Strategisch Redeneren

Redeneren (in logica):

- hoeft niet te eindigen
- kan zeer inefficiënt zijn
- hoeft de oplossingen niet altijd in de juiste volgorde op te leveren
- Otter: automatisch redenerend programma (voor clauses)
- Prolog: programmeertaal op basis van subset van predicatenlogica

Halfbeslisbaarheid

S: verzameling clauses in predicatenlogica

- Als S onvervulbaar is (en we weten dat), dan zal resolutie na eindige tijd $\Box$ afleiden ($S \vdash_{\mathcal{R}} \Box$)

  Echter alleen als alle mogelijke resolutiestappen ook in principe uitgevoerd kunnen worden ...
- Als S vervulbaar is (en we weten dit niet), dan zal resolutie niet altijd termineren

$\Rightarrow$

- Kennis over vervulbaarheid van S belangrijk
- Keuze van volgorde van toepassen van resolutie belangrijk

Resolutie als zoeken

Resolutie(regel) $\mathcal{R}$ (J.A. Robinson, 1965):

$$\frac{C \vee L, \quad C' \vee \neg L}{D}$$

met C, C' clauses, en

- $\mathrm{Cl} = C \vee L$ en $\mathrm{Cl}' = C' \vee \neg L$ parent clauses
- D de (binaire) resolvente gelijk aan $C \vee C'$ met verwijdering van dubbele literals

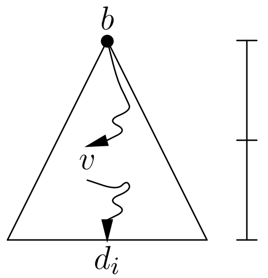
Toestandsruimte:

- $\mathrm{Cl} = C \vee L$ en $\mathrm{Cl}' = C' \vee \neg L$ zijn twee toestanden in de toestandsruimte
- D is een nieuwe toestand

Zoekstrategieën

- Zoeken in een dynamisch gegenereerde toestandsruimte (vs. statische graaf)
- Ongeïnformeerde zoekstrategieën:
  - depth-first strategie, backtracking
  - breadth-first strategie
  - bounded ...
- Geïnformeerde (heuristische) zoekstrategieën:
  - hill-climbing: kies de beste, vergeet de rest
  - best-first strategie: kies de beste, onthoud de rest
  - tabu search: kies de beste, onthoud wat gedaan

Evaluatiefunctie f^*



$$g^*(v) = \bar{k}(b, v)$$

$$h^*(v) = \bar{k}(v, d_i)$$

Evaluatiefunctie: $f^*(v) = g^*(v) + h^*(v)$

- b: begintoestand
- d_i: een van de doeltoestanden
- g^*: optimale kosten van b naar v
- h^*: optimale heuristische kosten

$$h^*(v) = \min\{\bar{k}(v, d_i) \mid d_i \text{ doeltoestand}\}$$

Onbeperkte resolutie

Beschouw de volgende verzameling clauses:
$S = \{P, \neg P \vee Q, \neg P \vee \neg Q \vee R, \neg R\}$; genummerd:

1. P
2. $\neg P \vee Q$
3. $\neg P \vee \neg Q \vee R$
4. $\neg R$

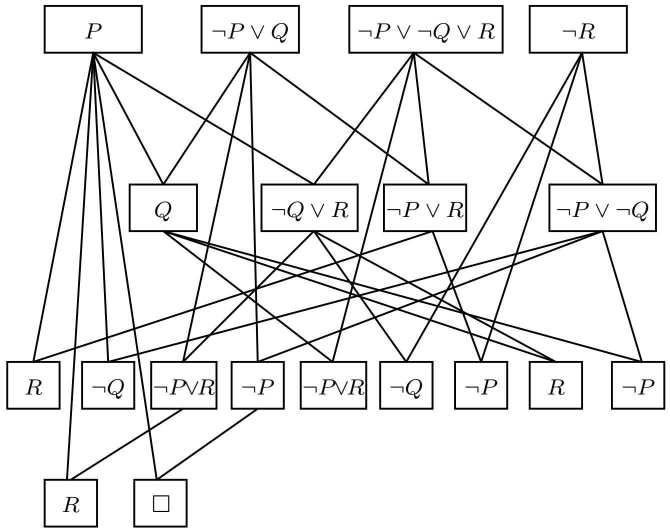
Via resolutie afgeleide clauses:

5. Q (via 1 en 2)
6. $\neg Q \vee R$ (via 1 en 3)
7. $\neg P \vee R$ (via 2 en 3)
8. $\neg P \vee \neg Q$ (via 3 en 4)

Vervolg

9. R (via 1 en 7)
10. $\neg Q$ (via 1 en 8)
11. $\neg P \vee R$ (via 2 en 6)
12. $\neg P$ (via 2 en 8)
13. $\neg P \vee R$ (via 3 en 5)
14. $\neg Q$ (via 4 en 6)
15. $\neg P$ (via 4 en 7)
16. R (via 5 en 6)
17. R (via 5 en 6)
18. $\neg P$ (via 5 en 8)
19. R (via 1 en 11)
20. $\square$ (via 1 en 12) $\Rightarrow$ 16 resolutiestappen

Refutatiegraaf



Strategie: laag voor laag – breadth-first strategie

Semantische resolutie

S: verzameling clauses die inconsistent is
($\Rightarrow S \vdash_{\mathcal{R}} \Box$)

Algemene idee:

1. definieer een waardering w voor clauses in S
2. splits de verzameling clauses S op in S_1 en S_2, met $\models_w S_1$ en $\not\models_w S_2$
3. kies één parent clause uit S_1 en één uit S_2
4. voeg resolvente toe aan S_1 of S_2 (afhankelijk van w)

Voorbeelden:

- set-of-support strategie, en
- hyperresolutie

Voorbeeld

Beschouw (opnieuw): $S = \{P, \neg P \vee Q, \neg P \vee \neg Q \vee R, \neg R\}$, met

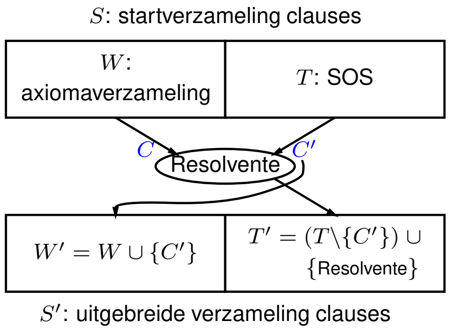
$$\begin{aligned} w(P) &= \mathit{false} \\ w(Q) &= \mathit{false} \\ w(R) &= \mathit{false} \end{aligned}$$

Resulterende opsplitsing van S:

$$\begin{aligned} S_1 &= \{P\} \\ S_2 &= \{\neg P \vee Q, \neg P \vee \neg Q \vee R, \neg R\} \end{aligned}$$

Resulterende resolventen (breadth-first): Q, $\neg Q \vee R$, $\neg P \vee R$, R en $\Box$

Set-of-support (SOS) strategie



Opzet:

- S: een verzameling clauses
- $T \subseteq S$ set of support (SOS)
- $W = S \backslash T$ axiomaverzameling (usuable set)
- parent clauses: $C \in T$ en $C' \in W$
- resolvente wordt toegevoegd aan T

Voorbeeld

$$\begin{aligned} \mathrm{KB} = \{ & \mathit{Vader}(\mathit{Zeus}, \mathit{Ares}), \\ & \mathit{Moeder}(\mathit{Hera}, \mathit{Ares}), \\ & \mathit{Vader}(\mathit{Ares}, \mathit{Harmonia}), \\ & \mathit{Moeder}(\mathit{Aphrodite}, \mathit{Harmonia}), \\ & \forall x \forall y (\mathit{Moeder}(x,y) \rightarrow \mathit{Ouder}(x,y)) \\ & \forall x \forall y (\mathit{Vader}(x,y) \rightarrow \mathit{Ouder}(x,y))\} \\ & \neg \mathit{Grootouder}(\ldots) \cdots \end{aligned}$$

- objecten: goden uit de Griekse mythologie
- relaties: Vader, Moeder, Ouder, Grootouder

Vraag: Is Zeus een grootouder van Harmonia?

Mogelijke opzet van W en SOS

'Typen' clauses:

- assumpties: clauses die een bepaald probleem beschrijven – in W
- speciale clauses: clauses die bepaalde eigenschappen van de te bewijzen vragen definiëren – in W
- hypothesen of doelen – in T (SOS)

Familierelatie:

$$W = \{Vader(Zeus, Ares), Moeder(Hera, Ares), Vader(Ares, Harmonia), Moeder(Aphrodite, Harmonia), \neg Moeder(x,y) \vee Ouder(x,y), \neg Vader(x,y) \vee Ouder(x,y)\} Grootouder(x,z) \vee \neg Ouder(x,y) \vee \neg Ouder(y,z)\}$$

met bijvoorbeeld $T = \{\neg Grootouder(Zeus, Harmonia)\}$

Resultaat (via Otter)

```
set(binary_res).

list(usable).
1 [] Vader(Zeus,Ares).
2 [] Moeder(Hera,Ares).
3 [] Vader(Ares,Harmonia).
4 [] Moeder(Aphrodite,Harmonia).
5 [] -Moeder(x,y)|Ouder(x,y).
6 [] -Vader(x,y)|Ouder(x,y).
7 [] Grootouder(x,z)| -Ouder(x,y)| -Ouder(y,z).
end_of_list.

list(sos).
8 [] -Grootouder(Zeus,Harmonia).
end_of_list.
```


Resultaat (via Otter)

```
---------------- PROOF ----------------
1 [] Vader(Zeus,Ares).
3 [] Vader(Ares,Harmonia).
6 [] -Vader(x,y)|Ouder(x,y).
7 [] Grootouder(x,z)| -Ouder(x,y)| -Ouder(y,z).
8 [] -Grootouder(Zeus,Harmonia).
9 [binary,8.1,7.1] -Ouder(Zeus,x)|
                   -Ouder(x,Harmonia).
10 [binary,9.1,6.2] -Ouder(x,Harmonia)|
                    -Vader(Zeus,x).
16 [binary,10.2,1.1] -Ouder(Ares,Harmonia).
17 [binary,16.1,6.2] -Vader(Ares,Harmonia).
18 [binary,17.1,3.1] $F.
------------ end of proof -------------
```

(hier staat `$F` voor $\Box$)

Hyperresolutie

Als S een eindige verzameling clauses is met waardering w dan heet $\{S_1, \ldots, S_n, N\} \subseteq S$ een semantisch conflict als:

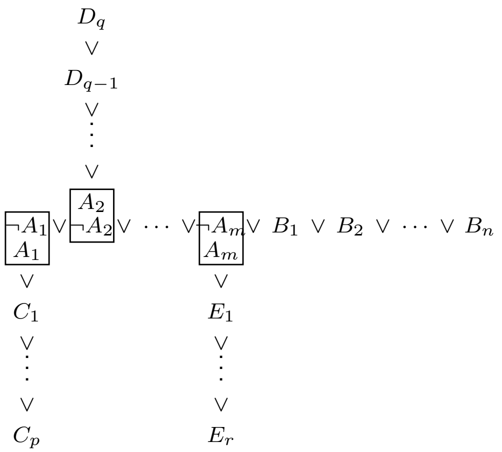
1. $w(S_i) = \mathit{false}$, $1 \leq i \leq n$
2. als $R_1 = N$, dan $\{S_i, R_i\} \vdash_{\mathcal{B}} R_{i+1}$, $i = 1, \ldots, n$
3. $w(R_{n+1}) = \mathit{false}$

Naamgeving:

- S_i: satellieten (of electronen)
- N: de nucleus, en
- R_{n+1}: de hyperresolvente

$\frac{S_1, \ldots, S_n, N}{R_{n+1}}$ is de hyperresolutieregel $\mathcal{H}$; afleiding via hyperresolutie: $S \vdash_{\mathcal{H}} C$

Positieve hyperresolutie $S \vdash_{\mathcal{P}} \Box$



- Alle positieve literals $L : w(L) = \mathit{false}$, dan positieve hyperresolutie
- Satellieten en hyperresolvente: positief

Positieve hyperresolutie & SOS strategie

Voorbeeld:

$$\{\neg P \vee \neg Q \vee R,\ \ P,\ \ Q\} \vdash_{\mathcal{H}} R$$

In combinatie met de SOS-strategie:

- positieve hyperresolutie: levert alleen positieve resolventen op
- SOS met positieve hyperresolutie strategie: mogelijk onvolledig

Voorbeeld:

$S = T \cup W$ met

$T = \{\neg P \vee \neg R\}$

$W = \{P,\ R \vee \neg Q,\ Q\}$

waarin T de SOS; W is vervulbaar en $T \cup W \vDash \Box$

Otter voorbeeld

```
list(usable).
1 [] p.
2 [] -q|r.
3 [] q.
end_of_list.

list(sos).
4 [] -p| -r.
end_of_list.

=========== start of search ===========
given clause #1: (wt=2) 4 [] -p| -r.
Search stopped because sos empty.

============ end of search ============
```


Negatieve hyperresolutie $S \vdash_{\mathcal{N}} \Box$

- Alle positieve literals $L : w(L) = \mathit{true}$, dan negatieve hyperresolutie
- Satellieten en hyperresolvente: negatief

Voorbeeld:

$S = \{\neg P \vee \neg Q \vee R \vee S,\ P \vee S,\ Q \vee R,\ \neg R,\ \neg S\}$

1. $\neg P \vee \neg Q \vee R \vee S$
2. $P \vee S$
3. $Q \vee R$
4. $\neg R$
5. $\neg S$
6. $\neg P \vee \neg Q$ (via clause 1, 4 en 5)
7. $\neg Q$ (via clause 2, 5 en 6)
8. $\Box$ (via clause 3,4 en 7). Dus: $S \vdash_{\mathcal{N}} \Box$

Weg met de zwakke clauses!

Beschouw:

- $P(x)$ (clause, dus x universeel gekwantificeerd)
- $\neg Q(a) \vee P(b)$ (a, b constanten)

Als $\models_I \forall x P(x)$ dan ook $\models_I \neg Q(a) \vee P(b)$ (voor interpretatie I), dus $\forall x P(x) \models \neg Q(a) \vee P(b)$

Welke clause moet weg?

1. $P(x)$ en $\neg Q(a) \vee P(b)$ samen in S: zinloos
2. $P(x)$ is sterker dan $\neg Q(a) \vee P(b)$
3. $\neg Q(a) \vee P(b)$ verwijderen

Net als hyperresolutie, pruning/snoeien toestandsruimte

Subsumptie

Gegeven:

- S_C: de verzameling literals in de clause C
- $S_{C'}$: verzameling literals in de clause C'
- C subsumeert C' $\Leftrightarrow$ er bestaat een substitutie σ: $S_C\sigma \subseteq S_{C'}$
- C' wordt gesubsumeerd clause genoemd

Voorbeeld:
$P(x)$ versus $\neg Q(a) \vee P(b)$: $\{P(x)\}\sigma \subset \{Q(a), P(b)\}$, met $\sigma = \{b/x\}$

In de context van resolutie: $S = \{P(x), P(b) \vee Q(a), \neg P(y)\}$, dus $P(b) \vee Q(a)$ is overbodig

Voor- en terugwaartse subsumptie

- Terugwaartse subsumptie (back subsumption): een nieuwe resolvente subsumeert clauses

```
given clause #2: 6
   [binary,5.1,2.3] -p| -r.
** KEPT: 7 [binary,6.1,4.1] -r.
** KEPT: 8 [binary,6.2,3.2] -p| -q.
7 back subsumes 6.
given clause #3: 7 [binary,6.1,4.1] -r.
** KEP: 9 [binary,7.1,3.2] -q.
9 back subsumes 8.
```

- Voorwaartse subsumptie (forward subsumption): een resolvente wordt gesubsumeerd door een gegeven clause

Speciale resolutiemethoden

- Rekening houdend met unit clauses, i.e. clauses bestaande uit één literal
- UR (unit-resulting) resolutieregel: $\frac{U_1,\ldots,U_n,N}{R_{n+1}}$, met:
  - U_i zijn unit clauses
  - als $R_1 = N$, dan $\{U_i, R_i\} \vdash_{\mathcal{B}} R_{i+1}$, voor $i = 1, \ldots, n$
  - R_{n+1} is een unit clause of de lege clause

Voorbeeld:

$$S = \{\neg P \vee Q \vee V,\ P,\ \neg Q\}$$

dan is $U_1 = P, U_2 = \neg Q$, $N = R_1 = \neg P \vee Q \vee R$ en $R_3 = V$

Speciale resolutiemethoden

- Horn clause: clause met maximaal één positieve literal $\neg A_1 \vee \cdots \vee \neg A_m \vee B$, ook genoteerd als $B \leftarrow A_1, \ldots, A_m$
- SLD (Selected, Linear, Definite) resolutie (voor Horn clauses):

$$\frac{B \leftarrow A_1, \ldots, A_m, \leftarrow B}{\leftarrow A_1, \ldots, A_m}$$

  met $\leftarrow B$ een goal clause

Voorbeeld:

$S = \{R(g(x)) \leftarrow T(x, y, f(x)),$
$\quad T(a, b, f(a)),\ P(v, w) \leftarrow R(v)\}$

en $\leftarrow P(u, b)$. Hieruit is met SLD-resolutie $\Box$ af te leiden