Wyprowadzanie negatywnych informacji

P — program w logice, A — formuła atomowa ($A \in B_P$)

$P \cup \{A\}$ jest zawsze spełnialny — modelem jest B_P

czyli $\neg A$ nie jest logiczną konsekwencją programu P

Wniosek
Z programów w logice można wyprowadzać wyłącznie pozytywne informacje.

metoda wnioskowania jest **poprawna** (ang. sound), jeśli dla dowolnej ustalonej formuły φ zachodzi:

$$P \vdash \varphi \quad \text{implikuje} \quad P \models \varphi$$

reguła monotoniczna: $P \vdash \varphi \Longrightarrow P \cup P' \vdash \varphi$

(jeśli istnieje SLD-dowód dla $P \cup \{G\}$, to istnieje SLD-dowód dla $P \cup P' \cup \{G\}$)

reguły niemonotoniczne: dodanie nowych aksjomatów może spowodować zmniejszenie zbioru uprzednio wyprowadzonych twierdzeń

metoda wnioskowania jest **słabo poprawna** (ang. weakly sound), jeśli dla dowolnej ustalonej formuły φ zachodzi:

$$P \vdash \varphi \quad \text{implikuje} \quad P \cup \{\varphi\} \text{ jest niesprzeczny}$$

Lemat
Jeśli dana metoda wnioskowania umożliwia wyprowadzanie negatywnych informacji, to nie jest ona poprawna.
Jeśli jest słabo poprawna, to nie jest monotoniczna.

Reguły wnioskowania

- reguła zamkniętości świata (CWA)
- reguła Herbranda
- reguła negacji przez porażkę (NF)

Reguła CWA (Closed World Assumption, R.Reiter, 1978)

A — ustalona formuła atomowa

CWA: jeśli A nie jest logiczną konsekwencją P, to wnioskuj $\neg A$

metareguła: $$\frac{P \not\models A}{\neg A}$$

$CWA(P) = \{\neg A : A$ jest ustalonym atomem takim, że nie istnieje SLD-dowód dla $P \cup \{\leftarrow A\}\}$

Lemat

$$\neg A \in CWA(P) \text{ wiw gdy } A \in B_P - M_P$$

inna charakterystyka reguły CWA:

M_P — ,,rzeczywista" interpretacja $$\frac{A \notin M_P}{\neg A}$$

CWA — najmocniejsza słabo poprawna reguła wnioskowania

zastosowanie: (relacyjne) bazy danych

ogólnie: problem czy formuła atomowa jest logiczną konsekwencją zbioru formuł jest nierozstrzygalny

programy w logice: nieskończona dziedzina (B_P),
klauzule nie tylko unarne

wniosek: w ogólności nie możemy stosować reguły CWA jeśli A nie jest logiczną konsekwencją P:

A nie należy do zbioru sukcesów P, mamy dwa przypadki:

— istnieje skończone SLD-drzewo (uzyskamy odpowiedź NIE, czyli wyprowadzimy $\neg A$)

— SLD-drzewo jest nieskończone (nie uzyskamy odpowiedzi)

Reguła NF (Negation as Failure, K.Clark, 1978)

Definicja
Zbiór skończonych SLD-porażek programu P:
zbiór $A \in B_P$, dla których istnieje skończone SLD-drzewo (dla $P \cup \{\leftarrow A\}$) nie zawierające gałęzi sukcesu

reguła NF:

jeśli A należy do zbioru skończonych SLD-porażek P
($\leftarrow A$ ponosi skończoną porażką),
to wnioskuj $\neg A$

$$\frac{\leftarrow A_1,\ldots,A_{i-1},\neg A_i,A_{i+1},\ldots,A_n \qquad \leftarrow A_i \text{ ponosi skończona porażke}}{\leftarrow A_1,\ldots,A_{i-1},A_{i+1},\ldots,A_n}$$

(skończona porażka — FF, ang. finite failure)

Wnioski:

— reguła NF jest słabo poprawna
— reguła NF jest niemonotoniczna
— rozstrzygalność
(czy skończone SLD-drzewo jest drzewem porażki)
— reguła NF jest słabsza od CWA
(ale w praktyce trudno coś więcej uzyskać)

Zbiór skończonych SLD-porażek

F_P^d – zbiór atomów z B_P, które ponoszą porażkę głębokości d

(i) $F_P^0 \stackrel{df}{=} \emptyset$

(ii) $A \in F_P^d,\ d \geq 1,$ jeśli
dla każdej klauzuli $B \leftarrow B_1, \ldots, B_n$
i każdego podstawienia θ takiego, że
– $A = B\theta$
– $B_1\theta, \ldots, B_n\theta$ — formuły ustalone
zachodzi: istnieje k, $1 \leq k \leq n$, takie że $B_k\theta \in F_P^{d-1}$

F_P — zbiór skończonych SLD-porażek programu P

$$F_P = \bigcup_{d \geq 1} F_P^d$$

SLD-wywód jest **sprawiedliwy** (ang. fair), jeśli jest skończony lub każdy atom będzie w końcu wybrany

SLD-drzewo jest sprawiedliwe, jeśli każda jego gałąź jest sprawiedliwym SLD-wywodem

funkcja wyboru jest sprawiedliwa, jeśli wszystkie SLD-drzewa powstałe przy jej użyciu są sprawiedliwe

Twierdzenie (charakterystyka reguły NF)

P — program, A — atom ustalony
Następujące warunki są równoważne:

(a) A należy do zbioru skończonych SLD-porażek P

(b) każde sprawiedliwe SLD-drzewo o korzeniu $\leftarrow A$ jest drzewem skończonej porażki

(c) $A \notin T_P \downarrow \omega$

Dolne potęgi operatora $\mathbf{T_P}$

$$T_P \downarrow 0 = B_P$$

$$T_P \downarrow (i+1) = T_P(T_P \downarrow i)$$

$$T_P \downarrow \omega = \bigcap_{i=0}^{\infty} T_P \downarrow i$$

Wniosek
Z równoważności warunków (a) i (b):

$$F_P = B_P - T_P \downarrow \omega$$

reguła NF — wyprowadzenie $\neg A$, gdzie $A \notin T_P \downarrow \omega$

Reguła Herbranda

P — program, A — atom ustalony

$$\frac{A \notin \text{gfp}(\ T_P\ )}{\neg A}$$

gdzie: $\text{gfp}(T_P)$ — największy punkt stały operatora T_P

Przykład

$$\text{lfp}(T_P) \neq \text{gfp}(T_P)$$

P: p $\leftarrow$ p
q $\leftarrow$

$T_P \uparrow 1 = T_P(\emptyset) = \{\ \text{q}\ \} = T_P \uparrow 2 = \text{lfp}(T_P)$

$T_P \downarrow 1 = T_P(B_P) = \{\ \text{p, q}\ \} = \text{gfp}(T_P)$

Przykład

$$\mathrm{gfp}(T_P) \neq T_P \downarrow \omega$$

P: p(f(x)) $\leftarrow$ p(x)
q(a) $\leftarrow$ p(x)

$T_P \downarrow \omega = \{$ q(a) $\}$
gfp$(T_P) = T_P \downarrow (\omega + 1) = \emptyset$

$B_P = \{$ q(a), p(a), q(f(a)), p(f(a)), ... $\}$
$M_P = \emptyset$

zatem: CWA(P) $= B_P$

gfp$(T_P) = \emptyset$, czyli RH(P) $= B_P$

w tym przypadku reguły CWA i Herbranda mają jednakową moc

lecz $T_P \downarrow \omega \neq \emptyset$,
mamy NF(P) $= B_P - \{$ q(a) $\}$

czyli stosując regułę NF nie uzyskamy $\neg$ q(a)

rzeczywiście, SLD-drzewo dla $\leftarrow$ q(a) jest nieskończone

SLD-wywód (kolejne rezolwenty):

$\leftarrow$ q(a)
$\leftarrow$ p(x_0) $\{x_0/f(x_1)\}$
$\leftarrow$ p(x_1) $\{x_1/f(x_2)\}$
$\leftarrow$ p(x_2) $\{x_2/f(x_3)\}$
....

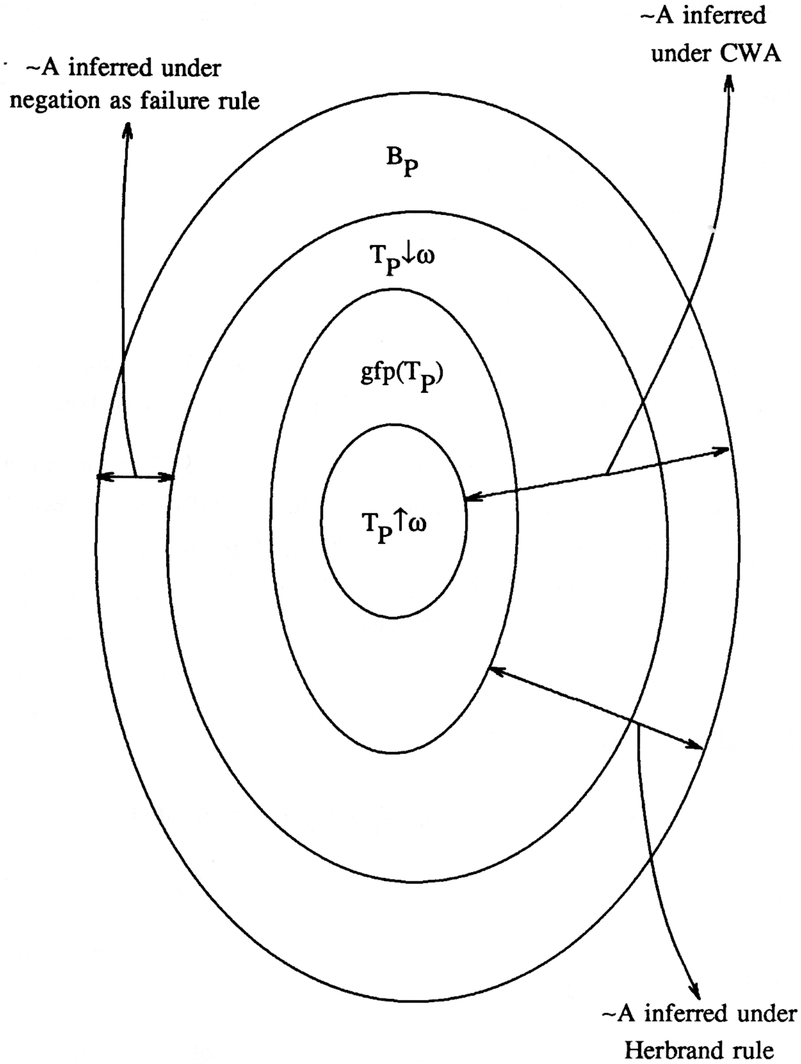
jest to sprawiedliwe SLD-drzewo

Uporządkowanie reguł wnioskowania wg ich mocy

$$M_P = \mathrm{lfp}(T_P) = T_P \uparrow \omega \ \subseteq \ \mathrm{gfp}(T_P) \ \subseteq T_P \ \downarrow \omega \subseteq B_P$$

w ogólności, powyższe trzy zbiory są różne,
czyli te trzy reguły wnioskowania są istotnie różne

Graficzna ilustracja zależności między różnymi regułami wnioskowania do wyprowadzania negatywnych informacji



(Rysunek pochodzi z książki Johna Lloyda „Foundations of Logic Programming”.)

Przykład

loves(x, y) ← mother(x), child_of(y, x)
loves(john, tom) ←
loves(mary, john) ←
mother(mary) ←
child_of(tom, mary) ←

za pomocą SLD-rezolucji nie możemy wyprowadzić np.:
mother(john) ani też ¬ mother(john)

nie wiemy również czy: loves(john, mary)

stosując regułę CWA otrzymamy: ¬ loves(john, mary)

stosując regułę NF również otrzymamy: ¬ loves(john, mary)
(SLD-drzewo skończonej porażki)

Zanegowane formuły nie ustalone

rozważmy np. formułę: ¬ loves(x, john)

dla klauzuli negatywnej: ← loves(x, john)
otrzymamy odpowiedź obliczoną: $\theta = \{$ x/mary $\}$
ponieważ loves(mary, john) jest logiczną konsekwencją

ale my szukamy odpowiedzi, dla których dana formuła nie jest logiczną konsekwencją

lecz istnieje np. tom: ¬ loves(tom, john)
gdyby nikt nie kochał johna, to uzyskalibyśmy odpowiedź

natomiast dla zapytania ← loves(x, mary) mamy SLD-drzewo skończonej porażki, czyli wyprowadzimy ¬ loves(x, mary)