RESOLUTSIOONI PREDIKAATLOOGIKA JAOKS

- Nagu lauseloogikagi juures on resolutsioon valem kehtestatavuskontrolli meetod, mis töötab ainult teatud kujul valemitega.
- Lausearvutuse juures oli selliseks kujuks konjunktiivne kuju, siin on selleks universaalne konjunktiivne kuju, st valem $\forall x_1 \ldots \forall x_n A$, kus A on literaalide disjunktsioonide konjunktsioon. Selline kuju on loomulikult üles kirjutatav klauslite hulgana (jättes üldsuskvantorid ja konjunktsioonid ilmutamata).
- Valemi viimine universaalsele KNKle on toimub tema viimisega preneksele KNKle, millele järgneb skolemiseerimine.
- Esimene teisendus viib valemiga temaga loogiliselt ekvivalentseks valemiks, teine kehtestatavuse mõttes ekvivalentseks valemiks.

RESOLUTSIOONIARVUTUS

- Resolutsioonimeetodi aluseks on järgmise kahe reegliga antud tuletussüsteem ehk arvutus, kus tuletatavateks objektideks on klauslid
- Resolutsioonireegel:

$$\frac{\Gamma \to A, \Delta \quad \Gamma', A' \to \Delta'}{(\Gamma, \Gamma' \to \Delta, \Delta')\sigma}$$

  kus $\sigma = \mathrm{mgu}(A, A')$ ja me eeldame, et eeldusklauslid on lahku standardiseeritud (st ümber nimetatud nii, et ei oma ühiseid muutujaid)
- Faktoriseerimisreeglid:

$$\frac{\Gamma, A', A \to \Delta}{(\Gamma, A \to \Delta)\sigma} \quad \frac{\Gamma \to A, A', \Delta}{(\Gamma \to A, \Delta)\sigma}$$

  kus $\sigma = \mathrm{mgu}(A, A')$

- Resolutsioonimeetodi idee on püüda etteantud klauslitest tuletada tühja klauslit $\rightarrow$ (mis esitab vastuolu). Kui see sellist tuletust ei ole, peaks etteantud klauslihulk olema kehtestatav.
- Resolutsiooniarvutus (s.o äsjadefineeritud tuletussüsteem) on korrektne ja täielik:
- **Teoreem** Klauslite hulk on kehtestatav parajasti siis, kui temast ei saa tuletada tühja klauslit.

- Faktoriseerimisreeglid on täielikkuse jaoks vajalikud: Vaatleme nt klauslihulka

$$\to p(x, a), p(a, x)$$
$$p(y, a), p(a, y) \to$$

Pelgalt resolutsioonireegliga saame juurde tuletada vaid

$$p(a, x) \to p(a, x)$$
$$p(y, a) \to p(y, a)$$
$$p(a, a) \to p(a, a)$$

(mis kõik on tautoloogilised), kuid faktoriseerimisreeglid annavad

$$\to p(a, a)$$
$$p(a, a) \to$$

milliste resolvendiks on tühi klausel.

- Näide resolutsioonituletusest:

  Soovime näidata valemi $\forall x \exists y p(x,y) \wedge \exists x \forall y \neg p(x,y)$ vastuolulisust.

  See valem esitub klauslihulgana

$$\to p(x, f(x))$$

$$p(c, x) \to$$

  Tühi klausel on tuletatav resolutsioonireegli ühekordse rakendamisega.

BAASALGORITM

- Põhiline resolutsioonialgoritm: “tasemete küllastamine”:
- Olgu antud hulk S klauseid, toimi järgmiselt:
- 1. Olgu $S_0 := S$, $n := 0$.

  2. Kui S_n sisaldab tühja klauslit, lõpeta vastusega “vastuoluline”.

  3. Olgu S_{n+1} hulga S_n kõigi resolventide hulk.

  4. Kui $S_{n+1} = S_n$, lõpeta vastusega “kehtestatav”.

  5. Vastasel korral omista $n := n + 1$ ning pöördu tagasi sammu 2 juurde.
- On selge, et nii tasemete küllastamise algoritm kui ka iga teine õiglane resolutsioonialgoritm (st algoritm, mis ei ignoreeri lõpmatult kaua ühtki paari tekitatud klausleid) säilitab resolutsiooniarvutuse korrektsuse.

  Selleks aga, et saavutada mõistlikku efektiivsust, tuleb baasalgoritmi optimiseerida.

OPTIMISATSIOONID: KUSTUTUSSTRATEEGIAD

- Kustutusstrateegiad elimineerivad klauslit, millest ei saa olla kasu.
- Ütleme, et klausel C *katab* C' (tähistus $C \geq C'$), kui C on kujul $\Gamma \to \Delta$ ja C' on kujul $\Gamma', \Gamma\sigma \to \Delta\sigma, \Delta'$.
- On selge, et kaetud klauslitest pole kasu, need võib eemaldada: Kui S on hulk klausleid ning S' on selle alamhulk, kusjuures iga klausel hulgast S on kaetud mingi klausliga hulgast S', siis juhul kui hulgast S saab k sammuga tuletada tühja klausli, saab selle k sammuga tuletada ka hulgast S'.
- Kaetuse tuvastamine on raske!

- *Tautoloogiliseks klausliks* nimetame klauslit kujul $\Gamma, A \rightarrow A, \Gamma'$.
- Tautoloogiad on samuti kasutud: Kui C on tautoloogiline, siis tema resolvent iga teise klausliga C' on kas samuti tautoloogiline või temaga kaetud.

- Klausel C on klauslihulgas S *puhas*, kui ta sisaldab puhast literaali, st on kujul $\Gamma \rightarrow A, \Delta$ (või vastavalt $\Gamma, A \rightarrow \Delta$), kus A ei unifitseeru ühegi aatomiga ühegi S klausli vasakus pooles (paremas pooles).
- Puhtad klauslid on kasutud: puhta klausli resolvent suvalise teise klausliga jääb puhtaks.

- *Ühikklausel* on klausel kujul $\to A$ või $A \to$ (kus A on muidugi aatom).

  Ühikresolutsiooniks nimetame resolutsiooniarvutuse varianti, kus resolutsioonireeglile on pandud kitsendus, et vähemalt üks eeldustest peab olema ühikklausel.

- Ühikresolutsioon pole täielik. Nt klauslihulgast

$$\to p, q$$

$$p \to q$$

$$p, q \to$$

$$q \to p$$

  saab tavalise resolutsiooniga tuletada tühja klausli, aga ühikresolutsiooniga mitte (sest pole ühikklausleid).

- Täielik ei ole ka mitte algoritm, kus eelistus on antud ühikklauslitele, s.o võimalusel rakendatakse resolutsioonireeglit alati ühikklauslile: Kui lisame eeltoodud klauslihulgale veel klauslid

$$\to r(a)$$

$$r(x) \to r(f(x))$$

  saame alati resolutsiooni rakendada uuele ühiklauslile, aga nii et jõua me kunagi tühja klauslini.

- *Horni klausliks* nimetame klauslit, mille paremal poolel on ülimalt üks aatom.

- Ühikresolutsioon on täielik Horni klauslite jaoks. (Seda asjaolu kasutab Prologi resolutsioonialgoritm!)

KODUÜLESANNE

- Näita resolutsiooniarvutuses, et valemitest

$$symm := \forall x \forall y (p(x,y) \supset p(y,x))$$
$$trans := \forall x \forall y \forall z (p(x,y) \wedge p(y,z) \supset p(x,z))$$
$$ser := \forall x \exists y p(x,y)$$

järeldub valem

$$refl := \forall x p(x,x)$$

LAHENDUS

- Näitame, et valem $\mathit{symm} \wedge \mathit{trans} \wedge \mathit{ser} \wedge \neg \mathit{refl}$ on vastuoluline.
- Klauselkujul esitub see valem nii:

$$p(x, y) \to p(y, x)$$
$$p(x, y), p(y, z) \to p(x, z)$$
$$\to p(x, f(x))$$
$$p(c, c) \to$$

- Võimalik tuletus resolutsiooniarvutuses:

$$
\begin{array}{lll}
1. & \to p(x, f(x)) & \text{input} \\
2. & p(x,y) \to p(y,x) & \text{input} \\
3. & \to p(f(x), x) & \text{from} 1, 2 \\
4. & p(x,y), p(y,z) \to p(x,z) & \text{input} \\
5. & p(f(x), z) \to p(x,z) & \text{from} 1, 4 \\
6. & \to p(x,x) & \text{from} 3, 6 \\
7. & p(c,c) \to & \text{input} \\
8. & \to & \text{from} 6, 7
\end{array}
$$