Algorithmen

NNF nach 3KNF

Mi,27.04.: Erdmann, Vogel; Fr,29.04.: FREI

Eingabe: Aussagenlogische Formel ψ in NNF

Setze $G := (V, E)$ mit $V = \{x, y\}$ und $E = \{(x \to y \colon \psi)\}$
while in G gibt es eine Kante $(u \to w \colon \alpha)$ und α ist kein Literal **do**
  if $\alpha = \alpha_1 \wedge \alpha_2$ **then**
    $E := (E \setminus \{(u \to w \colon \alpha)\}) \cup \{u \to w \colon \alpha_1, u \to w \colon \alpha_2\}$
  else if $\alpha = \alpha_1 \vee \alpha_2$ **then**
    $E := (E \setminus \{(u \to w \colon \alpha)\}) \cup \{u \to z_i \colon \alpha_1, z_i \to w \colon \alpha_2\}$
    $V := V \cup \{z_i\}$ für neue Variable z_i
$\Phi_G := \exists x \exists y \exists z_1 \cdots \exists z_t x \wedge \neg y \wedge \bigwedge_{(u \to w \colon \alpha) \in E} (\neg u \vee w \vee \alpha)$

Ausgabe: SP-graph G mit zugehöriger 3KNF Formel $\Phi_G \equiv \psi$

Monien-Speckenmeyer Algorithmen

Mi, 04.05.: Rauch, Schliephacke; Fr,06.05.: FREI

Algorithmus MS
Eingabe: Aussagenlogische Formel ϕ in k-KNF

if ϕ ist die leere Formel **then return** 1;
if ϕ enthält die leere Klausel **then return** 0;
Sei $K = \{\ell_1, \ldots, \ell_r\}$ die erste kürzeste Klausel in ϕ;
for $i = 1, \ldots, r$ **do**
  $\theta_i = \{\ell_1 = 0, \ldots, \ell_{i-1} = 0, \ell_i = 1\}$;
  if MS($\phi[\theta_i]$) **then Return** 1 **end if**
return 0;

Algorithmus MSa
Eingabe: Aussagenlogische Formel ϕ in k-KNF

if ϕ ist die leere Formel **then return** 1;
if ϕ enthält die leere Klausel **then return** 0;
Sei $K = \{\ell_1, \ldots, \ell_r\}$ die erste kürzeste Klausel in ϕ;
for $i = 1, \ldots, r$ **do**
  $\theta_i = \{\ell_1 = 0, \ldots, \ell_{i-1} = 0, \ell_i = 1\}$;

if θ_i ist autark für ϕ **then return** MSa($\phi[\theta_i]$);
for $i = 1, \ldots, r$ **do**
$\theta_i = \{\ell_1 = 0, \ldots, \ell_{i-1} = 0, \ell_i = 1\}$;
if MSa($\phi[\theta_i]$) **then return** 1;
return 0;

Randomisiert

Mi,11.05.: FREI; Fr,13.05.: Kramer, Oberschulte

Eingabe: Aussagenlogische Formel ϕ in k-KNF
Wähle zufällige Belegung θ;
for $j = 1, \ldots, m$ **do**
if $\theta \not\models C_j$ **then Repair**(C_j, θ, ϕ) **end if**
return θ;

Methode Repair
Eingabe: k-Klausel C, Belegung θ, Formel ϕ
Belege die k Variablen in C mit zufälligen Werten;
for $j = 1, \ldots, m$ **do**
if $\mathrm{Vars}(C_j) \cap \mathrm{Vars}(C) \neq \emptyset$ **then**
if $\theta \not\models \phi$ **then Repair**(C_j, θ, ϕ) **end if**

Krom-SAT

Mi,25.05.: Brosy, Scholz; Fr,27.05.: FREI

Eingabe: Krom-Formel ϕ
Wähle eine beliebige zufällige Anfangsbelegung θ über $\mathrm{Vars}(\phi)$.
if $\theta \models \phi$ **then**
akzeptiere
else
Sei $K = (u, v)$ eine Klausel in ϕ mit $\theta \not\models K$.
Mit W'keit jeweils $\frac{1}{2}$ ändere θ, sodass $\theta \models u$ oder $\theta \models v$.

Eingabe: Krom-Formel ϕ
Konstruiere den gerichteten Graphen $G = (V, E)$ wie oben beschrieben.
for jede Variable x in ϕ **do**
if es gibt einen Zyklus auf dem x und $\neg x$ liegen **then**
verwerfe
akzeptiere

Horn-SAT

Mi,01.06.: Kroll; Fr,03.06.: Friedrich, Prenner

Eingabe: Horn-Formel ϕ

Belegung θ mit $\theta(x) = 0$ für $x \in \mathrm{Vars}(\phi)$;
while $\{x\} \in \phi$ für eine Variable x **do**
    $\theta := \theta \cup \{x = 1\}$;
    $\phi := \phi[\theta]$;
if $\square \in \phi$ **then verwerfe end if**
akzeptiere

DPLL

Mi,08.06.: Glade; Fr,10.06.: Obaidi+1

Algorithmus Meta-DPLL
Eingabe: Klauselmenge ϕ
Ausgabe: Akzeptiert gdw. ϕ erfüllbar ist.

if $\square \in \phi$ **then verwerfe end if**
if $\phi = \emptyset$ **then akzeptiere end if**
if ϕ enthält Einheitsklausel $\{u\}$ **then return Meta-DPLL**$(\phi[u = 1])$ **end if**
if ϕ enthält pures Literal u **then return Meta-DPLL**$(\phi[u = 1])$ **end if**
Wähle eine Variable $x \in \mathrm{Vars}(\phi)$;
if Meta-DPLL$(\phi[x = 0])$ **then akzeptiere end if**
return Meta-DPLL$(\phi[x = 1])$;

Implementieren für DLIS, DLCS, MOM, Böhm, Jeroslaw-Wang, Kürzeste Klausel.

PPZ

Mi,15.06.: Janitschke; Fr,17.06.: Schink, Buntrock

Algorithmus PPZ (Paturi, Pudlák, Zane, 1997)
Eingabe: KNF-Formel ϕ mit $n = \mathrm{Vars}(\phi)$

$\theta := \emptyset$;
Wähle zufällige Permutation $\pi \in S_n$;
for $i := 1, \dots, n$ **do**
    if $\{x_{\pi(i)}\} \in \phi[\theta]$ **then**
        $\theta := \theta \cup \{x_{\pi(i)} = 1\}$;
    else if $\{\neg x_{\pi(i)}\} \in \phi[\theta]$ **then**
        $\theta := \theta \cup \{x_{\pi(i)} = 0\}$;
    else
        Wähle zufällig $a \in \{0, 1\}$;
        $\theta := \theta \cup \{x_{\pi(i)} = a\}$;
return θ;

CDCL

Mi,22.06.: Gruhl, Stadler; Fr,24.06.: FREI

Algorithmus CDCL
Eingabe: Klauselmenge ϕ
Implikationsgraph $G_I = (\emptyset, \emptyset)$, $d = 0$, $\theta = \emptyset$;
if UnitPropagation(ϕ, θ, G_I, d) == CONFLICT **then return** Unerfüllbar;
else
  while $\phi[\theta] \neq \emptyset$ **do**
    $d++$;
    Belege nächste Variable v mit einem Wert a und aktualisiere θ;
    Füge Knoten $(v = a)$ zu G_I hinzu und setze $\mathrm{mark}(v = a) = d$;
    while UnitPropagation(ϕ, θ, G_I, d) == CONFLICT **do**
      Bestimme Schnitt in G_I und Konfliktklausel $K = (\ell_1 \vee \cdots \vee \ell_k)$;
      Seien u_i die Knoten, die ℓ_i falsifizieren, $1 \leq i \leq k$;
      $b = \max\{\mathrm{mark}(u_i) \mid 1 \leq i \leq k\}$;
      if $b == 0$ **then return** Unerfüllbar;
      else
        Entferne alle u aus V und θ mit $\mathrm{mark}(u) \geq b$;
        Setze $\phi := \phi \cup \{K\}$, $d--$;
    $d++$;
  return Erfüllbar;

Algorithmus UnitPropagation
Eingabe: Klauselmenge ϕ, Belegung θ, Implikationsgraph G, Level d
while es gibt noch Einheitsklauseln in $\phi[\theta]$ **do**
  Sei $K = (\ell)$ die erste Einheitsklausel in $\phi[\theta]$;
  Ist $\ell = \neg x$, dann setze $a := 0$, sonst $a := 1$;
  $\theta := \theta \cup \{x = a\}$, *modified* = true;
  Sei $K = (\ell \vee \ell_1 \vee \cdots \vee \ell_k)$ die Klausel in ϕ (ohne angewendete Belegung).
  Füge Knoten $x = a$ zu G hinzu und setze $\mathrm{mark}(x = a) = d$;
  Füge Kanten $\big(u, (x = a)\big)$ zu G hinzu, wobei u ℓ_i falsifiziert, $1 \leq i \leq k$.
if *modified* **and** $\phi[\theta] \equiv 0$ **then return** CONFLICT **end if**

Lokale Suche

Mi,29.06.: Reinders; Fr,01.07.: Wiebking+1

Algorithmus LocalSearch
Eingabe: Klauselmenge ϕ, Belegung θ, Radius p
Ausgabe: 1, falls es ein $\theta' \models \phi$ gibt mit $d(\theta, \theta') \leq p$
if $\theta \models \phi$ **then return** θ; **end if**
if $p = 0$ **then return** 0; **end if**
Sei $K = (\ell_1 \vee \ell_2 \vee \ell_3)$ in ϕ eine Klausel mit $\theta \not\models K$;
for $i = 1, \ldots, 3$ **do**

if LocalSearch$(\phi, \theta[\ell_i = 1], p-1)$ **then return** 1; **end if**
return 0;

Algorithmus LS-SAT
Eingabe: Klauselmenge ϕ
if not LocalSearch$(\varphi, \{x_1 = 1, \dots, x_n = 1\}, \lceil \frac{|\mathrm{Vars}(\phi)|}{2} \rceil)$ **then**
  return LocalSearch$(\varphi, \{x_1 = 0, \dots, x_n = 0\}, \lceil \frac{|\mathrm{Vars}(\phi)|}{2} \rceil)$

Random Walk

Mi,06.07.: Doering, Stanke; Fr,08.07.: FREI

Algorithmus RandomWalk
Eingabe: Klauselmenge ϕ
for $i = 1, \dots, t$ **do**
  Wähle zufällige Anfangsbelegung θ;
  for $j = 1, \dots, |\mathrm{Vars}(\phi)|$ **do**
    if $\theta \models \phi$ **then return** "erfüllbar"; **end if**
    Wähle eine Klausel $K = (\ell_1 \vee \ell_2 \vee \ell_3)$ mit $\theta \not\models K$;
    Wähle zufällig $k \in \{1, 2, 3\}$;
    $\theta := \theta[\ell_k = 1]$;
return "unerfüllbar";

GSAT

Mi,13.07.: FREI; Fr,15.07.: FREI

Algorithmus GSAT
Eingabe: Klauselmenge ϕ, Integer maxFlips, Integer maxTries
for $i = 1, \dots,$maxTries **do**
  Sei θ eine zufällige Startbelegung
  for $j = 1, \dots,$maxFlips **do**
    if $\theta \models \phi$ **then return** θ; **end if**
    Wähle zufällige Variable $x \in \min_{x \in \mathrm{Vars}(\phi)} \mathrm{score}_\theta(x)$
    $\theta \leftarrow \theta[x]$;
return Keine erfüllende Belegung gefunden;

WalkSAT

Mi,13.07.: FREI; Fr,15.07.: FREI

Algorithmus WalkSAT

Eingabe: Klauselmenge ϕ, Integer maxFlips, Integer maxTries
for $i = 1, \dots,$maxTries **do**
  Sei θ eine zufällige Startbelegung
  for $j = 1, \dots,$maxFlips **do**
    if $\theta \models \phi$ **then return** θ; **end if**
    Wähle unter den Klauseln K mit $\theta \not\models K$ eine zufällige aus.
    if $\exists x \in \mathrm{Vars}(K)$, sodass es keine Klausel K' gibt mit $\theta \models K'$ und $\theta[x] \not\models K'$ **then**
      $\theta \leftarrow \theta[x]$;
    else
      Entscheide zufällig:
      p**:** wähle unter Gleichverteilung eine Variable x in K;
      $1 - p$**:** Wähle zufällige Variable $x \in \min_{x \in \mathrm{Vars}(\phi)} |\mathrm{score}_\theta(x)|$;
      $\theta \leftarrow \theta[x]$;
return Keine erfüllende Belegung gefunden;

Divide and Conquer

?

Algorithmus DivideConquer-SAT
Eingabe: ϕ in 3KNF, $\mathrm{Vars}(\phi) = \{x_1, \dots, x_n\}$ mit n gerade, reelle Zahl $0 < \delta < 1$
Sei $V_1 = \{x_1, \dots, x_{\frac{n}{2}}\}$, und $V_2 = \{x_{\frac{n}{2}+1}, \dots, x_n\}$.
$K_1 := \{K \in \phi \mid \mathrm{Vars}(K) \subseteq V_1\}$, $K_2 := \{K \in \phi \mid \mathrm{Vars}(K) \subseteq V_2\}$
$K_3 := \{K = (\ell_1 \vee \ell_2 \vee \ell_3) \in \phi \mid \mathrm{Vars}(\{\ell_1, \ell_2\}) \subseteq V_1, \mathrm{Vars}(\{\ell_3\}) \subseteq V_2\}$.
$K_4 := \{K = (\ell_1 \vee \ell_2 \vee \ell_3) \in \phi \mid \mathrm{Vars}(\{\ell_1, \ell_2\}) \subseteq V_2, \mathrm{Vars}(\{\ell_3\}) \subseteq V_1\}$.
$A := \{\theta \mid \theta \models K_1\}$, $B := \{\theta \mid \theta \models K_2\}$.
for $\theta \in A$ **do**
  Wende auf θ Unit Propagation an.
  if $\theta \not\models K_3$ **then** $A := A \setminus \{\theta\}$. **end if**
for $\theta \in B$ **do**
  Wende auf θ Unit Propagation an.
  if $\theta \not\models K_4$ **then** $B := B \setminus \{\theta\}$. **end if**
FindConsistentPair1(ϕ, A, B, δ)
FindConsistentPair1(ϕ, B, A, δ)
FindConsistentPair2(ϕ, A, B, δ)

Algorithmus FindConsistentPair1
Eingabe: ϕ in 3KNF, Belegungsmengen A, B, reelle Zahl $0 < \delta \leq 1$
Ausgabe: Konsistentes Paar in (A, B) sofern vorhanden
for $\theta \in A$ mit $n - |\mathrm{Vars}(\theta)| \leq (1 - \delta) \cdot \frac{n}{2}$ **do**
  for jede vollständige Erweiterung von θ **do**
    if es gibt ein passendes $\theta' \in B$ **then return** (θ, θ') **end if**

Algorithmus FindConsistentPair2
Eingabe: ϕ in 3KNF, Belegungsmengen A, B, reelle Zahl $0 < \delta < 1$

Ausgabe: Erfüllbar gdw. ein konsistentes Paar in (A, B) gibt

$A', B' \leftarrow \emptyset$.
for $\theta \in A$ mit $n - |\mathrm{Vars}(\theta)| > (1-\delta) \cdot \frac{n}{2}$ **do**
  for alle $V' \subseteq V_1$ mit $|V'| \leq \frac{\delta n}{2}$ **do**
    Sei θ' die Belegung in der alle Variablen aus $V_1 \setminus V'$ gelöscht werden.
    $A' \leftarrow A' \cup \{\theta'\}$.
for $\theta \in B$ mit $n - |\mathrm{Vars}(\theta)| > (1-\delta) \cdot \frac{n}{2}$ **do**
  for alle $V' \subseteq V_2$ mit $|V'| \leq \frac{\delta n}{2}$ **do**
    Sei θ' die Belegung in der alle Variablen aus $V_2 \setminus V'$ gelöscht werden.
    $B' \leftarrow B' \cup \{\theta'\}$.
if $A' \cap B' \neq \emptyset$ **then return** erfüllbar **end if**

Enumeration

Algorithmus EnumKrom
Eingabe: Krom-Formel ϕ, Partielle Belegung θ

if $\phi[\theta] \equiv 1$ und $\mathrm{Vars}(\theta) = \mathrm{Vars}(\phi)$ **then print** θ. **end if**
if $\phi[\theta]$ ist erfüllbar und $\mathrm{Vars}(\theta) \neq \mathrm{Vars}(\phi)$ **then**
  Wähle Variable $x \in \mathrm{Vars}(\phi[\theta])$.
  EnumKrom$(\phi, \theta \cup \{x = 0\})$.
  EnumKrom$(\phi, \theta \cup \{x = 1\})$.