарные высказывания верны в каждом конкретном состоянии).

Пусть $|AP| = n$, тогда можно считать, что $S \subseteq \{0,1\}^n$.
Для описания модели Крипке зададим:

1. характеристическую функцию $f_S$ для множества $S$;
2. характеристическую функцию $f_R$ для отношения $R(x_1, \ldots, x_n, x'_1, \ldots, x'_n)$.

Смотрите пример на рисунке 5.

## 2 Вычисление неподвижной точки

### 2.1 Опеределения

Рассмотрим отображение $\tau : 2^U \to 2^U$. Множество $S \subseteq U$ называется **неподвижной точкой** относительно $\tau$, если $\tau(S) = S$.
Множество $S$ — минимальная неподвижная точка, если:

1) $S$ — неподвижная точка;

2) Любая неподвижная точка содержит $S$.

Аналогично определяется максимальная неподвижная точка. Минимальная и маскимальная неподвижные точки обозначаются как $\mu S\,.\,\tau(S)$ и $\nu S\,.\,\tau(S)$.

Пусть $U$ конечно. $\tau$ называется монотонным, если $X \subset Y \Rightarrow \tau(X) \subset \tau(Y)$.

### 2.2 Теорема

**[Тарский:]**

Если отображение $\tau$ монотонно, то существуют минимальная и максимальная неподвижные точки.

**Доказательство.** Предъявим минимальную и максимальную неподвижные точки. Докажем следующие факты:

1) $\mu S\,.\,\tau(S) = \cup_{i=1}^{\infty}\tau^i(\varnothing)$

2) $\nu S\,.\,\tau(S) = \cap_{i=1}^{\infty}\tau^i(U)$

Во-первых, для любого $i$ верно $\tau^i(\varnothing) \subseteq \tau^{i+1}(\varnothing)$ (база: $\varnothing \subseteq \tau(\varnothing)$, индукционный переход — по монотонности $\tau$).

Пусть $S = \cup_{i=1}^{\infty}\tau^i(\varnothing)$. Покажем, что $S \subseteq \tau(S)$ и $\tau(S) \subseteq S$. Действительно, поскольку $U$ — конечное множество, $\tau^i(\varnothing) = \tau^{i+1}(\varnothing)$, начиная с какого-то $m$. Таким образом, $S = \tau^m(\varnothing)$. Но тогда $\tau(S) = \tau^{m+1}(\varnothing) = \tau^m(\varnothing) = S$.

Покажем, что $S$ содержится в любой неподвижной точке $S'$. Понятно, что $\varnothing \subseteq S'$. Применим $\tau$ к обеим частям, учитывая, что $S'$ — неподвижная точка. Получим $\tau(\varnothing) \subseteq S'$. Применим $\tau$ $m$ раз, получим $\tau^m(\varnothing) \subseteq S'$. А это и означает, что $S \subseteq S'$. Аналогично можно провести доказательсво для максимальной неподвижной точки.

## 3 Символьный алгоритм верификации CTL

### 3.1 Постановка задачи

Напомним, что логика CTL строится из конструкций вида:

1. $\neg f$ , $f \vee g$;
2. $\mathbf{EX}f$ (из данной вершины существует ребро в другую вершину, в которой выполнено свойство $f$);
3. $\mathbf{EG}f$ (существует путь из данной вершины, на котором в каждой вершине выполнено $f$);
4. $\mathbf{E}[f\mathbf{U}g]$ (существует путь из данной вершины, на котором до какого-то момента всегда выполняется $f$, а потом — всегда $g$).

Задача верификации CTL формулируется следующим образом: даны модель Крипке M = (S, R, L) и CTL-формула $f$; необходимо найти множество $\{s \in S \mid M, s \models f\}$ (множество вершин $s$, для которых выполняется $p$).

## 3.2 Алгоритм

Задаем модель Крипке в виде OBDD функций $f_S$ и $f_R$. Для каждой подформулы формулы $f$ будем строить OBDD состояний, ее выполняющих.

При этом начинаем с атомарных высказываний. Для них делаем подстановку в $f_S$. Например, для атомарного высказывания $A$ в OBDD для $f_s$ обрезаем все поддеревья, где $A = 0$, и упрощаем OBDD.

Для операций $\vee$ и $\neg$ используем алгоритм Бриана. Осталось разобраться с операциями $\mathbf{EX}f$, $\mathbf{EG}f$ и $\mathbf{E}[f\mathbf{U}g]$.

Множество состояний $\mathbf{EX}f$ определяется следующей формулой:

$$\exists \bar{v}'[f(v') \wedge R(v, v')].$$

OBDD для $f(v')$ у нас есть по идукцинному предположению. Используя алгоритм Бриана, получаем диаграмму для $f(v') \wedge R(v, v')$. Из этой диаграммы несложно получить требуемую. Действительно, чтобы получить диаграмму $\exists v' \ldots$ достаточно в старой диаграмме найти вершины, помеченные переменной $v'$, и их ее поддеревьям применить операцию $\vee$, используя алгоритм Бриана (в этих поддеревьях некоторые вершины могут совпадать — такие вершины надо предварительно продублировать).

Теперь построим OBDD для $EGf$. Рассмотрим оператор $\tau(Z) = (f \wedge \mathbf{EX})(Z)$. Этот оператор действует на подмножествах $S$ следующим образом: по множеству состояний $Z$ он возвращает состояния, где

1. выполнена формула $f$;
2. есть исходящее ребро в множество $Z$.

**Лемма 1.** Отображение $\tau$ монотонно.
Если $Z \subseteq Z'$, то множество вершин, из которых есть ребро в $Z$, содержится в множестве вершин, из которых есть ребро в $Z'$. То же справедливо для пересечений этих множеств с множеством вершин, в которых выполнено $f$. Значит, $\tau(Z) \subseteq \tau(Z')$.

**Лемма 2.** Для любого состояния из $F = \cap_{i=1}^{\infty} \tau^i(S)$ выполнена формула $EGf$.
Заметим, что $F$ — неподвижная точка оператора $\tau$ ($\tau$ монотонен). Значит, $\tau(F) = F$. Иными словами, в каждой вершине $F$ выполяется $f$ и из каждой вершины $F$ есть ребро в другую вершину $F$. Тогда взяв любую вершину $v_0 \in F$, мы можем перейти из нее в другую вершину $v_1 \in F$ и так далее. Значит, для любой вершины $v_0$ существует какой-то путь (очевидно, в нем будут циклы) $v_0 \ldots v_i \ldots$, где $\forall! v_j \in F$, значит, для каждой вершины этого пути выполняется $f$.

**Лемма 3.** Все состояния, для которых выполнено $EGf$ попали в $\cap_{i=1}^{\infty} \tau^i(S)$.
Рассмотрим произвольную вершину $x \in EGf$. Во-первых, $x \in S$. Из $x$ есть бесконечный путь, в каждой вершине которого выполняется $f$. Значит, из $x$ есть ребро в $S$. Значит, $x \in \tau(S)$. Но из $x$ есть ребро в вершину, в которой выполняется $f$ и из которой тоже есть бесконечный путь, на всех вершинах котрого выполянется $f$. Значит, $x \in \tau^2(S)$. Таким образом, по

индукции доказывается, что $\forall x \in \tau^i(S)$. А из этого утверждения следует доказываемая лемма.

Объединив Леммы 2 и 3, получим, что выполняющее множество $\mathbf{EG}f$ является наибольшей неподвижной точкой оператора $\tau(Z) = f \wedge \mathbf{EX}Z$.

На основе полученных результатов построим OBDD для $EGf$:

1. Начинаем с OBDD для $f_S$
2. Последовательно вычисляем OBDD для $(f \wedge \mathbf{EX})^i(S)$
3. Останавливаемся, когда степень $i + 1$ равна $i$-ой

Аналогичным образом строится OBDD для функции $\mathbf{E}[f\mathbf{U}g]$. Таким образом, мы научились строить выполняющее множетсве любой формулы CTL для заданной модели Крипке.

## Список литературы


[1] "Model checking"book, главы 5-6 [url]
(http://mitpress.mit.edu/catalog/item/default.asp?ttype=2&tid=3730)

[2] Перевод: "Верификация моделей программ"
(http://www.fizmatkniga.ru/product_info.php?products_id=1446)

[3] Курс Кларка, [URL]
(http://www.cs.cmu.edu/ẽmc/15-820A/reading/)