Mašina za zaključivanje

Aleksandar Milosavljević
Vladan Mihajlović

Mašina za zaključivanje

- Mašina za zaključivanje obezbeđuje izvršavanje “upita” nad skupom činjenica korišćenjem produkcionih pravila koja se mogu predstaviti na jedan od sledećih načina:
  - AKO *premisa* ONDA *zaključak*
  - IF *premisa* THEN *zaključak*
  - IZVEDI *zaključak* IZ *premisa*
  - INFER *zaključak* FROM *premisa*

Format zaključka

- Zaključak može da sadrži jedan predikat koji poseduje proizvoljan broj promenljivih i konstanti
  - Dva-u-nizu(el, i, j, Dole)
- Predikat se u Lisp-u predstavlja listom gde je prvi element naziv predikata, a ostali elementi argumenti
  - (Dva-u-nizu ?el ?i ?j Dole)
- Promenljive se predstavljaju simbolima koji u nazivu imaju prvo karakter ‘?’.

Format premise

- Premisa poseduje proizvoljan broj predikata koji su povezani logičkim veznikom I (AND).
  - premisa ⇔ P1 AND P2 AND ... AND Pn
- Predikati koji čine premisu mogu da sadrže promenljive, konstante i funkcije.
  - On(el, i, j) AND On(el, dec(i), j)
- Predstavljena u Lisp-u, pređašnja premisa ima sledeći oblik:
  - (AND (On ?el ?i ?j) (On ?el (=dec ?i) ?j))

- NAPOMENA: Zbog korišćenja lančanja u nazad nije dozvoljeno da se predikat iz zaključka javlja u premisi!

Funkcije u predikatima premise

- Funkcije koje se javljaju u predikatima poseduju u nazivu kao prvi karakter ‘=‘.
- Moraju biti definisane u kodu:
  - (defun =dec (x) (1- x))
- Promenljive koje su argumenti funkcija moraju biti prethodno “vezane” nekim drugim predikatom. Npr.:
  - ((On ?el **?i** ?j) (On ?el **(=dec ?i)** ?j))
  - Ovo pravilo važi i kod formiranja upita

Format pravila

- Iz prethodne analize se vidi da svako pravilo ima jedan predikat u zaključku, i jedan ili više predikata u premisi koji su povezani sa logičkim veznikom AND.
- Format pravila je sledeći:

  (IF (AND (On ?el ?i ?j) (On ?el (=dec ?i) ?j))
  THEN (Dva-u-nizu ?el ?i ?j Dole))
  ⇔
  IF On(el, i, j) AND On(el, dec(i), j)
  THEN Dva-u-nizu(el, i, j, Dole)

  (IF (King ?i ?j) THEN (There-is-King))
  ⇔
  IF King(i, j) THEN There-is-King()
- Baza pravila se predstavlja kao lista pojedinačnih pravila:
  - (R1 R2 ... Rn)

Primer 1

- Ukoliko su dati sledeći iskazi:
  1. Saša voli sve vrste hrane.
  2. Jabuke su hrana.
  3. Piletina je hrana.
  4. Hrana je sve ono sto neko jede i ne otruje se.
  5. Srđan jede kikiriki i još je živ.
  6. Ceca jede sve sto Srđan jede.
- Dokazati da Saša voli piletinu i kikiriki.
- Odrediti šta sve Saša voli.
- Odrediti ko sve voli piletinu i kikiriki.

Primer 1
Prevođenje iskaza u pravila i činjenice

- Na osnovu iskaza 1, 4 i 6 identifikovana su sledeća pravila:
  - IF Hrana(x) THEN Voli(Saša, x)
  - IF Jede(x, y) AND Ne-otruje-se(x, y) THEN Hrana(y)
  - IF Jede(Srđan, x) THEN Jede(Ceca, x)
- Na osnovu iskaza 2, 3 i 5 identifikovana su sledeće činjenice:
  - Hrana(Jabuka)
  - Hrana(Piletina)
  - Jede(Srđan, Kikiriki)
  - Ne-otruje-se(Srđan, Kikiriki)

Primer 1

Prevođenje u format koji koristi mašina za zaključivanje

- Definicija baze pravila *T1-RULES*:

```
(setq *T1-RULES* '(
  (IF (Hrana ?x) THEN (Voli Sasa ?x))
  (IF (AND (Jede ?y ?x) (Ne-otruje-se ?y ?x))
   THEN (Hrana ?x))
  (IF (Jede Srdjan ?x) THEN (Jede Ceca ?x))
  ))
```

- Definicija baze činjenica *T1-FACTS*:

```
(setq *T1-FACTS* '(
  (Hrana Jabuka)
  (Hrana Piletina)
  (Jede Srdjan Kikiriki)
  (Ne-otruje-se Srdjan Kikiriki)
  ))
```

Primer 1
Definisanje upita

- Format upita je identičan kao i format premise pravila:
  - jedan ili više predikata koji mogu da sadrže promenljive, konstante i funkcije.
- *Dokazati da Saša voli piletinu i kikiriki.*
  - (AND (Voli Sasa Piletina) (Voli Sasa Kikiriki))
- *Odrediti šta sve Saša voli.*
  - (Voli Sasa ?x)
- *Odrediti ko sve voli piletinu i kikiriki.*
  - (AND (Voli ?x Piletina) (Voli ?x Kikiriki))

Interfejsne funkcije

- Postoje 3 interfejsne funkcije mašine za zaključivanje:
  - (prepare-knowledge lr lf maxdepth)
    - Poziva se inicijalno sa parametrima **lr** (baza pravila), **lf** (baza činjenica) i maxdepth (maksimalna dubina lančanja pravila).
    - Služi za inicijalizaciju pravila i činjenica na osnovu kojih će se izvršavati upiti.
  - (infer q)
    - Glavna funkcija koja vraća rezultate u vidu liste lista smena za promenljive koje se javljaju u zadatom upitu **q**.
    - Ukoliko nema smena koje zadovoljavaju upit funkcija vraća **NIL**.
    - Ukoliko upit nema promenljivih, tj. ukoliko je reč o tvrđenju koje treba proveriti, funkcija vraća **(NIL)** ukoliko je tvrđenje dokazano.
  - (count-results q)
    - Za razliku od funkcije INFER, ova funkcija vraća broj lista smena koje zadovoljavaju upit.

Format liste smena

- Lista smena predstavlja asocijativnu listu gde se kao ključevi koriste promenljive, a kao vrednost konstanta kojom je promenljiva zamenjena.
- Primer:

```
((?x Pera) (?y 25) (?z "Perić"))
```

- Primer za listu listi smena koju vraća funkcija INFER:

```
(((?x Pera) (?y 25) (?z "Perić"))
 ((?x Laza) (?y 22) (?z "Lazić"))
 ((?x Mika) (?y 30) (?z "Mikić")))
```

Rezultati za primer 1

- (prepare-knowledge *T1-RULES* *T1-FACTS* 10)

- (infer '(AND (Voli Sasa Piletina) (Voli Sasa Kikiriki)))
  - -> (NIL)
- (infer '(Voli Sasa ?x))
  - -> (((?X JABUKA)) ((?X PILETINA)) ((?X KIKIRIKI)))
- (infer '(AND (Voli ?x Piletina) (Voli ?x Kikiriki)))
  - -> (((?X SASA)))

- (count-results '(AND (Voli Sasa Piletina) (Voli Sasa Kikiriki)))
  - -> 1
- (count-results '(Voli Sasa ?x))
  - -> 3
- (count-results '(AND (Voli ?x Piletina) (Voli ?x Kikiriki)))
  - -> 1

Predefinisani predikati

- Za razliku od običnih predikata, čije se važenje proverava nad bazom činjenica, kod predefinisanih predikati imamo direktno sračunavanje istinitosne vrednosti na osnovu zadatih argumenata.
- Tipični primeri predefinisanih predikata su:
  - Jednako(x,y), Različito(x,y), Veće(x,y), Manje(x,y), ...
- Predefinisani predikati se definišu kao LISP funkcije koje vraćaju TAČNO (t) ili NETAČNO (nil).
- Da bi mašina za zaključivanje znala da je neki predikat predefinisan, njegov naziv mora počinjati znakom uzvika (!).
  - Primeri: !eq, !ne, !gt, !lt

Predefinisani predikati

- Primeri predefinisanih predikata:

```
(defun !eq (a b)
   (equal a b))
(defun !ne (a b)
   (not (equal a b)))
```

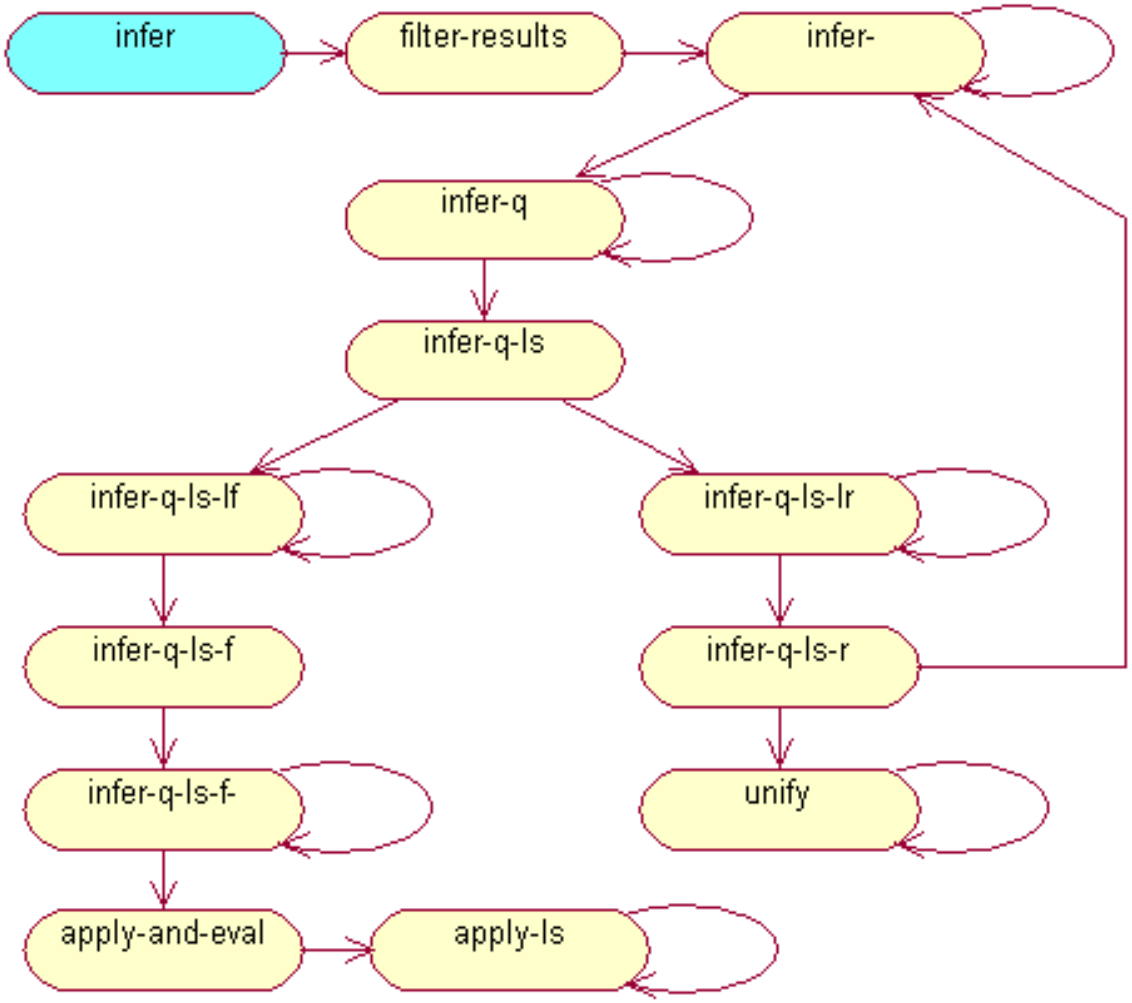
- Primer upita uz korišćenje predefinisanih predikata:

```
(infer '(AND (Voli 'Sasa ?x) (Voli 'Sasa ?y)
             (Voli 'Sasa ?z) (!ne ?x ?y)
             (!ne ?y ?z) (!ne ?x ?z)))
```

  - ->
```
(((?X 'JABUKA)   (?Y 'PILETINA) (?Z 'KIKIRIKI))
 ((?X 'JABUKA)   (?Y 'KIKIRIKI) (?Z 'PILETINA))
 ((?X 'PILETINA) (?Y 'JABUKA)   (?Z 'KIKIRIKI))
 ((?X 'PILETINA) (?Y 'KIKIRIKI) (?Z 'JABUKA))
 ((?X 'KIKIRIKI) (?Y 'JABUKA)   (?Z 'PILETINA))
 ((?X 'KIKIRIKI) (?Y 'PILETINA) (?Z 'JABUKA)))
```

- Pošto se predefinisani predikati direktno “izračunavaju” potrebno je da sve promenljive koje koriste prethodno bude povezane (*bind*-ovane)
  - Zbog toga su u prethodnom upitu predefinisani predikati smešteni na kraju.

Pregled funkcija



Pregled funkcija