DD1350 Logik för dataloger

Fö 4 – Predikatlogik

Kort repetition

- Satslogik
- Naturlig deduktion är ett **sunt** och **fullständigt** bevissystem för satslogik
- Avgörbarhet
- Så vad saknas?

Egenskaper

- Satslogiken är dålig på att uttrycka **egenskaper** hos objekt.
- T.ex. ”Anna är student och Eva är student”
- I satslogik blir detta ”p /\ q”, men detta verkar inte fånga hela betydelsen…
- Bättre:

  $Student(anna) \wedge Student(eva)$ eller

  $P(a) \wedge P(e)$

vilket i varje fall visar att det finns 2 objekt med samma egenskap.

Relationer

- Satslogiken är dålig på att uttrycka **relationer** mellan objekt.
- T.ex. ”Om en person känner en annan person, så känner den andra personen också den första.”
- Om vi vet ”Anna känner Eva” så vill vi kunna dra slutsatsen ”Eva känner Anna”.
- $\forall x \forall y(Känner(x,y) \rightarrow Känner(y,x))$

Fler brister hos satslogik

- Satslogiken kan **inte** uttrycka påståenden om ett **oändligt** antal element, t. ex.

  ”Alla primtal större än 2 är udda”

  $\forall x\ (\ Primtal(x) \wedge x>2 \rightarrow Udda(x))$

- Detta påstående kan inte uttryckas i satslogik.

Predikatlogik

- Predikatlogik utökar det satslogiska språket med:
  - variabler
  - konstanter
  - funktionssymboler
  - relationssymboler
  - kvantifierare

Variabler och konstanter

- **Variabler** kan bindas till termer (definieras snart), som representerar objekt
  - Som variabler använder vi typiskt *u*, *v*, *w*, *x*, *y*, *z*

- **Konstanter** representerar objekt
  - typiskt *a*, *b*, *c*, *anna*, *eva*, *sverige*, *stefan_löfven*

Termer

- **Funktionssymboler** är t.ex. f, g, +, •
  - Varje symbol har en **aritet** (antal argument)
- En **term** definieras rekursivt:
  - en variabel är en term
  - en konstant är en term
  - om f är en funktionssymbol av aritet n, och $t_1, \ldots, t_n$ är termer, så är $f\ (t_1, \ldots, t_n)$ en term
  - inget annat är en term

Formler

- **Relationssymboler**, t.ex P, Q, Känner, Primtal, Udda
- Formler definieras rekursivt:
  - om P är en relationssymbol med aritet n, och $t_1, \ldots, t_n$ är termer, så är $P(t_1, \ldots, t_n)$ en formel
  - om φ och ψ är formler så är följande formler: $\neg \varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$, $\varphi \rightarrow \psi$, $\forall x\, \varphi$, $\exists x\, \varphi$
  - inget annat är en formel

Exempel

Kvantifierare kan förekomma inne i formler:

”Alla människor har en mamma”

$\forall x\, (\mathit{Människa}(x) \rightarrow \exists y\, (\mathit{Mamma}(y,x)))$

”Alla svenska medborgare har en kung”

$\exists y\, \forall x\, (\mathit{Svensk}(\mathrm{x}) \rightarrow \mathit{Kung}(y,x))$

”Ingen människa saknar en mamma”

$\neg \exists x\, (\mathit{Människa}(\mathrm{x}) \wedge \neg \exists y\, (\mathit{Mamma}(y,x)))$

Likhet

'=' är en speciell relationssymbol som vi antar alltid finns i logiken.

$x = y$ betecknar att x och y är samma objekt

T.ex. max ett objekt har egenskapen E:

$$\forall x\, \forall y\, (E(x) \wedge E(y) \rightarrow x = y)$$

Räckvidd

Vilka *kvantifierare* binder vilka *förekomster* av vilka variabler?

$$\forall \boldsymbol{x}\, \exists \boldsymbol{y}\, (\boldsymbol{P}(\boldsymbol{x},\boldsymbol{y}) \wedge \exists \boldsymbol{x}\, \boldsymbol{Q}(\boldsymbol{x},\boldsymbol{y}))$$

Räckvidden (eng: scope) för en kvantifierare i en formel $\ldots(\forall x\, \phi)\ldots$ eller $\ldots(\exists x\, \phi)\ldots$ omfattar alla förekomster av variabeln x i delformeln ϕ, med undantag av de som i sin tur tillhör en delformel $\forall x\, \psi$ eller $\exists x\, \psi$ av ϕ

Fria och bundna variabelförekomster

- En förekomst av en variabel x i en formel ϕ är **bunden** om den ligger inom räckvidden för någon kvantifierare, och är **fri** annars
- Exempel: i formeln $\exists y\,(P(x,y) \wedge \exists x\, Q(x,y))$ är första förekomsten av variabeln x fri, och den andra bunden
- En kvantifierare $\forall x$ eller $\exists x$ **binder** alla (fria) förekomster av variabeln x inom sin räckvidd
- Vi kan byta namn på bundna variabler men inte på fria!

Substitution

För en variabel x, en term t och en formel ϕ, betecknar $\phi[t \,/\, x]$ formeln som är resultatet av substitutionen av alla fria förekomster av x i ϕ med t

Exempel:

$$\exists y\,(P(x,y) \wedge \exists x\, Q(x,y))\,[x+1/x]$$

$$= \exists y\,(P(x+1,y) \wedge \exists x\, Q(x,y))$$

Variabelinfångande

- Problem: $\phi(x) \equiv \exists y\ (x < y)$
  - vad är då $\phi(y)$? $\exists y\ (x < y)[y/x]$?

    men $\exists y\ (y < y)$ har *inte* samma mening!
  - **variabelinfångand***e* (eng: variable capture)
  - när vi gör en substitution måste vi **undvika variabelinfångande!**
- Substitutionen $\phi[t / x]$ **undviker infångande** om **t är fri för x *i* ϕ**

Variabelinfångande

Termen t är ***fri* för variabeln x i formeln** ϕ om ingen fri förekomst av x i ϕ är inom räckvidden för någon kvantifierare $\forall y$ eller $\exists y$ för någon variabel y i t

Detta kan *alltid* åstadkommas - genom *omdöpning* i formeln, t.ex:

$\phi(x) \equiv \exists y\ (x < y)$ y kan döpas om till z

$\phi(x) \equiv \exists z\ (x < z)$ med samma mening

$\phi(y) \equiv \exists z\ (y < z)$

Substitution som undviker variabelinfångande

OBS: I kursen kommer vi låta

$$\phi[t / x]$$

beteckna resultatet av substitutionen som undviker variabelinfångande!

1. i formeln ϕ, döp om alla kvantifierare som fångar någon variabel i termen t vid någon fri förekomst av variabeln x i formeln ϕ
2. utför substitutionen som vanligt, dvs substituera alla fria förekomster av x i ϕ med t