Toteutustekniikkaa

[Niemelä, Simons, 1999] `smodels` ja [Syrjänen, 1999] `lparse`

`http://www.tcs.hut.fi/pub/smodels/`

- Käytetään kompaktia karakterisointia (täydet joukot)
  ☞ hakuavaruus: not-literaalit
- Bottom-up haku & hakuavaruuden karsinta:
  (i) Oletukset tutkittavasta mallista tehdään yksi kerrallaan.
  (ii) Ennen seuraavan oletuksen tekoa lasketaan approksimaatio kaikille stabiileille malleille, jotka toteuttavat annetut oletukset ja tutkitaan, onko mahdollisesti saavutettu konflikti.

☞ Davis-Putnam -menetelmän tyylinen tekniikka

- Algoritmi säilyttää approksimaatiota stabiileista malleista literaalijoukkona (osittaismallina) FS:
  (i) Jos $A \in FS$, A kuuluu tutkittaviin stabiileihin malleihin.
  (ii) Jos $\mathrm{not}\ A \in FS$, A ei kuulu tutkittaviin stabiileihin malleihin ($\mathrm{not}\ A$ kuuluu täyteen joukkoon).
- Approksimointi perustuu stabiilien mallien ala- ja ylärajaan:
  $\mathrm{LB}(P, FS)$ on literaalijoukko, jonka jokainen FS:n toteuttava ohjelman P stabiili malli toteuttaa.
  $\mathrm{UB}(P, FS)$ on atomilauseiden joukko, joka sisältää jokaisen ohjelman P stabiilin mallin, joka toteuttaa FS:n.
- Approksimaatio **expand**(P, FS) on pienin joukko FS', joka sisältää FS:n ja on suljettu seuraavien sääntöjen suhteen:
  (i) jos literaali $L \in \mathrm{LB}(P, FS')$, $L \in FS'$ ja
  (ii) jos atomi $A \notin \mathrm{UB}(P, FS')$, $\mathrm{not}\ A \in FS'$.

Approksimaatioiden toteutus

- Tavoite: ala- ja yläraja laskettavissa lineaarisessa ajassa.
- Vähintään yhtä tarkat rajat kuin WF-mallilla.
- Hakutilanne (osittaismalli) hyödynnettävä tehokkaasti.
- Oletuksilla FS vielä käytettävissä olevat säännöt:
  $P_{FS} = \{H \leftarrow B_1 \wedge \cdots \wedge B_m \wedge \mathrm{not}\ B_{m+1} \wedge \cdots \wedge \mathrm{not}\ B_n \in P \mid \{\mathrm{not}\ B_1, \ldots, \mathrm{not}\ B_m, B_{m+1}, \ldots, B_n\} \cap FS = \emptyset\}$
- $\overline{L}$ on literaalin L:n komplementti:
  $\overline{\mathrm{not}\ A} = A$ ja $\overline{A} = \mathrm{not}\ A$.

$\mathrm{LB}(P, FS)$: pienin joukko FS', joka sisältää joukon FS ja on suljettu seuraavien sääntöjen suhteen:

S1: Jos $H \leftarrow L_1 \wedge \cdots \wedge L_n \in P_{FS'}$ ja $\{L_1, \ldots, L_n\} \subseteq FS'$, $H \in FS'$.

S2: Jos atomi H ei ole päänä ohjelmassa $P_{FS'}$, $\mathrm{not}\ H \in FS'$.

S3: Jos $H \in FS'$ on päänä vain yhdessä säännössä $H \leftarrow L_1 \wedge \cdots \wedge L_n \in P_{FS'}$, $\{L_1, \ldots, L_n\} \subseteq FS'$.

S4: Jos $\mathrm{not}\ H \in FS'$, $H \leftarrow L_1 \wedge \cdots \wedge L_n \in P_{FS'}$ ja $\{L_1, \ldots, L_{i-i}, L_{i+1}, \ldots, L_n\} \subseteq FS'$, $\overline{L_i} \in FS'$.

S5: Jos jollekin atomille H sekä $H \in FS'$ että $\mathrm{not}\ H \in FS'$, niin kaikki literaalit kuuluvat joukkoon FS'.

$\mathrm{UB}(P, FS)$: pienin malli ohjelmaklausuulijoukolle, joka saadaan poistamalla not-literaalit ohjelmasta P_{FS}.

Esimerkki 1. Tarkastellaan seuraavaa ohjelmaa:

$$
\begin{array}{lll}
P_1: & p \leftarrow \text{not } q & \text{(R1)} \\
& r \leftarrow \text{not } p & \text{(R2)} \\
& q \leftarrow p \wedge \text{not } r \wedge \text{not } s & \text{(R3)} \\
& s \leftarrow r \wedge \text{not } t & \text{(R4)}
\end{array}
$$

Alarajaan $\mathrm{LB}(P_1, \{\text{not } q\})$ saadaan not q:n lisäksi seuraavia literaaleja:

p	S1, sääntö R1
not r	S2, sääntö R2 ei enää käytettävissä
not s	S2, sääntö R4 ei enää käytettävissä
s	S4, sääntö R3

Koska saatiin konflikti s:n suhteen, laskenta voidaan keskeyttää: $\mathrm{LB}(P_1, \{\text{not } q\})$ sisältää kaikki literaalit.

Esimerkki 2. Tarkastellaan ohjelmaa

$$
\begin{array}{ll}
P_2: & p \leftarrow \text{not } q \\
& q \leftarrow \text{not } p \\
& r \leftarrow \text{not } p \\
& s \leftarrow \text{not } r \\
& t \leftarrow \text{not } s
\end{array}
$$

expand$(P_2, \{q\}) = \{q, \text{not } p, r, \text{not } s, t\}$

expand$(P_2, \{\text{not } q\}) = \{\text{not } q, p, \text{not } r, s, \text{not } t\}$

- Ala- ja ylärajoille voidaan antaa lineaarisen ajan toteutukset (Dowling-Gallier -tyyppiset tietorakenteet).
- Tarkkuus vähintään sama kuin WF-mallilla:
  Jos oletuksia ei ole, **expand**$(P, \{\})$ antaa ohjelman P WF-mallin.
- Hakutilanne (oletukset FS) hyödynnetään dynaamisesti.
  Esim. ohjelmalle P
  $p \leftarrow \text{not } q$
  $q \leftarrow \text{not } p$
  $p \leftarrow q$
  expand$(P, \{q\})$ palauttaa $\{q, p, \text{not } q, \text{not } p\}$.
  Huom! Ohjelman $P \cup \{q\}$ WF-malli on $\{q, p\}$
  (konflikti ei tule esille).

Algoritmi

function smodels(P, FS, φ)
$FS' :=$ **expand**(P, FS);
if conflict(P, FS') returns true **then** return false
else if $\mathrm{NA}(P)$ is covered by FS' **then** return **test**(P, FS', φ)
else
  $\chi :=$ **choose**(P, FS');
  if smodels$(P, FS' \cup \{\chi\}, \varphi)$ returns true **then**
    return true
  else return **smodels**$(P, FS' \cup \{\overline{\chi}\}, \varphi)$
  end if
end if

- Funktiota **test** käytetään löytyneen mallin testaukseen (φ on testiparametri).
- Funktio **conflict** tarkistaa, sisältääkö saatu approksimaatio konfliktin ($\{A, \text{not } A\} \subseteq FS$).
- Funktio **choose** toteuttaa hakuheuristiikan.
- **smodels** suorittaa *fokusoitua mallinetsintää*:
  smodels(P, FS, φ) palauttaa arvon true joss on olemassa P:n stabiili malli $M = \text{Dcl}(P, FS')$, joka toteuttaa FS:n ja **test**(P, FS', φ) palauttaa arvon true.

Esimerkki.

a) Tutkitaan, kuuluuko t P:n kaikkiin stabiileihin malleihin etsimällä vastamalli, johon t ei kuulu. Aloitetaan $FS = \{\text{not } t\}$ ja **test** palauttaa aina arvon true.

$p \leftarrow \text{not } q$	$\text{not } t$
$q \leftarrow \text{not } p$	s
$r \leftarrow p, \text{not } q$	$\text{not } r$
$r \leftarrow q, \text{not } p$	$p \mid \text{not } p$
$s \leftarrow \text{not } r$	$\text{not } q \mid q$
$t \leftarrow \text{not } s$	$r \mid r$
	$\times \mid \times$

Vastaus: kyllä

b) Etsitään ohjelman P

$p \leftarrow \text{not } q \wedge r$
$q \leftarrow \text{not } p$
$r \leftarrow \text{not } s$
$s \leftarrow \text{not } p$

stabiilit mallit: (aluksi $FS = \{\}$)

p	$\text{not } p$
$\text{not } s$	q
r	s
$\text{not } q$	$\text{not } r$

Löydettiin kaksi stabiilia mallia:
$\text{Dcl}(P, \{p, \text{not } s, r, \text{not } q\}) = \{p, r\}$
$\text{Dcl}(P, \{\text{not } p, q, s, \text{not } r\}) = \{q, s\}$

Heuristiikka

- lookahead-tekniikka:
  jokaiselle $A \in \text{NA}(P)$, jota FS ei peitä, lasketaan
  $FS_1 =$ **expand**$(P, FS \cup \{A\})$ ja
  $FS_2 =$ **expand**$(P, FS \cup \{\text{not } A\})$
- Saadaan tarkempi alaraja:
  Jos FS_1 sisältää konfliktin, lisätään $\text{not } A$ joukkoon FS ja jos FS_2 sisältää konfliktin, lisätään A joukkoon FS.
- FS_1 and FS_2 antavat arvion jäljellä olevasta hakuavaruudesta.
- Heuristiikka: valitaan atomi, jolla on pienin arvioitu jäljellä oleva hakuavaruus.

Arviointia

- Logiikkaohjelmat ja stabiilit mallit ilmaisuvoimainen rajoiteohjelmointiparadigma.
  (Esim. Boolen rajoitteet/lauselogiikka erikoistapaus).
- Hakuavaruuden rakenne poikkeaa lauselogiikan toteutuvuusongelmasta:
  (i) Vain not-literaalit lisäävät hakuavaruutta.
  (ii) Säännöt suunnattuja: Kerrostuneisuus rajaa hakuavaruutta.
  ☞ Enemmän vapautta tietämyksen esittämiseen
- Toteutustekniikat parantuneet ratkaisevasti.
  Esimerkiksi lineaarinen tilavaativuus.