BACKTRACKING A DPLL

proc backtracking

if $\square \in F$
  return 0

if $F = \emptyset$
  return 1

$x \leftarrow$ proměnná z $Var(F)$

if backtracking($F\{x=0\}$)
  return 1
else
  return backtracking($F\{x=1\}$)

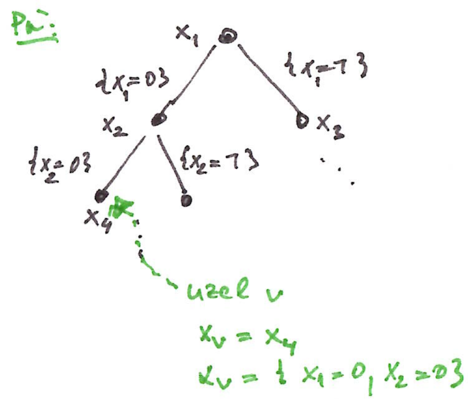
Jedná se o „meta"-algoritmus: není určeno, jak vybírat x. Doplněním nějakého způsobu získáme algoritmus.

Věta: Pro nesplnitelnou F a pro minimální počet r rekurzivních zavolání potřebných v algoritmu backtrack (pro libovolné zvolené pravidlo výběru x) existuje rezoluční zamítnutí F délky nejvýše r.

Důkaz: Ve stromu rekurze algoritmu backtrack má každý uzel v vyjma listů proměnnou x_v: je to proměnná, která byla vybrána v algoritmu v daném volání jako x.

V každém uzlu v pak máme ohodnocení α_v, které získáme na cestou z kořene do v.

Př:



Ukážeme, že každému uzlu ve stromu můžeme přiřadit klauzuli C_v, která

1) patří do F, nebo je z klauzulí F dosažena pomocí rezoluce. (ne nutně jedním krokem)

2) $C_v \alpha_v = 0$

Pokud je v list, pak je C_v libovolná klauzule z F, která je nepravdivá pro α_v. (a platí tak 1) a 2) výše).

Pro uzel v, který není listem (a má nutně 2 potomky) předpokládejme (indukce), že pro jeho potomky u, w máme C_u, C_w tak, že platí 1) a 2).

Potom:

a) $x_v \in Var(C_u) \cap Var(C_w)$.

nutně jedna z klauzulí C_u, C_w obsahuje x_v a druhá $\overline{x_v}$ (protože α_u, α_w obě „zneplatňují" C_u, resp C_w a liší se ohodnocením x_v).

Věta: Unit rezoluce je úplná vzhledem k zamítnutí, pokud je použita pro Hornovské formule.

Důkaz:

Musíme ukázat, že pokud je Hornova fle F kontradikce, pak unit rezoluce odvodíme $\square$. Důkaz provedeme indukcí přes počet proměnných v F, který označíme n.

1) situace je triviální pokud $n \leq 1$.

2) Pokud je F kontradikce, musí obsahovat pozitivní klausuli. Protože F je Hornova, je to klausule $\{x\}$ pro nějakou proměnnou x.

Provedeme ~~ze~~ unit-resoluční kroky ~~pro~~ ~~C~~ pro C, x, kde C ctenýr přes klausule z F obsahující $\overline{x}$. Tak dostaneme klausule z $F\{x=1\}$.

Použijeme indukční předpoklad a unit-resoluční zamítnutí pro $F\{x=1\}$ umístíme ~~za~~ nakonec za $\circledast$. Tak dostaneme zamítnutí F.

∎

Věta: Vstupní resoluce je úplná vzhledem k zamítnutí, pokud je použita pro Hornovy fle.

DPLL (F)

```
if □ ∈ F then return 0
if F = ∅ then return 1
if {u} ∈ F then return DPLL(F{u=1})
if u obsahuje pure literál u
    then return DPLL(F{u=1})

s adekvátní strategií vyber x ∈ Var(x)   (*)
if DPLL(F{x=0}) then return 1
return DPLL(F{x=1})
```

pure literál:
u je pure, pokud se v F nevyskytuje $\bar{u}$

F{u=1} a F jsou SAT-ekvivalentní

tedy víme, že {u=1}, jinak by klausule {u} nebyla splněna. Tomuto zjednodušení se říká unit-propagation

Autark ohodnocení

def: Ohodnocení α je autark pro F pokud pro všechny $C \in F$ platí: $Var(C) \cap Var(\alpha) \neq \emptyset$ implikuje $C\alpha = 1$.

Příkladem autark ohodnocení je {u=1}, kde u je pure literál.

thm: F ... Formule (v CNF)
α ... autark ohodnocení
Potom jsou F a $F\alpha$ SAT-ekvivalentní.

Důkaz: „⇐" předp. že $(F\alpha)\beta = 1$ pro ohodnocení β takové, že $Var(\alpha) \cap Var(\beta) = \emptyset$. Potom $F(\alpha \cup \beta) = 1$.

„⇒" předp. že ex. β tak, že $F\beta = 1$.
Ozn. β' ohodnocení, které je restrikcí β na proměnné z množiny $Var(F) \setminus Var(\alpha)$.
β' splní všechny klausule, které neobsahují proměnné z $Var(\alpha)$.
Tj. $(F\alpha)\beta' = 1$.

■

Příklady heuristik pro (*)

položíme $f_k(u)$ ... počet výskytů lit. u v klauzulích velikosti k

$F(u)$ = počet výskytů lit. u v F

DLIS
- vybereme u s max $F(u)$ a ohodnotíme {u=1} nejdřív a {u=0} případně později

DLCS
- vybereme proměnnou x s max $F(x) + F(\bar{x})$.
Pokud $F(x) \geq F(\bar{x})$, nejdříve {x=1}, jinak nejdříve {x=0}.

MOM
k ... délka nejkratší klausule
vybereme proměnnou x, která maximalizuje

$$\underbrace{(f_k(x) + f_k(\bar{x}))}_{\text{počet výskytů v nejkratších Clauses}} \cdot p + \underbrace{f_k(x) \cdot f_k(\bar{x})}_{\text{vyvažuje poz. a neg. literály}}$$

p zvolíme tak, aby $(f_k(x) + f_k(\bar{x})) > f_k(x) \cdot f_k(\bar{x})$

Böhmova heuristika

pro $\forall x \in Var(F)$ spočítáme $H(x) = (H_2(x), \dots, H_n(x))$

kde $H_i(x) = p_1 \cdot \max(f_i(x), f_i(\bar{x})) + p_2 \cdot \min(f_i(x), f_i(\bar{x}))$

p_1, p_2 volné parametry

vybereme x s maximálním H ~~podle~~ podle lex. uspořádání.

Je spousta dalších heuristik

Monien-Speckenmeyer Algoritmus

MS(F)

```
if □ ∈ F then return 0
if F = ∅ then return 1
C ∈ nejkratší klausule v F, předp C = {l_1, l_2, l_3 ... l_m}
for i ← 1 to m
    if MS(F{l_1=0, ..., l_{i-1}=0, l_i=1}) then return 1
return 0
```

Idea korektnosti:

- pokud zjistím, že $\{l_i = 1\}$ nevede ke splnění F, můžu předpokládat $\{l_i = 0\}$.

Na alg. se lze dívat jako na heuristiku pro DPLL, kde vybíráme literály z nejkratší klausule a nejdříve ohodnocujeme na 1.

To je proto, že pokud je $(l_1, \dots l_m)$ nejkratší klausule, je jí i $(l_2 \dots l_m)$.

Analýza:

$T(n)$ ... počet rekurzivních zavolání pro F s n proměnnými

m ... velikost vybrané klausule

n ... počet proměnných v F

Máme

$$T(n) \leq T(n-1) + T(n-2) + \dots T(n-m)$$

Pro k-CNF máme $m \leq k$. Předp. $T(n) = a^n$ pro nějaké a. Tím dostaneme (~~[illegible]~~)

$$a^k = a^{k-1} + \dots + a^1 + 1$$

$$= \frac{1-a^k}{1-a} \quad (\text{pro } a \neq 1)$$

Odtud

$$a^{k+1} + 1 = 2a^k$$

k	3	4	5	8	
	1.839	1.928	1.966	1.996	a
	0.879	...	...	0.997	$\log_2 a$

$$a^n = 2^{(\log_2 a) n}$$

jak se snížila výška stromu rekurze

Vylepšení pomocí autark assignment

MS2 (F)

if $\square \in F$ return 0

if $F = \emptyset$ return 1

$C \leftarrow$ nejkratší klauzule v F, ozn $\{l_1, \dots l_m\}$

for $i \leftarrow 1$ to m

  $\alpha_i \leftarrow \{l_1 = 0, l_2 = 0, \dots l_{i-1} = 0, l_i = 1\}$

  if α_i je autark

    return MS2$(F\alpha_i)$

for $i \leftarrow 1$ to m

  if MS2$(F\alpha_i)$ return 1

return

Analýza: pro k-SAT

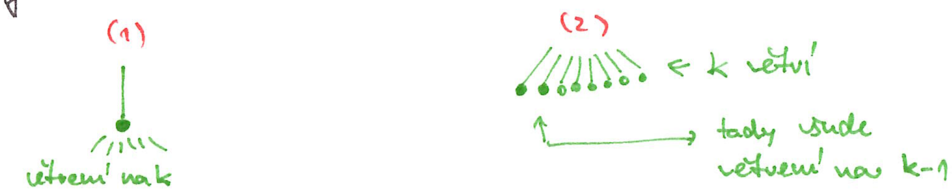
2 případy

- máme autark assignment $\to T(n) \le T(n-1)$ (1)
- nemáme autark asignment (2)

V rekurzivních zavoláních máme ~~...~~, ~~protože jsme~~ nejkratší klausule délky nejvýše $k-1$: autark assignment jsme nenašli $\to$ v klausulích je alespoň 1 nesplněný literál

položíme $T'(n)$ -- počet rek. zavolání za hypotézy, že v předchozím zavolání je větvení pouze do $k-1$ větví

Tj.



Dostaneme tak

$$T(n) = \max\{\underbrace{T(n-1)}_{①}, \underbrace{T'(n-1) + \dots\dots + T'(n-k)}_{(2)}\}$$

Pro $T'(n)$ provedeme obdobnou analýzu a dostaneme

$$T'(n) = \max\{T(n-1), T'(n-1) + \dots T'(n-k+1)\} \; !$$

Indukcí ukážeme

$$T(n-1) \le T'(n-1) + \dots + T'(n-k+1)$$

Pro $n = 2$ máme $T(1) = T'(1)$ (klausule mají max 1 literál)

Obecný krok: (předp. platnost pro $n-1$, dokážeme pro n).

$$\max \begin{cases} T(n-1) \le T'(n-2) + T'(n-3) + \dots + T'(n-k) \\ T'(n-1) + T'(n-2) + T'(n-3) + \dots + \cancel{T'(n-k)} \; T(n-k+1) \end{cases}$$

$\Rightarrow$ maximum je $T'(n-1) + T'(n-2) + \dots\dots T'(n-k)$, protože $T'(n-1) \ge T'(n-k)$.

Tj. máme

$$T(n) = T'(n-1) + \dots + T'(n-k) \quad \Rightarrow T(n) = O(T'(n))$$

$$T'(n) = T'(n-1) + \dots + T(n-k+1)$$

Hypotéza: $T'(n) = a^n$,

dostaneme

$$a^{k-1} = a^{k-2} + \dots + a^1 + 1$$

a podobně jako před zlepšením dostaneme

$$a^k + 1 = 2a^{k-1}.$$

k	3	4	...	8
a	1.618	1.839		1.992
$\log_2 a$	0.694	0.879		0.994