- Lauseet α ja β ovat loogisesti ekvivalentteja, $\alpha \equiv \beta$, jos ne ovat tosia samoissa malleissa:

  $\alpha \equiv \beta$ joss $\alpha \models \beta$ ja $\beta \models \alpha$

- Lause on validi eli tautologia jos se on tosi kaikissa malleissa

  $P \vee \neg P$

- Jokainen validi lause on loogisesti ekvivalentti arvon **T** kanssa

- **Deduktioteoreema**: Mille tahansa lauseille α ja β,

  $\alpha \models \beta$ joss lause $(\alpha \Rightarrow \beta)$ on validi

- Mallintarkistusalgoritmin voi ajatella tarkistavan lauseen $TK \Rightarrow \alpha$ validisuuden





- Lause on toteutuva (satisfiable), jos on olemassa malli, jossa se on tosi
- Esim. $TK = R_1 \wedge \ldots \wedge R_5$ on toteutuva, koska on olemassa kolme sen *toteuttavaa* mallia
- Propositiologiikan toteutuvuusongelma oli ensimmäinen NP-täydelliseksi todistettu ongelma (Cook 1971)

- α on validi joss ¬α ei ole toteutuva
- Kääntäen: α on toteutuva joss ¬α ei ole validi
- Ristiriitaan perustuva todistus:

  $\alpha \models \beta$ joss lause $(\alpha \wedge \neg\beta)$ ei ole toteutuva

  $\alpha \models \beta \Leftrightarrow (\alpha \Rightarrow \beta)$ on validi $\Leftrightarrow \neg(\alpha \Rightarrow \beta)$ ei ole toteutuva

  $\Leftrightarrow \neg(\neg\alpha \vee \beta)$ ei ole toteutuva $\Leftrightarrow (\alpha \wedge \neg\beta)$ ei ole toteutuva

# 2.3 PÄÄTTELY PROPOSITIOLOGIIKASSA

- **Modus Ponens**

$$\frac{\alpha \Rightarrow \beta,\ \alpha}{\beta}$$

- **Konjunktin eliminointi**

$$\frac{\alpha \wedge \beta}{\alpha}$$

- Nämä kaksi sääntöä ovat eheitä aina, ei ole tarvetta tehdä mallintarkistusta, vaan sääntöjä voidaan soveltaa suoraan
- Seuraavista tunnetuista loogisista ekvivalensseista saadaan kustakin kaksi päättelysääntöä
- Sen sijaan esim. Modus Ponensia ei voi kääntää





# Loogisia ekvivalensseja

| | |
|---|---|
| $(\alpha \wedge \beta) \equiv (\beta \wedge \alpha)$ | **kommutatiivisuus** |
| $(\alpha \vee \beta) \equiv (\beta \vee \alpha)$ | **kommutatiivisuus** |
| $((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma))$ | **assosiatiivisuus** |
| $((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma))$ | **assosiatiivisuus** |
| $\neg(\neg\alpha) \equiv \alpha$ | **negaation poisto** |
| $(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$ | **kontrapositio** |
| $(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta)$ | **implikaation poisto** |
| $(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$ | **ekvivalenssin poisto** |
| $\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$ | **de Morgan** |
| $\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$ | **de Morgan** |
| $(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma))$ | **distributiivisuus** |
| $(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$ | **distributiivisuus** |

- Pyritään *todistamaan* ettei ruudussa [1,2] ole kuoppaa
- Ekvivalenssin poistolla säännöstä $R_2$ saadaan

$$(V_{1,1} \Rightarrow (K_{1,2} \vee K_{2,1})) \wedge ((K_{1,2} \vee K_{2,1}) \Rightarrow V_{1,1})$$

- Soveltamalla tähän konjunktin poistoa seuraa

$$((K_{1,2} \vee K_{2,1}) \Rightarrow V_{1,1})$$

- Kontrapositiolla

$$(\neg V_{1,1} \Rightarrow \neg(K_{1,2} \vee K_{2,1}))$$

- Soveltamalla tähän ja sääntöön $R_4$: $\neg V_{1,1}$ Modus Ponensia saadaan $\neg(K_{1,2} \vee K_{2,1})$
- Lopulta de Morganin säännöllä voidaan todeta

$$\neg K_{1,2} \wedge \neg K_{2,1}$$





## Tietämyskanta (2)

$R_1$: $\neg K_{1,1}$
$R_2$: $V_{1,1} \Leftrightarrow (K_{1,2} \vee K_{2,1})$
$R_3$: $V_{2,1} \Leftrightarrow (K_{1,1} \vee K_{2,2} \vee K_{3,1})$
$R_4$: $\neg V_{1,1}$
$R_5$: $V_{2,1}$
$R_6$: $(V_{1,1} \Rightarrow (K_{1,2} \vee K_{2,1})) \wedge ((K_{1,2} \vee K_{2,1}) \Rightarrow V_{1,1})$
$R_7$: $( (K_{1,2} \vee K_{2,1}) \Rightarrow V_{1,1} )$
$R_8$: $( \neg V_{1,1} \Rightarrow \neg(K_{1,2} \vee K_{2,1}) )$
$R_9$: $\neg(K_{1,2} \vee K_{2,1})$
$R_{10}$: $\neg K_{1,2} \wedge \neg K_{2,1}$

- Koska päättely propositiologiikassa on NP-täydellistä, niin pahimmassa tapauksessa todistuksen etsiminen on yhtä tehotonta kuin mallintarkastus
- Käytännössä kuitenkin irrelevanttien tietojen huomiotta jättäminen usein tekee todistamisen tehokkaaksi
- Esim. edellisen todistuksen maalisymboli $K_{1,2}$ esiintyy vain TK:n säännössä $R_2$, jossa mainitut muut symbolit $V_{1,1}$ ja $K_{2,1}$ puolestaan esiintyvät sen lisäksi vain säännössä $R_4$
- Näin ollen säänöissä $R_1$, $R_3$ ja $R_5$ mainittuja symboleja $V_{2,1}$, $K_{1,1}$, $K_{2,2}$ ja $K_{3,1}$ ei tarvitse huomioida
- Logiikan *monotonisuus*: loogisten seurausten määrä voi vain kasvaa lisääntyneen tiedon myötä





- Mille tahansa lauseille $\alpha$ ja $\beta$ pätee
  jos $TK \models \alpha$ niin $TK \wedge \beta \models \alpha$
- Eli lisätieto $\beta$ voi auttaa vetämään uusia johtopäätöksiä, mutta se ei tee jo tehtyjä päätelmiä $\alpha$ epäpäteviksi
- Esimerkiksi $\beta$ voisi olla tieto, että kuoppia on kahdeksan kappaletta
- Edelleenkin jo tekemämme päättelyt yksittäisten ruutujen kuoppaisuudesta pätevät
- Monotonisuuden perusteella päättelysääntöjä voidaan soveltaa aina kun ennakkoehdot täyttyvät TK:ssa
- Seuraukset pätevät riippumatta TK:n sisältämästä muusta informaatiosta

## Resoluutio

- Yhdistettynä täydelliseen hakualgoritmiin edellä esitetyt päättelysäännöt ovat täydellinen päättelyalgoritmi
- Kuitenkin yhdenkin päättelysäännön poistaminen tuhoaa algoritmin täydellisyyden

- Resoluutio on yksi ainoa päättelysääntö, joka yhdistettynä täydelliseen hakualgoritmiin antaa täydellisen päättelyalgoritmin

- Kahden literaalin lauseisiin rajoitettuna resoluutioaskel on

$$\frac{\ell_1 \vee \ell_2,\ \neg\ell_2 \vee \ell_3}{\ell_1 \vee \ell_3}$$





- Kun agentti siirtyy ensimmäisen kerran tutkimaan ruutua [1,2], niin se aistii löyhkän muttei viimaa:
  $R_{11}$: $\neg V_{1,2}$ ja
  $R_{12}$: $V_{1,2} \Leftrightarrow (K_{1,1} \vee K_{2,2} \vee K_{1,3})$
- Vastaavalla päättelyllä kuin edellä, voimme todeta ettei ruuduissa [2,2] ja [1,3] ole kuoppaa $R_{13}$: $\neg K_{2,2}$ ja $R_{14}$: $\neg K_{1,3}$
- Ekvivalenssin poisto sovellettuna sääntöön $R_3$ yhdessä Modus Ponensin ja säännön $R_5$ kanssa antaa $R_{15}$: $K_{1,1} \vee K_{2,2} \vee K_{3,1}$
- Resoluutioaskel sovellettuna sääntöihin $R_{13}$ ja $R_{15}$ antaa
  $R_{16}$: $K_{1,1} \vee K_{3,1}$
- Edelleen resoluutioaskel sovellettuna sääntöihin $R_1$ ja $R_{16}$ antaa
  $R_{17}$: $K_{3,1}$

## Tietämyskanta (3)

$R_1$: $\neg K_{1,1}$
$R_2$: $V_{1,1} \Leftrightarrow (K_{1,2} \vee K_{2,1})$
$R_3$: $V_{2,1} \Leftrightarrow (K_{1,1} \vee K_{2,2} \vee K_{3,1})$
$R_4$: $\neg V_{1,1}$
$R_5$: $V_{2,1}$
$R_6$: $(V_{1,1} \Rightarrow (K_{1,2} \vee K_{2,1})) \wedge ((K_{1,2} \vee K_{2,1}) \Rightarrow V_{1,1})$
$R_7$: $( (K_{1,2} \vee K_{2,1}) \Rightarrow V_{1,1} )$
$R_8$: $( \neg V_{1,1} \Rightarrow \neg(K_{1,2} \vee K_{2,1}) )$
$R_9$: $\neg(K_{1,2} \vee K_{2,1})$
$R_{10}$: $\neg K_{1,2} \wedge \neg K_{2,1}$
$R_{11}$: $\neg V_{1,2}$
$R_{12}$: $V_{1,2} \Leftrightarrow (K_{1,1} \vee K_{2,2} \vee K_{1,3})$
$R_{13}$: $\neg K_{2,2}$
$R_{14}$: $\neg K_{1,3}$
$R_{15}$: $K_{1,1} \vee K_{2,2} \vee K_{3,1}$
$R_{16}$: $K_{1,1} \vee K_{3,1}$
$R_{17}$: $K_{3,1}$





- Edellä käytettiin yksikköresoluutiota

$$\frac{\ell_1 \vee \ldots \vee \ell_k, \; m}{\ell_1 \vee \ldots \vee \ell_{i-1} \vee \ell_{i+1} \vee \ldots \vee \ell_k},$$

  missä $\ell_i$ ja $m$ ovat komplementaariset literaalit

- Yleisessä muodossa resoluutioaskel sallii mielivaltaisen määrän komplementaarisia literaaleja

- Resoluutioaskelen johtopäätöksestä tulee poistaa literaalien duplikaatit

$$\frac{A \vee B, \; A \vee \neg B}{A}$$

- Resoluutio on eheä päättelysääntö ja se on myös täydellinen
- Tosin resoluutiolla ei voi tiedosta, että $A$ on tosi saada johtopäätöstä $A \vee B$
- Sen sijaan resoluutiolla voidaan tarkistaa onko $A \vee B$ tosi
- Osoittaaksemme, että $TK \models \alpha$ osoitamme, että $TK \wedge \neg\alpha$ on toteutumaton
- Ensin $TK \wedge \neg\alpha$ muunnetaan *konjunktiiviseen normaalimuotoon* (CNF) ja resoluutioalgoritmia sovelletaan sen tekijöihin
- Lopulta joko ei ole uusia lisättäviä tekijöitä, jolloin $\alpha$ ei ole $TK$:n looginen seuraus, tai resoluutioaskel tuottaa tyhjän tekijän (= **E**), jolloin $TK \models \alpha$
- Resoluutioaskel sovellettuna lauseeseen $P \wedge \neg P$ tuottaa ristiriidan





- Muunnetaan sääntö $R_2$: $V_{1,1} \Leftrightarrow (K_{1,2} \vee K_{2,1})$ CNF-muotoon

- Poistetaan ensin eksvivalenssi:
  $$(V_{1,1} \Rightarrow (K_{1,2} \vee K_{2,1})) \wedge ((K_{1,2} \vee K_{2,1}) \Rightarrow V_{1,1})$$
- Implikaatiot poistetaan seuraavaksi:
  $$(\neg V_{1,1} \vee K_{1,2} \vee K_{2,1}) \wedge (\neg(K_{1,2} \vee K_{2,1}) \vee V_{1,1})$$
- Negaatiot pitää siirtää literaaleihin:
  $$(\neg V_{1,1} \vee K_{1,2} \vee K_{2,1}) \wedge ((\neg K_{1,2} \wedge \neg K_{2,1}) \vee V_{1,1})$$
- Distributiivisuudella saadaan sääntö lopulta CNF-muotoon:
  $$(\neg V_{1,1} \vee K_{1,2} \vee K_{2,1}) \wedge (\neg K_{1,2} \vee V_{1,1}) \wedge (\neg K_{2,1} \vee V_{1,1})$$

## Hornin lauseet

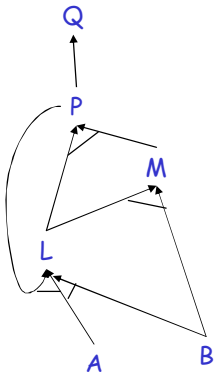
- Tietämyksen esittämiseen riittäävät usein implikaatiosäännöt, esim. $(Paikka_{1,1} \wedge Viima) \Rightarrow V_{1,1}$
- Säännön etujäsentä, joka muodostuu positiivisten literaalien konjunktiosta, kutsutaan lauseen rungoksi (body)
- Myös takajäsen on negatoimaton ja sitä nimitetään lauseen kärjeksi (head)
- Jos Hornin lauseella ei ole lainkaan runkoa, niin kyseessä on fakta
- Hornin lause $P_1 \wedge \ldots \wedge P_n \Rightarrow Q$ on loogisesti ekvivalentti disjunktion $(\neg P_1 \vee \ldots \vee \neg P_n \vee Q)$ kanssa
- Literaaleista siis korkeintaan yhden sallitaan olla positiivinen propositio





- Päättely Hornin lauseilla voi perustua joko *eteen- tai taaksepäin ketjutukseen*
- Molemmat päättelysuunnat ovat luontevia
- Loogisen seuraamisen tutkimisen aikavaativuus Hornin lauseilla on vain lineaarista tietämyskannan koon suhteen
- Eteenpäin ketjutus (*datalähtöinen päättely*) tutkii minkä kaikkien sääntöjen runko toteutuu tunnettujen faktojen perusteella ja lisää niiden kärjet uusiksi faktoiksi tietämyskantaan
- Kyseessä on siis toistuva Modus Ponens -säännön soveltaminen
- Prosessi jatkuu kunnes annettu kysely $q$ on lisätty faktojen joukkoon tai kunnes uusia päätelmiä ei enää voida tehdä

$P \Rightarrow Q$
$L \wedge M \Rightarrow P$
$B \wedge L \Rightarrow M$
$A \wedge P \Rightarrow L$
$A \wedge B \Rightarrow L$
$A$
$B$








- Eteenpäin ketjutus on sekä eheä että täydellinen päättelyalgoritmi
- Taaksepäin ketjutus on *tavoiteohjattua päättelyä* (goal-directed)
- Annetun kyselyn $q$ totuus pyritään toteamaan tarkastelemalla sellaisia sääntöjä, joiden kärki $q$ on
- Taaksepäin ketjuttaen tutkitaan sääntöjen runkojen konjunktien totuutta
- Taaksepäin ketjutus käsittelee vain relevantteja faktoja kun taas eteenpäin ketjutus tuottaa sokeasti kaikki

## 2.4 PROPOSITIOLOGIIKAN RIITTÄMÄTTÖMYYS

- Jo äärimmäisen yksinkertaisessa peliesimerkissämme propositiologiikan ilmaisuvoima osoittautuu riittämättömäksi
- Tietämyskannan alustamiseksi pelin säännöillä meidän on jokaiselle ruudulle [x,y] annettava sääntö viiman tuntemuksesta (sekä vastaava sääntö löyhkän aistimukselle)

$$V_{x,y} \Leftrightarrow (K_{x,y+1} \vee K_{x,y-1} \vee K_{x+1,y} \vee K_{x-1,y})$$

- Hirviöitä on ainakin yksi

$$H_{1,1} \vee H_{1,2} \vee \ldots \vee H_{4,3} \vee H_{4,4}$$

ja toisaalta korkeintaan yksi, joka voidaan ilmaista sääntöparein

$$\neg H_{1,1} \vee \neg H_{1,2}$$





- Kun TK:aan vielä lisätään säännöt $\neg K_{1,1}$ ja $\neg H_{1,1}$, niin pelkästään näihin yksinkertaisiin sääntöihin on tarvittu $2 + 2 \cdot 16 + 1 + 120 = 155$ sääntöä ja eri symboleita on käytössä 64 kappaletta
- Mallintarkastuksen tulisi käydä läpi $2^{64} \approx 1.8 \times 10^{19}$ mahdollista mallia
- Tehokkaammilla päättelyalgoritmeilla tätäkin esitystä toki voidaan käyttää
- Lisäksi TK ei vielä suinkaan sisällä pelin kaikkia sääntöjä (nuoli, kulta, seinät) eikä tietoa agentin mahdollisista toiminnoista (rintamasuunta, askelet eteenpäin), joten sitä ei voi käyttää toimintojen valintaan
- Raskautensa lisäksi propositiologiikka on epätyydyttävä ratkaisu

# 2.5 PREDIKAATTILOGIIKKA

Lause → Atomilause | (Lause Konnektiivi Lause)
| Kvanttori M-lista: Lause | ¬Lause
Atomilause → Predikaatti(T-lista) | Termi = Termi
T-lista → Termi | Termi, T-lista
Termi → Funktio(T-lista) | Vakio | Muuttuja
Konnektiivi → ∧ | ∨ | ⇨ | ⇔
Kvanttori → ∀ | ∃
M-lista → Muuttuja | Muuttuja, M-lista
Vakio → A | $X_1$ | Jussi | Meeri | …
Muuttuja → a | x | s | …
Predikaatti → Teekkari | On-väriltään | Äiti | …
Funktio → Isä | Vasen-jalka | …





- (1. kertaluvun) predikaattilogiikan maailmassa on *olioita*, joilla on *ominaisuuksia*, ja joiden välillä vallitsee *suhteita*
- Vakiosymbolit viittaavat olioihin,
  predikaattisymbolit suhteisiin eli n-paikkaisiin relaatioihin,
  Äiti(Jussi, Meeri),
  ja ominaisuudet ovat yksipaikkaisia relaatioita,
  Teekkari(Jussi)
- Funktio on totaalinen kuvaus, joka liittää argumenttiensa järjestettyyn kokoelmaan täsmälleen yhden tulosolion

- Kvanttorit antavat mahdollisuuden universaaliin ja eksistentiaaliseen kvantifiointiin

$$\forall y: \text{Teekkari}(y) \Rightarrow \text{Opiskelee}(y)$$

$$\exists x: \text{Isä}(\text{Jussi}) = x$$

- Semantiikan määräämiseksi syntaktiset merkinnät (vakio-, predikaatti- ja funktiosymbolit) on sidottava *tulkinnalla* (interpretation) (reaali)maailman vastineisiinsa
- Maailma yhdessä merkintöjen tulkinnan kanssa muodostavat predikaattilogiikan mallin





- Termi $f(t_1, \ldots, t_n)$ viittaa siihen olioon, joka saadaan ottamalla termien $t_1, \ldots, t_n$ viittaamat oliot $d_1, \ldots, d_n$ ja soveltamalla niihin sitä funktiota $F$, joka on $f$:n tulkinta
- Atomilause $P(t_1, \ldots, t_n)$ on tosi mikäli oliot, joihin termit $t_1, \ldots, t_n$ viittaavat ovat keskenään siinä relaatiossa, joka on predikaattisymbolin $P$ tulkinta
- Konnektiivein muodostettujen lauseiden totuus on kuten propositiologiikassa
- Lause $\exists x: \varphi(x)$ on tosi annetussa mallissa ja tulkinnassa, jos lauseesta $\varphi$ saadaan tosi sijoittamalla muuttujan $x$ tulkinnaksi jokin olio
- Lause $\forall x: \varphi(x)$ on tosi annetussa mallissa ja tulkinnassa, jos lause $\varphi$ on tosi sijoitettiinpa muuttujan $x$ tulkinnaksi mikä tahansa olio

- Koska ∀ on itse asiassa konjunktio yli maailman olioiden ja ∃ disjunktio, niin ne noudattavat de Morganin sääntöjä

$$\forall x: \neg\varphi(x) \Leftrightarrow \neg\exists x: \varphi(x)$$
$$\neg\forall x: \varphi(x) \Leftrightarrow \exists x: \neg\varphi(x)$$
$$\forall x: \varphi(x) \Leftrightarrow \neg\exists x: \neg\varphi(x)$$
$$\neg\forall x: \neg\varphi(x) \Leftrightarrow \exists x: \varphi(x)$$

- Yhtäsuuruus on erikoisasemassa oleva relaatio, jonka tulkintaa ei voi asettaa
- Termit $t_1$ ja $t_2$ ovat tässä relaatiossa vain jos niiden tulkinta on sama olio





## Sukulaisuussuhteita

$\forall n, l$: Äiti(l) = n ⇔ Nainen(n) ∧ Vanhempi(n, l).

$\forall v, m$: Aviomies(m, v) ⇔ Mies(m) ∧ Puoliso(m, v).

$\forall x$: Mies(x) ⇔ ¬Nainen(x).

$\forall v, l$: Vanhempi(v, l) ⇔ Lapsi(l, v).

$\forall i, l$: Isovanhempi(i, l) ⇔ ∃ v: Vanhempi(i, v) ∧ Vanhempi(v, l).

$\forall x, y$: Sisarus(x, y) ⇔ x ≠ y ∧ ∃ v: Vanhempi(v, x) ∧ Vanhempi(v, y).

## Peanon aksioomat

- Määrittelevät *luonnolliset luvut* ja yhteenlaskun yhden vakiosymbolin 0 ja seuraajafunktion S avulla

$$LL(0).$$
$$\forall\, n\colon LL(n) \Rightarrow LL(S(n)).$$

- Luonnolliset luvut siis ovat 0, S(0), S(S(0)), ...
- Seuraajafunktion ominaisuuksia

$$\forall\, n\colon 0 \neq S(n).$$
$$\forall\, m, n\colon m \neq n \Rightarrow S(m) \neq S(n).$$

- Yhteenlasku ja seuraajafunktio

$$\forall\, m\colon LL(m) \Rightarrow +(m, 0) = m.$$
$$\forall\, m, n\colon LL(m) \wedge LL(n) \Rightarrow +(S(m), n) = S(+(m, n))$$





## Joukot

- Joukkojen määrittelemiseksi käytämme
  - vakiosymbolia $\emptyset$, joka viittaa tyhjään joukkoon
  - yksipaikkaista predikaattia Set, joka on tosi joukoille
  - kaksipaikkaisia predikaatteja $x \in s$ ja $s_1 \subseteq s_2$
  - Binäärisiä funktioita ovat $s_1 \cup s_2$, $s_1 \cap s_2$ ja $\{ x | s \}$, joka viittaa siihen joukkoon, joka saadaan kun alkio $x$ lisätään joukkoon $s$
- Joukkoja ovat vain tyhjä joukko ja ne, jotka on muodostettu lisäämällä alkioita joukkoihin

$$\forall\, s\colon Set(s) \Leftrightarrow s = \emptyset \vee \exists\, x, s_2\colon Set(s_2) \wedge s = \{ x | s_2 \}.$$

- Tyhjällä joukolla ei ole alkioita

$$\neg\exists\, x, s\colon \{ x | s \} = \emptyset.$$

- Samaa alkiota ei voi lisätä joukkoon kahdesti
  $\forall x, s: x \in s \Leftrightarrow s = \{ x | s \}.$
- Joukossa on vain siihen lisätyt alkiot
  $\forall x, s: x \in s \Leftrightarrow [\exists y, s_2: (s = \{ y | s_2 \} \wedge (x = y \vee x \in s_2))].$
- Joukkojen sisältyvyys
  $\forall s_1, s_2: s_1 \subseteq s_2 \Leftrightarrow (\forall x: x \in s_1 \Rightarrow x \in s_2).$
- Joukkojen ekvivalenssi
  $\forall s_1, s_2: (s_1 = s_2) \Leftrightarrow (s_1 \subseteq s_2 \wedge s_2 \subseteq s_1).$

$\forall x, s_1, s_2: x \in (s_1 \cap s_2) \Leftrightarrow (x \in s_1 \wedge x \in s_2).$
$\forall x, s_1, s_2: x \in (s_1 \cup s_2) \Leftrightarrow (x \in s_1 \vee x \in s_2).$





## 2.6 PÄÄTTELY PREDIKAATTILOGIIKASSA

- Jos predikaattilogiikan lauseista eliminoidaan kvanttorit, niin voidaan siirtyä propositiologiikan lauseisiin ja käyttää tuttuja päättelysääntöjä
- Universaalikvantifioidun lauseen $\forall v: \alpha$ muuttuja $v$ voidaan korvata millä tahansa *perustermillä* (ground term), joka ei sisällä muuttujia
- Sijoitus eli *substituutio* $\theta$ on sidontalista, jossa kullekin muuttujalle annetaan sen korvaava perustermi
- Merkitään $\alpha(\theta)$ on lause, joka saadaan kun sijoitusta $\theta$ sovelletaan lauseeseen $\alpha$
- Universaalikvanttorin eliminoimiseksi voimme päätellä

$$\frac{\forall v: \alpha}{\alpha(\{v/g\})},$$

missä $v$ on mikä tahansa muuttuja ja $g$ mv. perustermi

- Eksistenssikvanttorin eliminoimisessa muuttuja v korvataan Skolem vakiolla k, joka ei ennestään esiinny tietämyskannassa

$$\frac{\exists\, v: a\ .}{a(\{v/k\})}$$

- Esimerkiksi voimme lauseesta ∃ x: Isä(Jussi) = x päätellä *instantiaation* Isä(Jussi) = $F_1$, missä $F_1$ on uusi vakio
- Eliminoimalla eksistenssikvanttorit korvaamalla muuttujat Skolem-vakioilla ja universaalikvanttorit kaikilla mahdollisilla instantiaatioillaan, muuttuu tietämyskanta oleellisesti propositionaaliseksi
- Muuttujattomat atomilauseet kuten Teekkari(Jussi) ja Äiti(Jussi, Meeri) on vain nähtävä propositiosymboleina
- Täten propositiologiikan päättelyä voidaan soveltaa predikaattilogiikan lauseisiin





- Jos tietämyskannassa kuitenkin käytetään funktiosymboleita, niin mahdollisten korvaavien perustermien lukumäärä on ääretön

Isä(Isä(Isä(Jussi)))

- Herbrandin kuuluisan tuloksen mukaan 1. kertaluvun teorian loogisilla seurauksilla on todistus "propositionalisoidussa" teoriassa, joka käsittelee vain äärellistä osaa siitä
- Täten sisäkkäisiä funktioita voidaan käsitellä syvyyden mukaan kasvavassa järjestyksessä ilman, että menetetään mahdollisuus johtaa kaikki loogiset seuraukset
- Päättely on siis täydellistä
- Analogia Turingin koneiden pysähtymisongelmaan kuitenkin osoittaa, että *ongelma on ratkeamaton*
- Tarkemmin: *ongelma on osittain ratkeava*, hahmoteltu menetelmä löytää todistuksen loogisille seurauksille

## Samaistus

- Päättely propositiologiikassa on ilmiselvästi turhan raskasta
- Muuttujasijoitusten aukikirjoittaminen vaikuttaa turhalta
- Jos meillä on sijoitus θ s.e. $p'_i(\theta) = p_i(\theta)$, kaikilla $i$, missä $p'_i$ ja $p_i$ ovat atomilauseita kuten myös $q$, niin voimme käyttää *yleistettyä Modus Ponensia*

$$\frac{p'_1, \ldots, p'_n, (p_1 \wedge \ldots \wedge p_n \Rightarrow q)}{q(\theta)}$$

- Esimerkiksi faktasta Teekkari(Jussi) sekä lauseista
  ∀ x: Ahkera(x) ja
  ∀ y: Teekkari(y) ∧ Ahkera(y) ⇨ DI_2009(y)
  voimme päätellä DI_2009(Jussi) sidonnan
  { y/Jussi, x/Jussi} perusteella





- Yleistetty Modus Ponens on eheä päättelysääntö
- Samaan tapaan kuin yMP voidaan "korottaa" propositiologiikasta predikaattilogiikkaan, voidaan myös eteen- ja taaksepäin ketjutus sekä resoluutio muokata

- Päättelyalgoritmeissa keskeinen käsite on lauseiden samaistus (unification)
- Samaistusalgoritmi Unify palauttaa syötteenä annetut lauseet samaistavan sijoituksen, jos sellainen on olemassa

  Unify(p, q) = θ, s.e. p(θ) = q(θ)

  muuten samaistus epäonnistuu (fail)

Unify( Tuntee(Jussi, x), Tuntee(Jussi, Jaana) ) = { x/Jaana }
Unify( Tuntee(Jussi, x), Tuntee(y, Pekka) )= { x/Pekka, y/Jussi }
Unify( Tuntee(Jussi, x), Tuntee(y, Äiti(y)) ) =
{ y/Jussi, x/Äiti(Jussi) }
Unify( Tuntee(Jussi, x), Tuntee(x, Elina) ) = fail

- Viimeinen samaistus epäonnistuu, koska muuttujaa x ei voida samanaikaisesti sitoa sekä Jussiin että Elinaan
- Koska muuttujat ovat universaalikvantifioituja, niin Tuntee(x, Elina) tarkoittaa, että kaikki tuntevat Elinan
- Sikäli samaistuksen pitäisi onnistua