

**Def. 1** Пусть  $X = \{x_i\}_{i=1}^{\infty}$  - множество переменных,  $C = \{c_i\}_{i=1}^{\infty}$  - множество констант.  $\mathbb{F}$ -множество формул, такое что:

1.  $c_i \in \mathbb{F}$ ;
2.  $x_i \in \mathbb{F}$ ;
3.  $\forall f, g \in \mathbb{F} \Rightarrow \vee(f, g), \wedge(f, g), \neg f \in \mathbb{F}$ .  
 $\vee(f, g), \wedge(f, g)$  обозначим через  $f \vee g, f \wedge g$ .

**Def. 2** Уравнением  $e$  называется пара  $(f_1, f_2)$ , где  $f_1, f_2 \in \mathbb{F}$ .  
 $E$ - множество всех уравнений.  
Уравнение называется **простейшим**, если оно имеет вид  $(x_i, f)$ , где  $f$  - произвольная формула или  $(c_i, f)$ , где  $f \neq x_j, \forall j \in \mathbb{N}$ .

**Def. 3**  $P$ -множество предикатов, такое что:

1.  $f == g \in P$ ;
2.  $f \subset g \in P$ ;
3.  $f \equiv g \in P$ .

**Def. 4**  $U$ - множество условий, таких что:

1.  $TRUE, FALSE \in U$
2.  $p \in U$ , где  $p \in P$ ;
3.  $\forall u_1, u_2 \in U \Rightarrow \vee(u_1, u_2), \wedge(u_1, u_2), \bar{u}_1 \in U$ .

**Def. 5** Деревом называется  $T = (V, \phi, \psi, \alpha, \beta)$ , где:

- $V$  - множество вершин,
- $\phi : V \rightarrow (V \times V) \cup V \cup \emptyset$  - функция потомков;
- $\psi : V \rightarrow V \cup \emptyset$  - функция предков;
- $\alpha : V \rightarrow U$  - функция условия;
- $\beta : V \rightarrow \{e_i\}_{i=1}^n \cup \emptyset, n \in \mathbb{N}, e \in E$  - функция уравнений.

**Алгоритм 1** Алгоритм сведения уравнения к системе простейших уравнений:

Шаг 1. а) Если  $e = (\wedge(f, g), \wedge(u, v))$ , то выполняем шаг 1 для  $e_1 = (f, u)$  и  $e_2 = (g, v)$ . Результат решений объединяем;

б) Если  $e = (\vee(f, g), \vee(u, v))$ , то выполняем шаг 1 для  $e_1 = (f, u)$  и  $e_2 = (g, v)$ . Результат решений объединяем;

в) Если  $e = (\neg f, \neg u)$ , то выполняем шаг 1 для  $e_1 = (f, u)$ ;

г) Если  $e$  -простейшее, то результат -  $e$ ;

д) Если  $e = (f, x_i)$ , то результат -  $(x_i, f)$ ;

- е) Если  $e = (f, c_i)$ , то результат -  $(c_i, f)$ ;  
 ж) Если не выполнены пункты а) — е), то результатом является  $\emptyset$ .

**Def. 6**  $Vars : F \rightarrow 2^X$ , такая что:

1.  $Vars(x_i) = \{x_i\}$ ;
2.  $Vars(c_i) = \emptyset$ ;
3.  $Vars(\vee(f, g)) = Vars(f) \cup Vars(g)$ ;
4.  $Vars(\wedge(f, g)) = Vars(f) \cup Vars(g)$ ;
5.  $Vars(\neg f) = Vars(f)$ .

**Def. 7** Простейшее уравнение называется **разрешимым** если:

1.  $e = (c_i, c_i), \forall i \in \mathbb{N}$ ;
2.  $e = (x_i, f) \Leftrightarrow x_i \notin Vars(f)$ .

**Алгоритм 2** Алгоритм подстановки переменной  $(x_i, f)$  в другое уравнение  $g$ :

- а) Если  $g = x_i$ , то ответ  $f$ ;
- б) Если  $g = \neg u$ , то ответ  $\neg l$ , где  $l$  - подстановка  $(x_i, f)$  в  $u$ ;
- в)  $\vee(f, g)$ , то ответ  $\vee(l, t)$ , где  $l$  - подстановка  $(x_i, f)$  в  $u$ ,  $t$  - подстановка  $(x_i, f)$  в  $g$ ;
- г)  $\wedge(f, g)$ , то ответ  $\wedge(l, t)$ , где  $l$  - подстановка  $(x_i, f)$  в  $u$ ,  $t$  - подстановка  $(x_i, f)$  в  $g$ ;
- д) Если  $g = c_i$ , то ответ  $c_i$ ;
- е) Если  $g = x_j, i \neq j$ , то ответ  $x_j$ .

**Алгоритм 3** Алгоритм выражения переменных через константы и другие переменные:

Шаг 1. Применяем **Алгоритм 1**. Если получено  $\emptyset$ , то выразить ничего нельзя. Переходим к шагу 2.

Шаг 2. Если среди уравнений вида  $e = (c_i, f)$  все разрешимые, то удаляем их из системы и переходим к шагу 3, иначе ничего нельзя выразить.

Шаг 3. Если у нас есть уравнения вида  $e_1 = (x_i, f)$  и  $e_2 = (x_i, f)$ , то удаляем  $e_2$ , переходим к шагу 3, иначе переходим к шагу 4.

Шаг 4. Если у нас есть уравнения вида  $e_1 = (x_i, f)$  и  $e_2 = (x_i, g)$ , то удаляем  $e_2$ , и переходим к шагу 1 для уравнения  $e_3 = (g, f)$ . Иначе переходим к шагу 5.

Шаг 5. Если оставшиеся уравнения разрешимы, то переходим к шагу 6, иначе ничего нельзя выразить.

Шаг 6. Если у нас есть уравнения вида  $e_1 = (x_i, f)$  и  $e_2 = (x_j, g)$ , где  $x_j \in Vars(f)$ , то по **Алгоритму 2** подставляем в уравнение  $e_1$  выражение для  $x_j$  и переходим к шагу 5. Иначе выводим систему выражения переменных.

**Алгоритм 4** Алгоритм проверки синтаксического равенства  $f == g$  при известных выражениях  $X_i = \{x_{i_1} \dots x_{i_n}\}$ ,  $i = 1, \dots, n$ :

Шаг 1. Решаем уравнение  $(f, g)$  по **Алгоритму 1**. Получаем  $T = \{t_i\}_{i=1}^n$ - систему простейших. Если  $T = \emptyset$ , то *FALSE*.

Шаг 2.  $\forall i$  удаляем  $c_i = c_i$  и  $x_i = x_i$ ;

Если  $\exists x_j \in X_i$ , то по **Алгоритму 2** подставляем  $x_j$  в уравнение  $e = (x_j, f)$  и переходим к шагу 1 для  $e$ , результат объединяем. Иначе переходим к шагу 3.

Шаг 3. Если  $T = \emptyset$  - *TRUE*, иначе *NOTDEF*.

**Алгоритм 5** Алгоритм проверки подформульного предиката  $f \subset g$  при известных  $X_i = \{x_{i_1} \dots x_{i_n}\}$ ,  $i = 1, \dots, n$ :

Шаг 1. Применяем **Алгоритм 4** к  $f == g$ . Пусть  $q_1$  её результат.

Если  $q_1 = TRUE$ , то выводим *TRUE*, иначе переходим к шагу 2.

Шаг 2.а) Если  $g = c_i$ , то выводим  $q_1$ ;

б) Если  $g = x_i$ , то выводим  $q_1$ ;

в) Если  $g = \neg h$ , то применяем шаг 1 к  $f \subset h$  и получаем результат  $q_2$ . Если  $q_2 = TRUE$ , то выводим *TRUE*, если  $q_1$  или  $q_2$  равны *NOTDEF*, то возвращаем *NOTDEF*, иначе возвращаем *FALSE*.

г) Если  $g = \vee(h_1 h_2)$  или  $g = \wedge(h_1 h_2)$ , то применяем Алгоритм 5 к  $f \subset h$  и получаем результат  $q_2$ . Если  $q_2 = TRUE$ , то возвращаем  $q_2$ , иначе применяем шаг 1 к  $f \subset h_2$  и получаем результат  $q_3$ . Если  $q_3 = TRUE$ , то возвращаем *TRUE*. Если одна из  $q_1, q_2, q_3 = NOTDEF$ , то возвращаем *NOTDEF*. Иначе, возвращаем *FALSE*.

Введём элементарные эквивалентности:

1)  $u \wedge FALSE \equiv FALSE$ ;

2)  $u \wedge TRUE \equiv u$ ;

3)  $u \vee FALSE \equiv u$ ;

4)  $u \vee TRUE \equiv TRUE$ ;

5)  $u \wedge \neg u \equiv FALSE$ ;

6)  $u \vee \neg u \equiv TRUE$ ;

7)  $\neg \neg u \equiv u$ ;

8)  $u \wedge (u \vee v) \equiv u$ ;

9)  $u \vee (u \wedge v) \equiv u$ ;

10)  $u \vee u \equiv u$ ;

11)  $u \wedge u \equiv u$ .

Таблицы истинности для функций  $\wedge$ ,  $\vee$ ,  $\neg$  определяются аналогично таблицам алгебры логики.

**Алгоритм 6** Алгоритм проверки функционального предиката  $f \equiv g$  при известных  $X_i = \{x_{i_1} \dots x_{i_n}\}$ ,  $i = 1, \dots, n$ :

Шаг 1. Упрощаем  $f$  и  $g$  по эквивалентностям, пока возможно. Переходим к шагу 2.

Шаг 2. Подставляем по **Алгоритму 2**  $X_i = \{x_{i_1} \dots x_{i_n}\}$  в  $f$  и  $g$ . Если получили  $TRUE$ , то возвращаем  $TRUE$ . Если получили  $FALSE$ , то возвращаем  $FALSE$ .

**Def. 9** Вершина дерева  $v$  называется **вершиной уравнений**, если  $\phi(v) = w$ ,  $w \in V$ , где  $V$  из **Def.5**.

**Def. 10** Вершина дерева  $v$  называется **выходом**, если  $\phi(v) = \emptyset$ .

**Def. 11** Вершина дерева  $v$  называется **вершиной условия**, если  $\phi(v) = (w, w')$ ,  $w, w' \in V$ .

**Def. 12** Вершина дерева  $v$  называется **корнем дерева**, если  $\psi(v) = \emptyset$ .

**Алгоритм 7** Алгоритм проверки условий для оставшихся уравнений:

По эквивалентностям упрощаем условие, пока это возможно. Затем по **Алгоритму 4**, **Алгоритму 5**, **Алгоритму 6** находим значения для всех выражений. Если все значения  $TRUE$ , то возвращаем  $TRUE$ . Если все значения  $FALSE$ , то возвращаем  $FALSE$ . Если  $\exists$  значения одновременно  $TRUE$  и  $FALSE$ , то возвращаем  $NOTDEF$ .

**Алгоритм 8** Алгоритм разрешения последующих условий относительно данной вершины условия:

Пусть  $v$  текущая вершина условия. Пусть  $\phi(v) = (v_1, v_2)$ . Пусть  $\phi(w) = (w_1, w_2)$ . Применяем шаг 1 и шаг 2 соответственно к вершинам  $v_1, v_2$ .

Шаг 1. а) Если текущая вершина  $w$  - вершина уравнений, то применяем шаг 1 к вершине  $\phi(w)$ .

б) Если текущая вершина  $w$  - вершина условия, то проверяем условие  $\wedge(\alpha(v), \alpha(w))$  по **Алгоритму 7**. Если результат - *TRUE*, то "отрезаем правую ветку" вершины  $w$  и применяем шаг 1 к  $w_1$ . Если результат - *FALSE*, то "отрезаем левую ветку" вершины  $w$  и применяем шаг 1 к  $w_2$ . Если результат - *NOTDEF*, то применяем шаг 1 к  $w_1$  и  $w_2$ .

Шаг 2. а) Если текущая вершина  $w$  - вершина уравнений, то применяем шаг 2 к вершине  $\phi(w)$ .

б) Если текущая вершина  $w$  - вершина условия, то проверяем условие  $\vee(\alpha(v), \alpha(w))$  по **Алгоритму 7**. Если результат - *TRUE*, то "отрезаем правую ветку" вершины  $w$  и применяем шаг 2 к  $w_1$ . Если результат - *FALSE*, то "отрезаем левую ветку" вершины  $w$  и применяем шаг 2 к  $w_2$ . Если результат - *NOTDEF*, то применяем шаг 2 к  $w_1$  и  $w_2$ .

**Алгоритм 9** Алгоритм согласования выражения всех переменных уравнения  $e$ :

Пусть есть  $n$  выражений, добавляем в каждое выражение переменные, выраженные в уравнении  $e$ . Затем проверяем на противоречивость все  $n$  выражений, как в **Алгоритме 3**. Если получили противорочение во всех  $n$  системах, то удаляем уравнение, иначе оставляем только непротиворечивые выражения.

### Алгоритм упрощения дерева 1

Пусть  $v$  - текущая вершина. Начиная с корня дерева, выполняем следующие шаги:

Шаг 1. а) Если  $v$  - вершина условия, то применяем **Алгоритм 7** к  $\alpha(v)$ , если *TRUE*, то "отрезаем правую ветку". Если *FALSE*, то "отрезаем левую ветку". Иначе, при переходе в "левую ветку" рассматриваем только те выражения, которые дали *TRUE* на данном условии, а при переходе в "левую ветку" рассматриваем только те выражения, которые дали *FALSE* на данном условии. Далее переходим к шагу 2.

б) Если  $v$  - вершина уравнений, то пусть  $\beta : V \rightarrow \{e_i\}_{i=1}^n, n \in \mathbb{N}, e \in E$ , тогда  $\forall i$ , где  $1 \leq i \leq n$  выражаем переменные по **Алгоритму 3**. Если  $e_i$  не разрешимо, то удаляем это уравнение. Далее по **Алгоритму 8** проверяем условие  $\alpha(v)$  для оставшихся уравнений. Удаляем уравнения, для которых условие не выполнено. Далее переходим к шагу 3.

Шаг 2. Применяем **Алгоритм 8**. Пусть  $\phi(v) = (w, w')$ , тогда приме-

няем шаг 1 к вершинам  $w, w'$ .

Шаг 3. Применяем **Алгоритм 9** на согласование всех выражений.  
Применяем шаг 1 к вершине  $\phi(v)$ .