Logiikka tietotekniikassa: perusteet

Luennot: tiistai klo 12-14 sali TT1

Luennoitsija: TkT Tomi Janhunen
(Tomi.Janhunen@hut.fi, 451 3255)

Laskuharjoitukset:

1. ryhmä: tiistai, klo 14-15, sali TT1

2. ryhmä: torstai, klo 9-10, sali TT1

alkaa 15. syyskuuta (tänään)

... pitää: TY Tommi Syrjänen (tssyrjan@saturn.hut.fi)

1. tentti: maanantai 7. joulukuuta, klo 9-12, salit GL

Vastaanotto: syksyllä tiistaisin klo 15-16 huone TB335

Tiedottaminen: ilmoitustaulu (sijoitus auki), kurssin kotisivu

`http://saturn.hut.fi/html/Tik-79.144.html`

ja otaxin uutisryhmä `opinnot.tik.logiikka`.

Kirjallisuus:

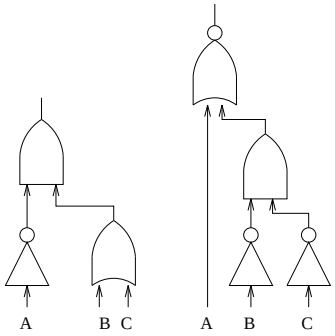
- A. Nerode ja R. A. Shore, *Logic for Applications*, Springer-Verlag, 1997. Kolme ensimmäistä lukua. (luennoilla esitetyssä laajuudessa)
- Opetusmonisteissa toimitetaan laskuharjoitusten mallivastaukset ja luentokalvot

Kolme pakollista kotitehtävää!

Kurssin sisältö

- lauselogiikka
  - syntaksi ja semantiikka
  - todistusteoria
  - semanttiset/analyyttiset taulut
  - normaalimuodot
  - resoluutiosääntö
- predikaattilogiikka (lauselogiikan yleistys)
  - lisänä induktioperiaate
- päättelyn automatisointi
  - teoreemantodistin (OTTER)
  - logiikkaohjelmointi

Esimerkki: kombinatoriset piirit



Lauselogiikan lauseina: $\neg A \wedge (B \vee C)$ ja $\neg(A \vee (\neg B \wedge \neg C))$

Esimerkki: ohjelmien ehtolausekkeet

Olkoon `myptr` tyyppiä "`char *`" C-kielessä.

```
/* TAPA 1 */

if(myptr != NULL && myptr[0] == '/') doit(myptr);
if(!(myptr == NULL || myptr[0] == '.')) dothat(myptr);

/* TAPA 2 */

if(myptr != NULL) {
  if(myptr[0] == '/') doit(myptr);
  if(myptr[0] != '.') dothat(myptr);
}
```


Esimerkki: ohjelmankehitys

Tiedetään: jos X1 $<$ 0 tai X2 $<$ 0 tai ... tai Xn $<$ 0, niin Z $<$ 0

Vertaa:

```
if (Z < 0)
    do_0()
else {
    if (X1 < 0)
        do_1();
    if (X2 < 0)
        do_2();
    ...
    if (Xn < 0)
        do_n();
}
```

```
if (Z < 0)
    do_0();
```


Esimerkki: ohjelmien oikeellisuuden toteaminen

- **Induktio** tietorakenteiden suhteen

  ```
  duplicate([],[]).
  duplicate([E|R1],[E|[E|R2]]) :- duplicate(R1,R2).
  ```

  Voidaan osoittaa *rakenteisella induktiolla*, että proseduuri `duplicate` käsittelee oikein mielivaltaisen pitkiä listoja.

- **Induktio** silmukan suorituskertojen suhteen

  ```
  while(n<10) n = n+1;
  ```

  Voidaan osoittaa, että n ≤ 10 on silmukan *invariantti* (ominaisuus joka säilyy silmukkaa suoritettaessa).

Esimerkki: relaatiotietokannat

- relaatioiden R_1 ja R_2 unioni

  $$\forall \overline{x}(R_1(\overline{x}) \vee R_2(\overline{x}) \rightarrow P(\overline{x}))$$

- relaatioiden R_1 ja R_2 leikkaus

  $$\forall \overline{x}(R_1(\overline{x}) \wedge R_2(\overline{x}) \rightarrow P(\overline{x}))$$

- projektio $XYZ \mapsto XZ$

  $$\forall x \forall y \forall z(R(x,y,z) \rightarrow P(x,z))$$

- relaatioiden R_1 ja R_2 natural join (alla y:n suhteen)

  $$\forall x \forall y \forall z(R_1(x,y) \wedge R_2(y,z) \rightarrow R(x,y,z))$$

Esimerkki: deduktiiviset tietokannat

sisarus(a, b)
sisarus(c, e)
vanhempi(a, c)
vanhempi(d, e)

$\forall x \forall y(\text{sisarus}(x, y) \rightarrow \text{sisarus}(y, x))$
$\forall x \forall y \forall z(\text{sisarus}(x, y) \wedge \text{sisarus}(y, z) \rightarrow \text{sisarus}(x, z))$
$\forall x \forall y(\text{vanhempi}(x, y) \rightarrow \neg\text{vanhempi}(y, x))$
$\forall x \forall y \forall z \forall v(\text{vanhempi}(x, y) \wedge \text{vanhempi}(z, v) \wedge \text{sisarus}(x, z) \rightarrow$
serkku$(y, v))$

Esimerkki: logiikkaohjelmointi

```
append([],L,L).
append([A|T],L,[A|S]) :- append(T,L,S).

?- append([1,2,3],[4,5,6],X).

X = [1,2,3,4,5,6]

?- append([1,X,3],Y,[1,2,3,4]).

X = 2, Y=[4]
```


Lauselogiikan syntaksi

Lauselogiikan *aakkosto*:

- atomiset lauseet: $A, A_1, A_2, \ldots, B, B_1, \ldots, C, \ldots$
- negaatiosymboli: $\neg$ (ei)
- konjunktiosymboli: $\wedge$ (ja)
- disjunktiosymboli: $\vee$ (tai)
- implikaatiosymboli: $\rightarrow$ (jos...niin)
- ekvivalenssisymboli $\leftrightarrow$ (jos ja vain jos)
- sulut: ()

Symbolit $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$ ovat *konnektiiveja*.

Lauselogiikan *lauseenmuodostussäännöt*:

1. atomiset lauseet ovat *lauseita*
2. jos P ja Q ovat lauseita, niin myös
   $(\neg P)$
   $(P \vee Q)$
   $(P \wedge Q)$
   $(P \rightarrow Q)$
   ja $(P \leftrightarrow Q)$
   ovat *lauseita*.
3. vain edellä olevien sääntöjen perusteella muodostetut merkkijonot ovat *lauseita*.

Esimerkkejä lauseista ovat:

$$B$$
$$(B \to (\neg A))$$
$$((\neg(C \vee D)) \to (A \vee B))$$

A = "Australiassa on kevät",
B = "Belgiassa on kevät",
C = "Kanadassa on talvi",
D = "Uudessa Seelannissa on talvi".

Huomio 1. implikaatio $\to$ *ei* tarkoita syy-seuraus-suhdetta. Esim. olkoon

A = "Suomi sijaitsee Afrikassa",
B = "Ruotsi sijaitsee Aasiassa".

Lause $(A \to B)$ on tosi.

Huomio 2. Disjunktio $(A \vee B)$ on tosi, jos A tai B tai sekä A että B (OR eikä XOR).

Esimerkki. Muotoillaan seuraava luonnollisen kielen lause lauselogiikan lauseena.

Jos tänään sataa, niin otan mukaan sateenvarjon tai kastun.

Valitaan atomiset lauseet

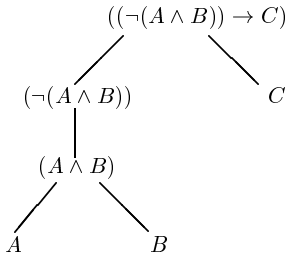
A = "Tänään sataa",
B = "Otan mukaan sateenvarjon",
C = "Kastun".

Saadaan: jos A, niin B tai C.

Tunnistetaan konnektiivit: $(A \to (B \vee C))$

Jokaisella lauselogiikan lauseella on yksikäsitteinen *jäsennyspuu*.

Esimerkki. Lauseen $((\neg(A \wedge B)) \to C)$ jäsennyspuu on seuraava:



Lauseiden ominaisuuksia voidaan todistaa induktiolla yli lauseen (tai sitä vastaavan jäsennyspuun) rakenteen.

Määritelmä. Lauseen *alilauseet*:

Atomisen lauseen A ainoa alilause on A.

Lauseen $(\neg\alpha)$ alilauseita ovat α:n alilauseet ja $(\neg\alpha)$.

Lauseen $(\alpha \wedge \beta)$ alilauseita ovat α:n ja β:n alilauseet ja $(\alpha \wedge \beta)$.

Lauseiden $(\alpha \vee \beta)$, $(\alpha \to \beta)$ ja $(\alpha \leftrightarrow \beta)$ alilauseet määritellään samaan tapaan kuin $(\alpha \wedge \beta)$:n.

Esimerkki. Lauseen $(A \to (B \vee C))$ alilauseet ovat $A, B, C, (B \vee C)$ ja $(A \to (B \vee C))$.

Esimerkki: rakenteinen induktio

Väite. Lauseessa ϕ on alilauseita korkeintaan niin monta kuin siinä on atomisia lauseita ja konnektiiveja.

Todistetaan väite rakenteisella induktiolla alilauseiden lukumäärän suhteen. Merkitään lauseen ϕ alilauseiden lukumäärää $\#\phi$:llä, atomisten lauseiden lukumäärää $A\phi$:llä ja konnektiivien lukumäärää $K\phi$:llä.

Näillä merkinnöillä väite saadaan muotoon $\#\phi \leq A\phi + K\phi$.

Perustapaus: ϕ on atominen lause A.

Koska $\#\phi = 1$ määritelmän mukaan, $A\phi = 1$ ja $K\phi = 0$, väittämä pitää tässä tapauksessa paikkansa.

Induktioaskel: ϕ on muotoa $(\neg\alpha)$.

Määritelmän mukaisesti: $\#(\neg\alpha) = 1 + \#\alpha$.

Induktiohypoteesin perusteella: $\#\alpha \leq A\alpha + K\alpha$.

Täten
$$\begin{aligned} \#(\neg\alpha) &\leq 1 + A\alpha + K\alpha \\ &= A\alpha + (1 + K\alpha) \\ &= A\alpha + (1 + K\alpha) \\ &= A(\neg\alpha) + K(\neg\alpha). \end{aligned}$$

Näin väittämä tuli todistetuksi muotoa $(\neg\alpha)$ oleville lauseille.

Binääriset konnektiivit $*$ tulee käsitellä samaan tapaan.
Lause ϕ on tällöin muotoa $(\alpha * \beta)$ ja induktiohypoteesia joudutaan soveltamaan sekä α:n että β:n suhteen.

Sulkeiden käytöstä

- uloimmat sulkeet tapana jättää pois: $A \to B$ eikä $(A \to B)$
- ketjudisjunktiot/konjunktiot: $A \vee B \vee C \vee D$ eikä $A \vee (B \vee (C \vee D))$
- konnektiivien presedenssi
  1. $\neg$ on vahvin konnektiiveista
  2. $\vee$, $\wedge$ ovat heikompia kuin $\neg$, mutta vahvempia kuin $\to$, $\leftrightarrow$
  3. $\to$, $\leftrightarrow$ ovat heikoimmat konnektiivit

  Esimerkiksi $\neg A \to B$ eikä $(\neg A) \to B$,
  $A \wedge B \to B \vee C$ eikä $(A \wedge B) \to (B \vee C)$,
  mutta $(A \to B) \vee (B \leftrightarrow C)$.