Normalisering av meningar inför resolution

På samma sätt som i satslogiken är resolution i predikatlogiken en process vars syfte är att vederlägga att en klausulmängd är satisfierbar. Det förutsätter dock att godtyckliga meningar (!) kan skrivas om till en form motsvarande satslogikens CNF.

Omskrivningen är betydligt mer komplicerad och består av 8 olika steg (enligt beskrivningen som ges i det följande).

OBS! Omskrivningen funkar endast för meningar. Inga fria variabler får förekomma!

Steg 1: Eliminera alla $\rightarrow$ och $\leftrightarrow$

Det här görs på precis samma sätt som i satslogiken, dvs $\varphi \rightarrow \varphi'$ ersätts med $\neg\varphi \vee \varphi'$ och $\varphi \leftrightarrow \varphi'$ med $(\varphi \wedge \varphi') \vee (\neg\varphi \wedge \neg\varphi')$.

Steg 2: Flytta alla negationer framför atomära formler

Det görs som i satslogiken så länge inga kvantifierare är involverade. För att handskas med kvantifierare används ekvivalenserna

$$\neg(\forall x)(\varphi) \equiv (\exists x)(\neg\varphi) \quad \text{och}$$
$$\neg(\exists x)(\varphi) \equiv (\forall x)(\neg\varphi).$$

Steg 3: Ge varje variabel ett unikt namn

Byt namn på variabler som förekommer i samband med fler en en kvantifierare, så att varje kvantifierare får sitt unika variabelnamn. T ex

$$(\exists x)(P(x) \vee (\forall y)(Q(x) \wedge (\exists x)(R(x,y))))$$
$$\rightsquigarrow (\exists x_1)(P(x_1) \vee (\forall x_2)(Q(x_1) \wedge (\exists x_3)(R(x_3,x_2)))).$$

Steg 4: Skriv om formeln till prenex normalform

En formel är på prenex normalform (PNF) om den består av en prefix som endast innehåller kvantifierare och en efterföljande formel utan kvantifierare. T ex

$$(\exists x)(\forall y)(\forall x)(P(f(x,y)) \vee Q(x)).$$

(Fr.o.m. nu utelämnas parenteser om kvantifierare följer direkt på varandra.)

Efter steg 1–3 kan varje mening skrivas om till en mening på PNF genom att flytta alla kvantifierare till vänster utan att byta ordning på dem. T ex

$$(\exists x_1)(P(x_1) \vee (\forall x_2)(Q(x_1) \wedge (\exists x_3)(R(x_3,x_2))))$$
$$\rightsquigarrow (\exists x_1)(\forall x_2)(\exists x_3)(P(x_1) \vee (Q(x_1) \wedge R(x_3,x_2))).$$

Steg 5: Konvertera formelns efterled till CNF

$$(\exists x_1)(\forall x_2)(\exists x_3)\underline{(P(x_1) \vee (Q(x_1) \wedge R(x_3,x_2)))}$$

Det här går på samma sätt som när det gäller satslogiska formler.

Steg 1-5: Exempel

$$\neg(\forall x)(\neg P(x,c_0) \rightarrow ((\exists y)(P(f(x,y),c_1)) \wedge \neg(\exists y)(\neg P(y,c_0) \wedge P(f(x,y),c_0))))$$

$$\neg(\forall x)(P(x,c_0) \vee ((\exists y)(P(f(x,y),c_1)) \wedge \neg(\exists y)(\neg P(y,c_0) \wedge P(f(x,y),c_0))))$$

$$(\exists x)(\neg P(x,c_0) \wedge ((\forall y)(\neg P(f(x,y),c_1)) \vee (\exists y)(\neg P(y,c_0) \wedge P(f(x,y),c_0))))$$

$$(\exists x)(\neg P(x,c_0) \wedge ((\forall y)(\neg P(f(x,y),c_1)) \vee (\exists z)(\neg P(z,c_0) \wedge P(f(x,z),c_0))))$$

$$(\exists x)(\forall y)(\exists z)(\neg P(x,c_0) \wedge (\neg P(f(x,y),c_1) \vee (\neg P(z,c_0) \wedge P(f(x,z),c_0))))$$

$$(\exists x)(\forall y)(\exists z)(\neg P(x,c_0) \wedge ((\neg P(f(x,y),c_1) \vee \neg P(z,c_0)) \wedge (\neg P(f(x,y),c_1) \vee P(f(x,z),c_0))))$$

$$(\exists x)(\forall y)(\exists z)\underline{(\neg P(x,c_0)} \wedge \underline{(\neg P(f(x,y),c_1) \vee \neg P(z,c_0))} \wedge \underline{(\neg P(f(x,y),c_1) \vee P(f(x,z),c_0))})$$

Steg 6: Skolemisering (1)

Vi har nu en mening på formen

$$(K_1)\cdots(K_m)(\varphi_1 \wedge \cdots \wedge \varphi_n)$$

där $K_1, \ldots, K_m$ är kvantifierare och $\varphi_1, \ldots, \varphi_n$ är disjunktioner av litteraler. Kvantifierarna gör att $\varphi_1, \ldots, \varphi_n$ inte kan betraktas var för sig. Målet är att bryta upp kopplingen genom att flytta kvantifierarna in till $\varphi_1, \ldots, \varphi_n$.

Det kan dock endast göras med allkvantifieraren $\forall$ eftersom den distribuerar över $\wedge$ medan existenskvantifieraren $\exists$ inte gör det.

Det gäller alltså att först bli av med existenskvantifierarna bland $K_1, \ldots, K_m$. Processen kallas skolemisering efter logikern Thoralf Skolem.

Steg 6: Skolemisering (2)

Skolemisering skapar en mening som är satisfierbar om och endast om originalmeningen är det. OBS! Syftet är inte att skapa en ekvivalent formel.

Betrakta t ex $\varphi = (\forall x)(\forall y)(\exists z)(\forall z')(\varphi')$ och en tolkning J där $\varphi^J = 1$.

- Man ska alltså för alla $d_x, d_y \in \mathrm{dom}(J)$ kunna välja något $d_z \in \mathrm{dom}(J)$ som gör att $(\forall z')(\varphi')$:s värde blir 1 under J-värderingarna som tilldelar x, y, z värdena d_x, d_y, d_z.
- Värdet d_z beror på d_x och d_y (men inte på $d_{z'}$). Man kan alltså se det som en funktion choose_z som väljer rätt värde beroende på d_x, d_y, dvs $d_z = \mathrm{choose}_z(d_x, d_y)$.
- Vi kan därför utöka logiken med en ny (!) funktionssymbol $f_z^{(2)}$ och ersätta formeln med $\psi = (\forall x)(\forall y)(\forall z')(\varphi'')$ där φ'' fås genom att ersätta alla förekomster av z i φ' med $f_z(x, y)$.
- Om J utökas på så sätt att f_z tolkas som choose_z så gäller $\psi^J = 1$.

Exempel

$$\varphi = (\forall x)(\exists y)(\forall z)((x < y) \wedge ((z < y) \rightarrow z - 1 < x))$$

(där infixnotation används för att öka läsbarheten).

- Om $\mathrm{dom}(J) = \mathbb{N}$ och alla funktions- och relationssymboler tolkas som vanligt är $\varphi^J = 1$:

  För varje $d_x \in \mathbb{N}$ finns det ett $d_y \in \mathbb{N}$, nämligen $d_x + 1$, som uppfyller $d_x < d_y$ samtidigt som $d_z < d_y$ implicerar $d_z \leq d_x$, alltså $d_z - 1 < d_x$.

- Vi kan alltså lägga till funktionssymbolen $f_y^{(1)}$ och konvertera formeln till

  $$\psi = (\forall x)(\forall z)((x < f_y(x)) \wedge ((z < f_y(x)) \rightarrow z - 1 < x)).$$

  Om vi sedan definierar $f_y^J(d) = d + 1$ gäller igen $\psi^J = 1$.

Steg 6: Skolemisering (3)

Generellt introduceras för varje existenskvantifierare $(\exists x)$ en ny funktionssymbol $f_x^{(n)}$ där n är antalet allkvantifierare $(\forall x_1)\cdots(\forall x_n)$ till vänster om $(\exists x)$. (Om $n = 0$ blir f_x en konstant istället.) Sedan ersätts varje förekomst av x med $f_x(x_1, \ldots, x_n)$ (respektive f_x om $n = 0$).

Teorem Skolemiseringen av en mening som har förenklats enligt steg 1–5 är satisfierbar om och endast om originalmeningen är satisfierbar.

Steg 7: Skapa en mängd av klausuler

Formeln är nu på formen $(K_1)\cdots(K_m)(\varphi_1 \wedge \cdots \wedge \varphi_n)$ där $K_1, \ldots, K_m$ är allkvantifierare. Pga att $\forall$ distribuerar över $\wedge$ kan meningen skrivas om till

$$(K_1)\cdots(K_m)(\varphi_1) \wedge \cdots \wedge (K_1)\cdots(K_m)(\varphi_n)$$

vilket i sin tur kan skrivas som en mängd Φ av klausuler:

$$\Phi = \{\ (K_1)\cdots(K_m)(\varphi_1),$$
$$\vdots$$
$$(K_1)\cdots(K_m)(\varphi_n)\ \}.$$

Som en notationell förenkling nämns allkvantifierarna inte längre explicit, dvs Φ skrivs som $\{\varphi_1, \ldots, \varphi_n\}$.

OBS! Kom ihåg att variablerna inte är fria nu – kvantifierarna är fortfarande där fast de inte explicit skrivs ned.

Steg 8: Byt namn på variablerna än en gång

Till slut byts namnen på variablerna i var och en av $\varphi_1, \ldots, \varphi_n$ på så sätt att φ_i och φ_j inte längre innehåller några gemensamma variabelnamn, för $1 \leq i < j \leq n$.

Omskrivningen är klar nu!

Steg 6-8: Exempel

$(\exists x)(\forall y)(\exists z)(\neg P(x, c_0) \wedge (\neg P(f(x, y), c_1) \vee \neg P(z, c_0)) \wedge (\neg P(f(x, y), c_1) \vee P(f(x, z), c_0)))$

$(\forall y)(\neg P(f_x, c_0) \wedge (\neg P(f(f_x, y), c_1) \vee \neg P(f_z(y), c_0)) \wedge (\neg P(f(f_x, y), c_1) \vee P(f(f_x, f_z(y)), c_0)))$

$(\forall y)(\neg P(f_x, c_0)) \wedge (\forall y)(\neg P(f(f_x, y), c_1) \vee \neg P(f_z(y), c_0)) \wedge \ldots$

$\{\neg P(f_x, c_0),\ \neg P(f(f_x, y), c_1) \vee \neg P(f_z(y), c_0),\ \neg P(f(f_x, y), c_1) \vee P(f(f_x, f_z(y)), c_0)\}$

$\{\neg P(f_x, c_0),\ \neg P(f(f_x, y_2), c_1) \vee \neg P(f_z(y_2), c_0),\ \neg P(f(f_x, y_3), c_1) \vee P(f(f_x, f_z(y_3)), c_0)\}$