МОДИФИКАЦИИ DPLL

Гусева Екатерина 6373

Расширения алгоритма DPLL

Текущие исследования по улучшению алгоритма выполняются в трех направлениях:
1) Определение различных оптимизирующих методов выбора «переменной ветвления»
2) Разработка новых структур данных для ускорения работы алгоритма, особенно для распространения переменной
3) Разработка различных вариантов базового алгоритма перебора с возвратом.

Последнее направление включает «***нехронологические возвраты***» и ***запоминание плохих случаев (конфликтов)***.
Эти улучшения можно описать как метод возврата после достижения ложной дизъюнкты, при котором запоминается, какое именно присвоение значения переменной повлекло этот результат, чтобы в дальнейшем избежать точно такой же последовательности вычислений, тем самым сократив время работы.

Определения и обозначения

Клауза – дизъюнкция конечного числа литералов

$$(x1\ v\ x2\ v\ x3)\ \&\ (x2\ v\ x5)$$

клаузы

Приписание - множество $\cup_{i \in X}\{(x_i, \sigma_i)\}$ для X = {1,2 … n} (набор значений x_i)

приписание (полное)

$$\begin{cases} x1 = 1 \\ x2 = 1 \\ x3 = 0 \\ x5 = 0 \end{cases}$$

частичное приписание

Если в приписании определена только часть переменных (какие-то переменные пока не имеют значений), такое приписание называется **частичным**. Если определены все x, то приписание называется **полным**.

Использование необходимых приписаний

Если в какой-то клаузе

$$\omega_k = (\mathrm{x}_1 \mathrm{V}\, \mathrm{x}_2\, \mathrm{V}\ \ldots \mathrm{V}\, \mathrm{x}_\mathrm{n})$$

определены $n - 1$ значений x, и клауза на этом наборе обращается в ноль, то нужно *выбрать такое значение* для оставшейся неопределенной переменной x_n, чтобы клауза ω_k стала равна 1. Такое присваивание называется **необходимым приписанием.**

Использование необходимых приписаний

Рассмотрим пример

Пусть заданы $\omega_1 = (\mathrm{x1\ V\ x2\ V\ x3}), \quad \{x1 = 0, x2 = 0\}.$

Значит, чтобы

$$\omega_1 = 1$$

необходимо присвоить

$$x3 = 1.$$

В противном случае $\omega_1 = (\mathrm{x1\ V\ x2\ V\ x3}) = (0\ V\ 0\ V\ 0) = 0.$

Если в КНФ хотя бы одна клауза $\omega_k = 0,$ то и вся формула будет **невыполнима**. Значит, бессмысленно рассматривать наборы значений (например, $\{x1 = 0, x2 = 0, \mathrm{x3} = 0\}$), заведомо обращающие одну из клауз в ноль. Использование необходимых приписаний позволяет сократить количество возможных наборов значений переменных, для которых проверяется выполнимость формулы.

Пример использования

На рисунке приведена булева функция и частично построенное дерево разбора с применением необходимых присваиваний.

1) Сначала делаем предположение, что x1 = 1

2) Потом x4 = 1

(на рисунке изображены кривыми).

Из клаузы

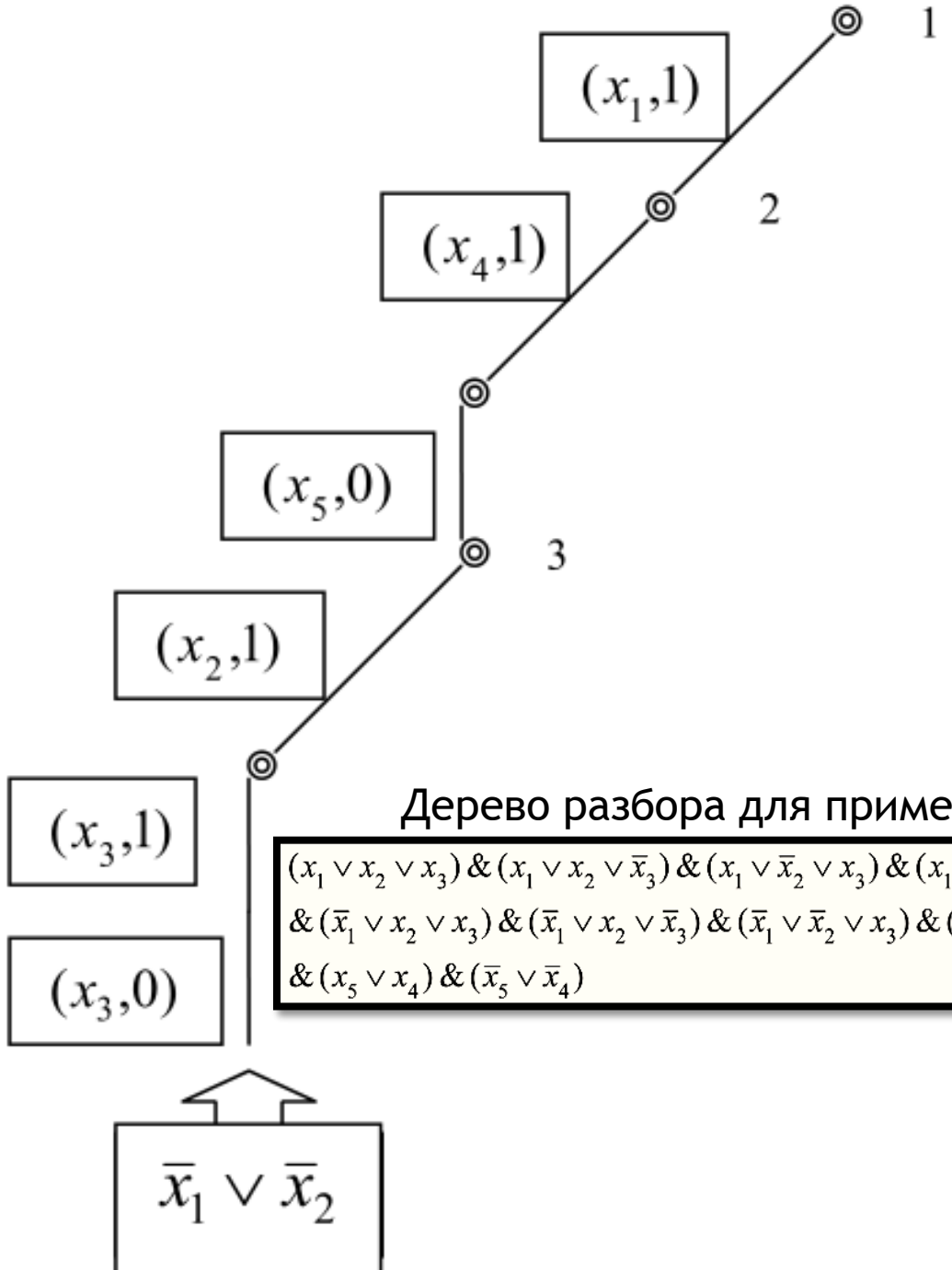
$$(\overline{x_5} \vee \overline{x_4})$$

следует **обязательное присваивание**

x5 = 0

(на рисунке изображено прямой).

Таким образом, одним таким присваиванием мы исключаем перебор четырех наборов возможных значений (если бы при {x1 = 1, x4 = 1, x5 = 1} мы бы продолжили перебирать значения для x2 и x3).



Дерево разбора для примера

$$(x_1 \vee x_2 \vee x_3) \& (x_1 \vee x_2 \vee \overline{x}_3) \& (x_1 \vee \overline{x}_2 \vee x_3) \& (x_1 \vee \overline{x}_2 \vee \overline{x}_3) \&$$
$$\& (\overline{x}_1 \vee x_2 \vee x_3) \& (\overline{x}_1 \vee x_2 \vee \overline{x}_3) \& (\overline{x}_1 \vee \overline{x}_2 \vee x_3) \& (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3) \&$$
$$\& (x_5 \vee x_4) \& (\overline{x}_5 \vee \overline{x}_4)$$

Возникновение конфликта

Если продолжить предполагать дальше, что

3) x2 = 1,

то из клаузы

$$(x_1 V\ x_2\ V\ x_3)$$

следует, что для x3 необходимо присвоить

x3 = 1.

Однако, из клаузы

$$(\overline{x_1}\ V\ \overline{x_2}\ V\ \overline{x_3})$$

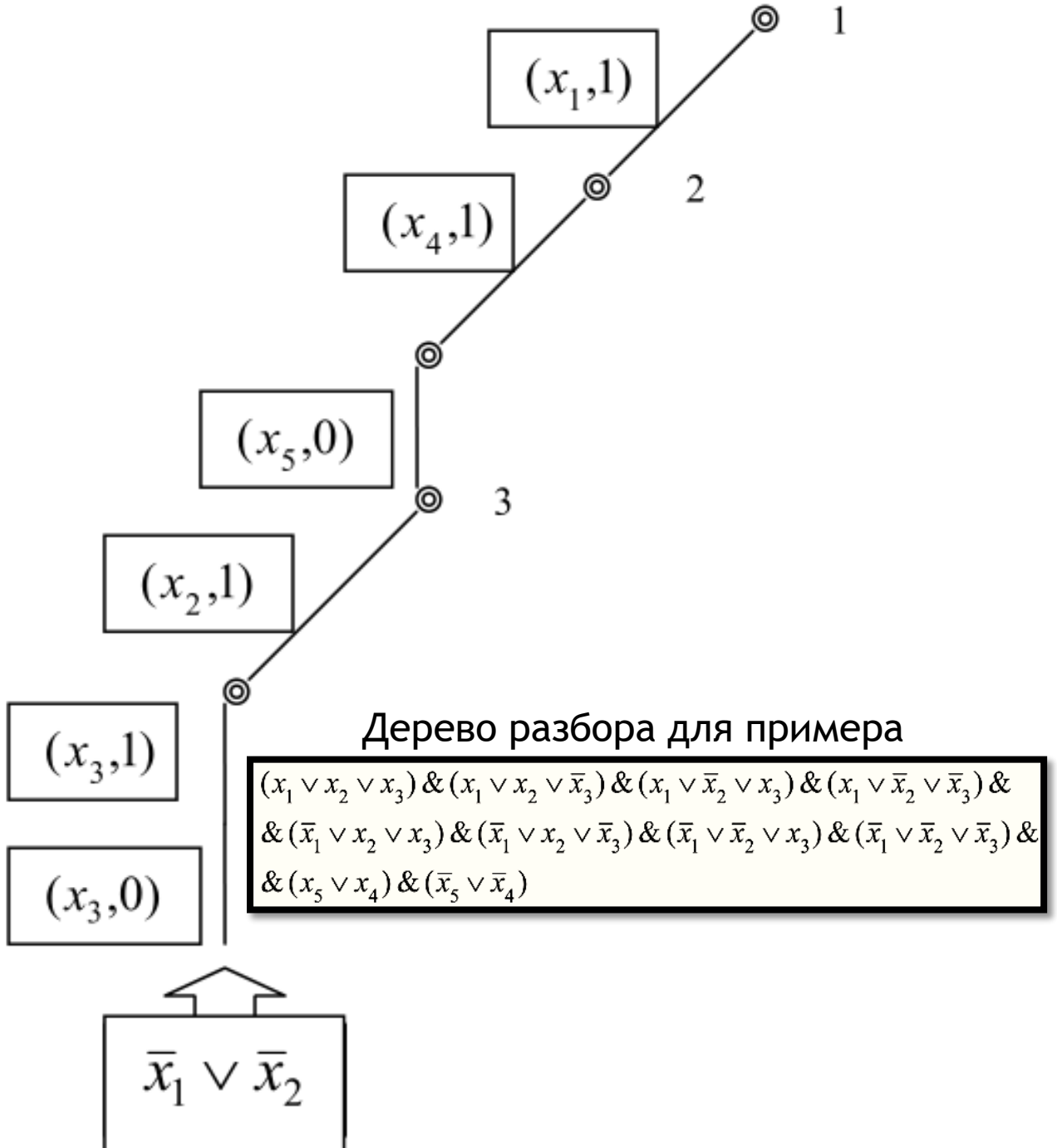
следует необходимость присваивания

x3 = 0.

Такая ситуация называется конфликтом.

Конфликт указывает на то, что выбранный метод присвоения переменных не подходит для определения выполняющего набора, значит, при выборе произвольных присваиваний мы должны рассмотреть другие варианты присвоения переменных.

Если во всей формуле при одном из наборов значений переменных не возникнет конфликта, значит, мы нашли набор, на котором булева формула будет выполнимой



Дерево разбора для примера

$$(x_1 \vee x_2 \vee x_3)\,\&\,(x_1 \vee x_2 \vee \overline{x}_3)\,\&\,(x_1 \vee \overline{x}_2 \vee x_3)\,\&\,(x_1 \vee \overline{x}_2 \vee \overline{x}_3)\,\&$$
$$\&\,(\overline{x}_1 \vee x_2 \vee x_3)\,\&\,(\overline{x}_1 \vee x_2 \vee \overline{x}_3)\,\&\,(\overline{x}_1 \vee \overline{x}_2 \vee x_3)\,\&\,(\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3)\,\&$$
$$\&\,(x_5 \vee x_4)\,\&\,(\overline{x}_5 \vee \overline{x}_4)$$

Запись клауз (Clause Recording Schemes)

Идея заключается в следующем:

После каждого конфликта определяется множество литералов, которого достаточно для его получения. На основе этого множества создаётся клауза (конфликтная клауза – conflict clause или изученная клауза), которая записывается (добавляется к КНФ).

В классическом алгоритме DLL каждое приписание в узле дерева имеет флаг -исследовалось ли обратное приписание. При конфликте ищется верхний уровень, для которого не просматривалось обратное приписание.

При реализации алгоритма с конфликтными клаузами необходимость во флаге отпадает: по дереву поднимаемся пока конфликтная клауза равна нулю, затем выполняем приписание, обращающее её в единицу.

Получение по графу конфликтных клауз

$\omega_1 = (\neg x_1 + x_2)$

$\omega_2 = (\neg x_1 + x_3 + x_9)$

$\omega_3 = (\neg x_2 + \neg x_3 + x_4)$

$\omega_4 = (\neg x_4 + x_5 + x_{10})$

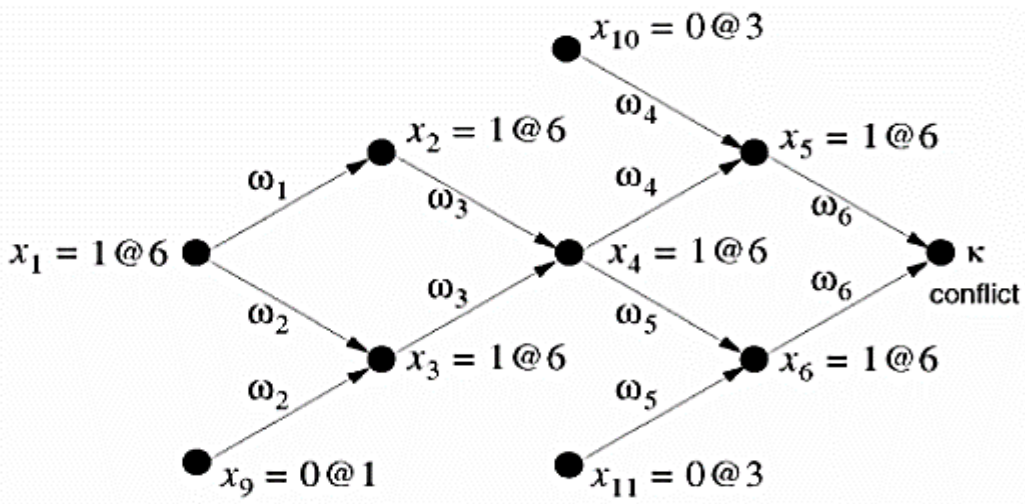
$\omega_5 = (\neg x_4 + x_6 + x_{11})$

$\omega_6 = (\neg x_5 + \neg x_6)$

$\omega_7 = (x_1 + x_7 + \neg x_{12})$

$\omega_8 = (x_1 + x_8)$

$\omega_9 = (\neg x_7 + \neg x_8 + \neg x_{13})$



Рассмотрим граф, на котором условно изображены приписания, сделанные во время последовательного присваивания значений для переменных.

Выражение $x_i = n \ @ \ k$ означает, что данное присваивание n было совершено на уровне k (уровень - число принятых решений о значении переменных в текущем приписании).

А ω_k - номер клаузы, анализ которой повлек за собой обязательное присваивание.

Получение по графу конфликтных клауз

$\omega_1 = (\neg x_1 + x_2)$

$\omega_2 = (\neg x_1 + x_3 + x_9)$

$\omega_3 = (\neg x_2 + \neg x_3 + x_4)$

$\omega_4 = (\neg x_4 + x_5 + x_{10})$

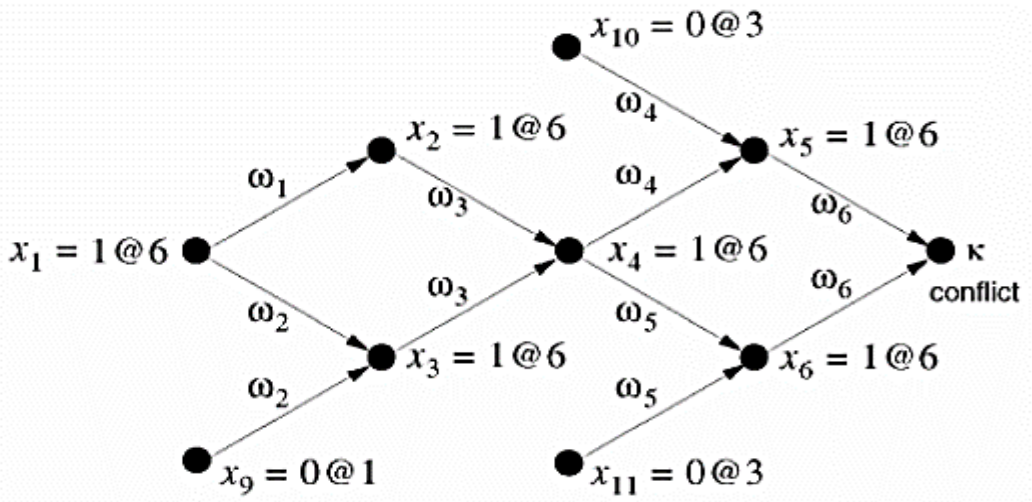
$\omega_5 = (\neg x_4 + x_6 + x_{11})$

$\omega_6 = (\neg x_5 + \neg x_6)$

$\omega_7 = (x_1 + x_7 + \neg x_{12})$

$\omega_8 = (x_1 + x_8)$

$\omega_9 = (\neg x_7 + \neg x_8 + \neg x_{13})$



Вершины, **в которые не входят ребра**, отражают переменные, значения которых были присвоены произвольно (мы предположили, что данная переменная будет иметь такое значение во время перебора возможных наборов).

Вершины же, **в которые ребра входят**, отражают переменные, которым значение присваивается вынужденно (необходимые приписания) на основе анализа клауз ω_k.

Получение по графу конфликтных клауз

$\omega_1 = (\neg x_1 + x_2)$

$\omega_2 = (\neg x_1 + x_3 + x_9)$

$\omega_3 = (\neg x_2 + \neg x_3 + x_4)$

$\omega_4 = (\neg x_4 + x_5 + x_{10})$

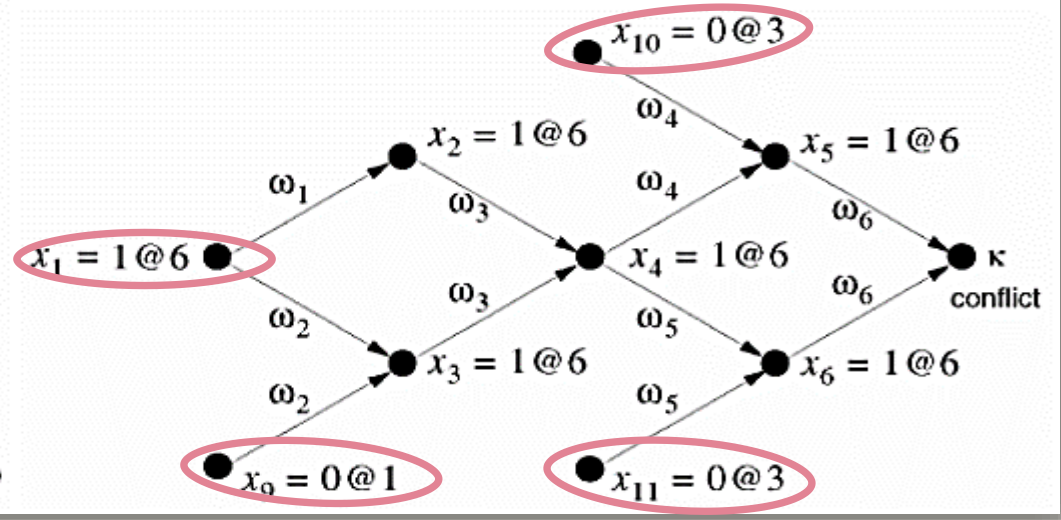
$\omega_5 = (\neg x_4 + x_6 + x_{11})$

$\omega_6 = (\neg x_5 + \neg x_6)$

$\omega_7 = (x_1 + x_7 + \neg x_{12})$

$\omega_8 = (x_1 + x_8)$

$\omega_9 = (\neg x_7 + \neg x_8 + \neg x_{13})$



Нетрудно видеть, что к конфликту привела часть текущего приписания {x1 = 1, x9 = 0, x10 = 0, x11 = 0}. Если посмотреть на каких уровнях были присвоены значения переменных, то можно заметить, что последним присваивалось значение для x1 (на уровне шесть, тогда как остальные значения были присвоены на уровнях выше).

Получение по графу конфликтных клауз

$\omega_1 = (\neg x_1 + x_2)$

$\omega_2 = (\neg x_1 + x_3 + x_9)$

$\omega_3 = (\neg x_2 + \neg x_3 + x_4)$

$\omega_4 = (\neg x_4 + x_5 + x_{10})$

$\omega_5 = (\neg x_4 + x_6 + x_{11})$

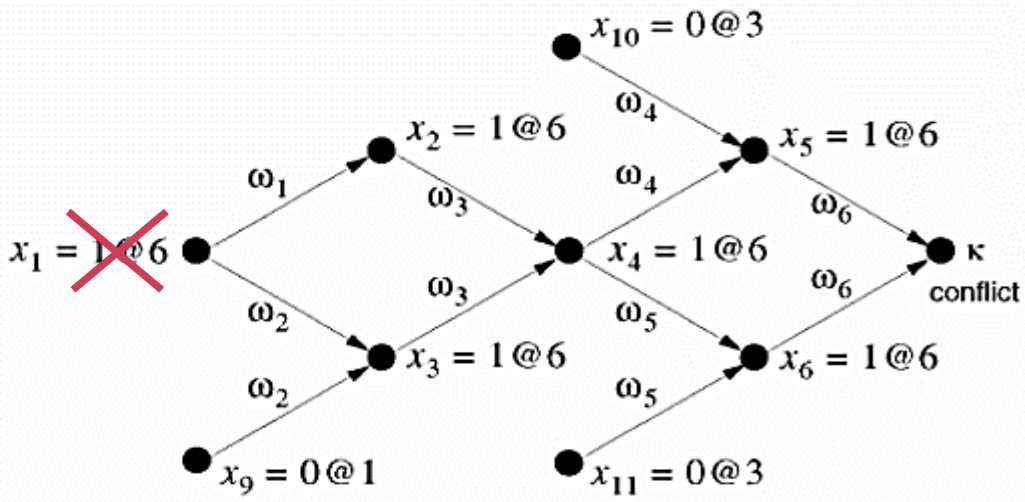
$\omega_6 = (\neg x_5 + \neg x_6)$

$\omega_7 = (x_1 + x_7 + \neg x_{12})$

$\omega_8 = (x_1 + x_8)$

$\omega_9 = (\neg x_7 + \neg x_8 + \neg x_{13})$

+ $\omega_{10} = (\overline{x_1}\ V\ x_9\ V\ x_{10}\ V\ x_{11})$



Тогда, если мы пополним КНФ конфликтной клаузой

$$\omega_{10} = (\overline{x_1}\ V\ x_9\ V\ x_{10}\ V\ x_{11})$$

То на уровне шесть анализ клауз определит для x1 вынужденное присваивание противоположного значения, значит, не произойдёт “плохого” приписания, которое повлекло за собой данный конфликт.

Таким образом мы “учитываем” ошибку и впредь набор значений {x1 = 1, x9 = 0, x10 = 0, x11 = 0} сразу будет исключаться