

Paradigme de Programare

Conf. dr. ing. Andrei Olaru

andrei.olaru@cs.pub.ro | cs@andreiolaru.ro
Departamentul de Calculatoare

2018

Cursul 11

Logica cu predicate de ordinul I

Cursul 11: Logica cu predicate de ordinul I $P \vee \bar{P}$

- 1 Logica propozițională
- 2 Evaluarea valorii de adevăr
- 3 Logica cu predicate de ordinul întâi
- 4 LPOI – Semantică
- 5 Forme normale
- 6 Unificare și rezoluție

- formalism simbolic pentru reprezentarea faptelor și raționament.
- se bazează pe ideea de **valoare de adevăr** – e.g. *Adevărat* sau *Fals*.
- permite realizarea de argumente (argumentare) și demonstrații – deducție, inducție, rezoluție, etc.

Logica propozițională

- Cadru pentru:
 - descrierea proprietăților obiectelor, prin intermediul unui limbaj, cu o semantică asociată;
 - deducerea de noi proprietăți, pe baza celor existente.
- Expresia din limbaj: propoziția, corespunzătoare unei afirmații, ce poate fi adevărată sau falsă.
- Exemplu: “Afară este frumos.”
- Accepții asupra unei propoziții:
 - secvența de simboluri utilizate sau
 - înțelesul propriu-zis al acesteia, într-o interpretare.

- 2 categorii de propoziții
 - simple \rightarrow fapte **atomice**: “Afară este frumos.”
 - compuse \rightarrow **relații** între propoziții mai simple: “Telefonul sună și câinele latră.”
- Propoziții simple: p, q, r, \dots
- Negatii: $\neg \alpha$
- Conjuncții: $(\alpha \wedge \beta)$
- Disjuncții: $(\alpha \vee \beta)$
- Implicații: $(\alpha \Rightarrow \beta)$
- Echivalențe: $(\alpha \Leftrightarrow \beta)$

- Scop: dezvoltarea unor mecanisme de prelucrare, aplicabile **independent** de valoarea de adevăr a propozițiilor într-o situație particulară.
- Accent pe **relațiile** între propozițiile compuse și cele constituente.
- Pentru explicitarea propozițiilor → utilizarea conceptului de **interpretare**.

+ **Interpretare** Mulțime de **asocieri** între fiecare propoziție **simplă** din limbaj și o valoare de adevăr.



Exemplu

Interpretarea I :

- $p^I = false$
- $q^I = true$
- $r^I = false$

Interpretarea J :

- $p^J = true$
- $q^J = true$
- $r^J = true$

- cum știu dacă p este adevărat sau fals? Pot ști dacă știu **interpretarea** – p este doar un *nume* pe care îl dau unei propoziții concrete.

- Sub o interpretare **fixată** → **dependența** valorii de adevăr a unei propoziții compuse de valorile de adevăr ale celor constituente

- **Negație**: $(\neg\alpha)^I = \begin{cases} true & \text{dacă } \alpha^I = false \\ false & \text{altfel} \end{cases}$

- **Conjuncție**:

$$(\alpha \wedge \beta)^I = \begin{cases} true & \text{dacă } \alpha^I = true \text{ și } \beta^I = true \\ false & \text{altfel} \end{cases}$$

- **Disjuncție**:

$$(\alpha \vee \beta)^I = \begin{cases} false & \text{dacă } \alpha^I = false \text{ și } \beta^I = false \\ true & \text{altfel} \end{cases}$$

- Implicație:

$$(\alpha \Rightarrow \beta)^I = \begin{cases} false & \text{dacă } \alpha^I = true \text{ și } \beta^I = false \\ true & \text{altfel} \end{cases}$$

- Echivalență:

$$(\alpha \Leftrightarrow \beta)^I = \begin{cases} true & \text{dacă } \alpha \Rightarrow \beta \wedge \beta \Rightarrow \alpha \\ false & \text{altfel} \end{cases}$$

Evaluarea valorii de adevăr

Cum determinăm valoarea de adevăr?

+ **Evaluare** Determinarea **valorii de adevăr** a unei **propoziții**, sub o **interpretare**, prin aplicarea regulilor semantice anterioare.



Exemplu

- Interpretarea I :

- $p^I = \text{false}$

- $q^I = \text{true}$

- $r^I = \text{false}$

- Propoziția: $\phi = (p \wedge q) \vee (q \Rightarrow r)$

$$\phi^I = (\text{false} \wedge \text{true}) \vee (\text{true} \Rightarrow \text{false}) = \text{false} \vee \text{false} = \text{false}$$

+ | **Satisfiabilitate** Proprietatea unei propoziții care este adevărată sub **cel puțin o** interpretare. Acea interpretare **satisface** propoziția.

+ | **Validitate** Proprietatea unei propoziții care este adevărată în **toate** interpretările. Propoziția se mai numește **tautologie**.

Ex | Exemplu Propoziția $p \vee \neg p$ este **validă**.

+ | **Nesatisfiabilitate** Proprietatea unei propoziții care este falsă în **toate** interpretările. Propoziția se mai numește **contradicție**.

Ex | Exemplu Propoziția $p \wedge \neg p$ este **nesatisfiabilă**.

Ex) Metoda tabelii de adevăr

| p | q | r | $(p \wedge q) \vee (q \Rightarrow r)$ |
|--------------|--------------|--------------|---------------------------------------|
| <i>true</i> | <i>true</i> | <i>true</i> | <i>true</i> |
| <i>true</i> | <i>true</i> | <i>false</i> | <i>true</i> |
| <i>true</i> | <i>false</i> | <i>true</i> | <i>true</i> |
| <i>true</i> | <i>false</i> | <i>false</i> | <i>true</i> |
| <i>false</i> | <i>true</i> | <i>true</i> | <i>true</i> |
| <i>false</i> | <i>true</i> | <i>false</i> | <i>false</i> |
| <i>false</i> | <i>false</i> | <i>true</i> | <i>false</i> |
| <i>false</i> | <i>false</i> | <i>false</i> | <i>false</i> |

\Rightarrow Propoziția $(p \wedge q) \vee (q \Rightarrow r)$ este **satisfiabilă**.

+ **Derivabilitate logică** Proprietatea unei propoziții de a reprezenta **consecința logică** a unei mulțimi de alte propoziții, numite **premise**. Mulțimea de propoziții Δ derivă propoziția ϕ ($\Delta \models \phi$) dacă și numai dacă **orice** interpretare care satisface toate propozițiile din Δ satisface și ϕ .



Exemplu

- $\{p\} \models p \vee q$
- $\{p, q\} \models p \wedge q$
- $\{p\} \not\models p \wedge q$
- $\{p, p \Rightarrow q\} \models q$

- Verificabilă prin metoda tabelii de adevăr: **toate** intrările pentru care **premisele** sunt adevărate trebuie să inducă adevărul **concluziei**.

Demonstrăm că $\{p, p \Rightarrow q\} \models q$.

| p | q | $p \Rightarrow q$ |
|--------------|--------------|-------------------|
| <i>true</i> | <i>true</i> | <i>true</i> |
| <i>true</i> | <i>false</i> | <i>false</i> |
| <i>false</i> | <i>true</i> | <i>true</i> |
| <i>false</i> | <i>false</i> | <i>true</i> |

Singura intrare în care ambele premise, p și $p \Rightarrow q$, sunt adevărate, precizează și adevărul concluziei, q .



Exemplu

- $\{\phi_1, \dots, \phi_n\} \models \phi$

sau

- Propoziția $\phi_1 \wedge \dots \wedge \phi_n \Rightarrow \phi$ este **validă**

sau

- Propoziția $\phi_1 \wedge \dots \wedge \phi_n \wedge \neg \phi$ este **nesatisfiabilă**

- Creșterea **exponențială** a numărului de interpretări în raport cu numărul de propoziții simple.
- De aici, **diminuarea** valorii practice a metodelor **semantice**, precum cea a tabelii de adevăr.
- Alternativ, metode **sintactice**, care manipulează doar reprezentarea simbolică.
 - Inferență → Derivare **mecanică** → demers de **calcul**, în scopul verificării derivabilității logice.
 - folosind **metodele de inferență**, putem construi o **mașină de calcul**.

Definiție

+ **Inferența** – Derivarea **mecanică** a concluziilor unui set de premise.

+ **Regulă de inferență** – **Procedură** de calcul capabilă să deriveze concluziile unui set de premise. Derivabilitatea mecanică a concluziei ϕ din mulțimea de premise Δ , utilizând **regula de inferență** *inf*, se notează $\Delta \vdash_{inf} \phi$.

⊗ Modus Ponens (MP) :

$$\alpha \Rightarrow \beta$$

$$\alpha$$

$$\beta$$

⊗ Modus Tollens :

$$\alpha \Rightarrow \beta$$

$$\neg \beta$$

$$\neg \alpha$$

+ **Consistență (*soundness*)** – Regula de inferență determină **numai** propoziții care sunt, într-adevăr, **consecințe logice** ale premiselor. $\Delta \vdash_{inf} \phi \Rightarrow \Delta \models \phi$.

+ **Completitudine (*completeness*)** – Regula de inferență determină **toate consecințele logice** ale premiselor. $\Delta \models \phi \Rightarrow \Delta \vdash_{inf} \phi$.

- Ideal, **ambele** proprietăți – “nici în plus, nici în minus” – $\Delta \models \phi \Leftrightarrow \Delta \vdash_{inf} \phi$
- **Incompletitudinea** regulii *Modus Ponens*, din imposibilitatea scrierii oricărei propoziții ca implicație.

Logica cu predicate de ordinul întâi

- **Extensie** a logicii propoziționale, cu explicitarea:
 - **obiectelor** din universul problemei;
 - **relațiilor** dintre acestea.
- Logica propozițională:
 - p : “Andrei este prieten cu Bogdan.”
 - q : “Bogdan este prieten cu Andrei.”
 - $p \Leftrightarrow q$ – pot ști doar din interpretare.
 - **Opacitate** în raport cu obiectele și relațiile referite.
- FOPL:
 - Generalizare: $prieten(x, y)$: “ x este prieten cu y .”
 - $\forall x. \forall y. (prieten(x, y) \Leftrightarrow prieten(y, x))$
 - Aplicare pe cazuri **particulare**.
 - **Transparentă** în raport cu obiectele și relațiile referite.

- + **Constante** – obiecte particulare din universul discursului: *c, d, andrei, bogdan, ...*
- + **Variabile** – obiecte generice: *x, y, ...*
- + **Simboluri funcționale** – *succesor, +, abs ...*
- + **Simboluri relaționale** (**predicate**) – relații *n*-are peste obiectele din universul discursului:
prieten = {(andrei, bogdan), (bogdan, andrei), ...},
impar = {1, 3, ...}, ...
- + **Conectori logici** $\neg, \wedge, \vee, \Rightarrow, \Leftarrow$
- + **Cuantificatori** \forall, \exists

+ Termeni (obiecte):

- Constante;
- Variabile;
- Aplicații de funcții: $f(t_1, \dots, t_n)$, unde f este un simbol **funcțional** n -ar și t_1, \dots, t_n sunt termeni.

Ex Exemple

- $\text{succesor}(4)$: succesul lui 4, și anume 5.
- $+(2, x)$: aplicația funcției de adunare asupra numerelor 2 și x , și, totodată, suma lor.

+ **Atomi** (relatii): atomul $p(t_1, \dots, t_n)$, unde p este un **predicat** n -ar și t_1, \dots, t_n sunt termeni.

Ex) Exemple

- $impar(3)$
- $varsta(ion, 20)$
- $= (+ (2, 3), 5)$

+ **Propoziții** (fapte) – dacă x variabilă, A atom, și α și β propoziții, atunci o propoziție are forma:

- Fals, Adevărat: \perp, \top
- **Atomi**: A
- **Negații**: $\neg\alpha$
- **Conectori**: $\alpha \wedge \beta, \alpha \Rightarrow \beta, \dots$
- **Cuantificări**: $\forall x.\alpha, \exists x.\alpha$

“Sora Ioanei are un prieten deștept”



Exemplu

“Sora Ioanei are un prieten deștept”

$$\exists X. \underbrace{\underbrace{X}_{\text{termen}}, \underbrace{\text{sora}(\text{ioana})}_{\text{termen}}}_{\text{atom/propoziție}} \wedge \underbrace{\text{deștept}(X)}_{\text{atom/propoziție}}$$

$\underbrace{\hspace{15em}}_{\text{propoziție}}$



Exemplu

LPOI – Semantică

+ Interpretarea constă din:

- Un domeniu nevid, D , de concepte (obiecte)
- Pentru fiecare constantă c , un element $c^I \in D$
- Pentru fiecare simbol funcțional, n -ar f , o funcție $f^I : D^n \rightarrow D$
- Pentru fiecare predicat n -ar p , o funcție $p^I : D^n \rightarrow \{false, true\}$.

- Atom:

$$(p(t_1, \dots, t_n))' = p'(t_1', \dots, t_n')$$

- Negăție, conectori, implicații: v. logica propozițională

- Cuantificare **universală**:

$$(\forall x. \alpha)' = \begin{cases} false & \text{dacă } \exists d \in D. \alpha'_{[d/x]} = false \\ true & \text{altfel} \end{cases}$$

- Cuantificare **existențială**:

$$(\exists x. \alpha)' = \begin{cases} true & \text{dacă } \exists d \in D. \alpha'_{[d/x]} = true \\ false & \text{altfel} \end{cases}$$

Ex) Exemple cu cuantificatori

1 “Vrăbia mălai visează.”

Ex) Exemple cu cuantificatori

1 “Vrabia mălai visează.”

$$\forall x. (vrabie(x) \Rightarrow viseaza(x, malai))$$

Ex) Exemple cu cuantificatori

1 “Vrăbia mălai visează.”

$$\forall x. (vrăbie(x) \Rightarrow viseaza(x, malai))$$

2 “Unele vrăbii visează mălai.”

Ex) Exemple cu cuantificatori

1 “Vrăbia mălai visează.”

$$\forall x. (vrăbie(x) \Rightarrow viseaza(x, malai))$$

2 “Unele vrăbii visează mălai.”

$$\exists x. (vrăbie(x) \wedge viseaza(x, malai))$$

Ex) Exemple cu cuantificatori

- 1 “Vrăbia mălai visează.”
 $\forall x. (vrăbie(x) \Rightarrow viseaza(x, malai))$
- 2 “Unele vrăbii visează mălai.”
 $\exists x. (vrăbie(x) \wedge viseaza(x, malai))$
- 3 “Nu toate vrăbiile visează mălai.”

Ex) Exemple cu cuantificatori

- 1 “Vrăbia mălai visează.”
 $\forall x. (vrăbie(x) \Rightarrow viseaza(x, malai))$
- 2 “Unele vrăbii visează mălai.”
 $\exists x. (vrăbie(x) \wedge viseaza(x, malai))$
- 3 “Nu toate vrăbiile visează mălai.”
 $\exists x. (vrăbie(x) \wedge \neg viseaza(x, malai))$

Ex) Exemple cu cuantificatori

- 1 “Vrabia mălai visează.”
 $\forall x.(vrabie(x) \Rightarrow viseaza(x, malai))$
- 2 “Unele vrăbii visează mălai.”
 $\exists x.(vrabie(x) \wedge viseaza(x, malai))$
- 3 “Nu toate vrăbiile visează mălai.”
 $\exists x.(vrabie(x) \wedge \neg viseaza(x, malai))$
- 4 “Nicio vrabie nu visează mălai.”

Ex Exemple cu cuantificatori

- 1 “Vrabia mălai visează.”
 $\forall x. (vrabie(x) \Rightarrow viseaza(x, malai))$
- 2 “Unele vrăbii visează mălai.”
 $\exists x. (vrabie(x) \wedge viseaza(x, malai))$
- 3 “Nu toate vrăbiile visează mălai.”
 $\exists x. (vrabie(x) \wedge \neg viseaza(x, malai))$
- 4 “Nicio vrabie nu visează mălai.”
 $\forall x. (vrabie(x) \Rightarrow \neg viseaza(x, malai))$

Ex | Exemple cu cuantificatori

- 1 “Vrabia mălai visează.”
 $\forall x. (vrabie(x) \Rightarrow viseaza(x, malai))$
- 2 “Unele vrăbii visează mălai.”
 $\exists x. (vrabie(x) \wedge viseaza(x, malai))$
- 3 “Nu toate vrăbiile visează mălai.”
 $\exists x. (vrabie(x) \wedge \neg viseaza(x, malai))$
- 4 “Nicio vrabie nu visează mălai.”
 $\forall x. (vrabie(x) \Rightarrow \neg viseaza(x, malai))$
- 5 “Numai vrăbiile visează mălai.”

Ex | Exemple cu cuantificatori

- 1 “Vrabia mălai visează.”
 $\forall x.(vrabie(x) \Rightarrow viseaza(x, malai))$
- 2 “Unele vrăbii visează mălai.”
 $\exists x.(vrabie(x) \wedge viseaza(x, malai))$
- 3 “Nu toate vrăbiile visează mălai.”
 $\exists x.(vrabie(x) \wedge \neg viseaza(x, malai))$
- 4 “Nicio vrabie nu visează mălai.”
 $\forall x.(vrabie(x) \Rightarrow \neg viseaza(x, malai))$
- 5 “Numai vrăbiile visează mălai.”
 $\forall x.(viseaza(x, malai) \Rightarrow vrabie(x))$

- $\forall x. (vrabie(x) \Rightarrow viseaza(x, malai))$

- $\forall x. (vrabie(x) \Rightarrow viseaza(x, malai))$
→ corect: “Toate vrăbiile visează mălai.”
- $\forall x. (vrabie(x) \wedge viseaza(x, malai))$

- $\forall x. (vrabie(x) \Rightarrow viseaza(x, malai))$
→ corect: “Toate vrăbiile visează mălai.”
- $\forall x. (vrabie(x) \wedge viseaza(x, malai))$
→ **greșit**: “Toți sunt vrăbii și toți visează mălai.”
- $\exists x. (vrabie(x) \wedge viseaza(x, malai))$

- $\forall x. (vrabie(x) \Rightarrow viseaza(x, malai))$
→ corect: “Toate vrăbiile visează mălai.”
- $\forall x. (vrabie(x) \wedge viseaza(x, malai))$
→ **greșit**: “Toți sunt vrăbii și toți visează mălai.”
- $\exists x. (vrabie(x) \wedge viseaza(x, malai))$
→ corect: “Unele vrăbii visează mălai.”
- $\exists x. (vrabie(x) \Rightarrow viseaza(x, malai))$

- $\forall x. (vrabie(x) \Rightarrow viseaza(x, malai))$
→ corect: “Toate vrăbiile visează mălai.”
- $\forall x. (vrabie(x) \wedge viseaza(x, malai))$
→ **greșit**: “Toți sunt vrăbii și toți visează mălai.”
- $\exists x. (vrabie(x) \wedge viseaza(x, malai))$
→ corect: “Unele vrăbii visează mălai.”
- $\exists x. (vrabie(x) \Rightarrow viseaza(x, malai))$
→ **greșit**: probabil nu are semnificația pe care o intenționăm. Este adevărată și dacă luăm un x care nu este vrabie (fals implică orice).

- **Necomutativitate:**

- $\forall x. \exists y. \text{viseaza}(x, y) \rightarrow$ “Toți visează la ceva anume.”
- $\exists x. \forall y. \text{viseaza}(x, y) \rightarrow$ “Există cineva care visează la orice.”

- **Dualitate:**

- $\neg(\forall x. \alpha) \equiv \exists x. \neg \alpha$
- $\neg(\exists x. \alpha) \equiv \forall x. \neg \alpha$

- Satisfiabilitate.
- Validitate.
- Derivabilitate.
- Inferență.

Forme normale

Definiții

+ **Literal** – Atom sau **negația** unui atom.

Ex) Exemplu $prieten(x, y), \neg prieten(x, y)$.

+ **Clauză** – **Mulțime** de literali dintr-o expresie clauzală.

Ex) Exemplu $\{prieten(x, y), \neg doctor(x)\}$.

+ **Forma normală conjunctivă – FNC** – Reprezentare ca **mulțime de clauze**, cu semnificație conjunctivă.

+ **Forma normală implicativă – FNI** – Reprezentare ca **mulțime de clauze** cu clauzele în forma **grupată**
 $\{\neg A_1, \dots, \neg A_m, B_1, \dots, B_n\}, \Leftrightarrow (A_1 \wedge \dots \wedge A_m) \Rightarrow (B_1 \vee \dots \vee B_n)$

+ **Clauză Horn** – Clauză în care **cel mult un** literal este în formă pozitivă:
 $\{\neg A_1, \dots, \neg A_n, A\}$,
corespunzătoare **implicației**
 $A_1 \wedge \dots \wedge A_n \Rightarrow A$.

Ex | **Exemplu** Transformarea propoziției
 $\forall x. vrabie(x) \vee ciocarlie(x) \Rightarrow pasare(x)$ în formă normală,
utilizând clauze Horn:
FNC: $\{\neg vrabie(x), pasare(x)\}, \{\neg ciocarlie(x), pasare(x)\}$

- 1 Eliminarea **implicațiilor** (\Rightarrow)
- 2 Împingerea **negațiilor** până în fața atomilor (\neg)
- 3 **Redenumirea** variabilelor cuantificate pentru obținerea **unicității** de nume (R):

$$\forall x.p(x) \wedge \forall x.q(x) \vee \exists x.r(x) \rightarrow \forall x.p(x) \wedge \forall y.q(y) \vee \exists z.r(z)$$

- 4 Deplasarea cuantificatorilor la **începutul** expresiei, conservându-le **ordinea** (forma normală *prenex*) (P):

$$\forall x.p(x) \wedge \forall y.q(y) \vee \exists z.r(z) \rightarrow \forall x.\forall y.\exists z.(p(x) \wedge q(y) \vee r(z))$$

5 Eliminarea cuantificatorilor **existențiali** (skolemizare) (S):

- Dacă **nu** este precedat de cuantificatori universali:
înlocuirea aparițiilor variabilei cuantificate printr-o
constantă (bine aleasă):

$$\exists x.p(x) \rightarrow p(c_x)$$

- Dacă este **precedat** de cuantificatori universali:
înlocuirea aparițiilor variabilei cuantificate prin aplicația
unei **funcții** unice asupra variabilelor anterior cuantificate
universal:

$$\begin{aligned} &\forall x.\forall y.\exists z.(p(x) \wedge q(y) \vee r(z)) \\ &\rightarrow \forall x.\forall y.(p(x) \wedge q(y) \vee r(f_z(x, y))) \end{aligned}$$

- 6 Eliminarea cuantificatorilor **universali**, considerați, acum, impliciți (\forall):

$$\forall x. \forall y. (p(x) \wedge q(y) \vee r(f_z(x, y))) \rightarrow p(x) \wedge q(y) \vee r(f_z(x, y))$$

- 7 **Distribuirea** lui \vee față de \wedge (\vee/\wedge):

$$\alpha \vee (\beta \wedge \gamma) \rightarrow (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$$

- 8 Transformarea expresiilor în **clauze** (C).

Conversia propozițiilor în FNC – Exemplu $P \vee \bar{P}$

Ex | Exemplu “Cine rezolvă toate laboratoarele este apreciat de cineva.”

Ex | Exemplu “Cine rezolvă toate laboratoarele este apreciat de cineva.”

$$\forall x. (\forall y. (lab(y) \Rightarrow rezolva(x, y)) \Rightarrow \exists y. apreciaza(y, x))$$

Ex Exemplu “Cine rezolvă toate laboratoarele este apreciat de cineva.”

$\forall x. (\forall y. (lab(y) \Rightarrow rezolva(x, y)) \Rightarrow \exists y. apreciaza(y, x))$

$\not\equiv \forall x. (\neg \forall y. (\neg lab(y) \vee rezolva(x, y)) \vee \exists y. apreciaza(y, x))$

$\Rightarrow \forall x. (\exists y. \neg (\neg lab(y) \vee rezolva(x, y)) \vee \exists y. apreciaza(y, x))$

$\Rightarrow \forall x. (\exists y. (lab(y) \wedge \neg rezolva(x, y)) \vee \exists y. apreciaza(y, x))$

R $\forall x. (\exists y. (lab(y) \wedge \neg rezolva(x, y)) \vee \exists z. apreciaza(z, x))$

P $\forall x. \exists y. \exists z. ((lab(y) \wedge \neg rezolva(x, y)) \vee apreciaza(z, x))$

S $\forall x. ((lab(f_y(x)) \wedge \neg rezolva(x, f_y(x))) \vee apreciaza(f_z(x), x))$

$\not\equiv (lab(f_y(x)) \wedge \neg rezolva(x, f_y(x))) \vee apreciaza(f_z(x), x)$

$\vee/\wedge (lab(f_y(x)) \vee apr(f_z(x), x)) \wedge (\neg rez(x, f_y(x)) \vee apr(f_z(x), x))$

C $\{lab(f_y(x)), apr(f_z(x), x)\}, \{\neg rez(x, f_y(x)), apr(f_z(x), x)\}$

Unificare și rezoluție

- Regulă de inferență foarte puternică.
- Baza unui demonstrator de teoreme **consistent și complet**.
- Spațiul de căutare mai mic decât în alte sisteme.
- Se bazează pe lucrul cu propoziții în **forma clauzală** (clauze):
 - propoziție = mulțime de **clauze** (semnificație conjunctivă)
 - clauză = mulțime de **literali** (semnificație disjunctivă)
 - literal = **atom** sau **atom negat**
 - atom = **propoziție simplă**



Ideea (în LP):

$$\frac{\begin{array}{l} \{p \Rightarrow q\} \\ \{\neg p \Rightarrow r\} \end{array}}{\{q, r\}} \quad \rightarrow \text{“Anularea” lui } p$$

- p falsă $\rightarrow \neg p$ adevărată $\rightarrow r$ adevărată
- p adevărată $\rightarrow q$ adevărată
- $p \vee \neg p \Rightarrow$ Cel puțin una dintre q și r adevărată ($q \vee r$)
- Forma generală a pasului de rezoluție:

$$\frac{\begin{array}{l} \{p_1, \dots, r, \dots, p_m\} \\ \{q_1, \dots, \neg r, \dots, q_n\} \end{array}}{\{p_1, \dots, p_m, q_1, \dots, q_n\}}$$

- Clauza **vidă** \rightarrow indicator de **contradicție** între premise

$$\frac{\{\neg p\} \\ \{p\}}{\{\} = \emptyset}$$

- **Mai mult de 2** rezolvenți posibili \rightarrow se alege doar unul:

$$\frac{\{p, q\} \\ \{\neg p, \neg q\}}{\{p, \neg p\} \text{ sau } \{q, \neg q\}}$$

- Demonstrarea **nesatisfiabilității** \rightarrow derivarea clauzei **vide**.
- Demonstrarea **derivabilității** concluziei ϕ din premisele $\phi_1, \dots, \phi_n \rightarrow$ demonstrarea **nesatisfiabilității** propoziției $\phi_1 \wedge \dots \wedge \phi_n \wedge \neg\phi$.
- Demonstrarea **validității** propoziției $\phi \rightarrow$ demonstrarea **nesatisfiabilității** propoziției $\neg\phi$.

Demonstrăm că $\{p \Rightarrow q, q \Rightarrow r\} \vdash p \Rightarrow r$,
i.e. mulțimea $\{p \Rightarrow q, q \Rightarrow r, \neg(p \Rightarrow r)\}$ conține o **contradicție**.



Demonstrăm că $\{p \Rightarrow q, q \Rightarrow r\} \vdash p \Rightarrow r$,
i.e. mulțimea $\{p \Rightarrow q, q \Rightarrow r, \neg(p \Rightarrow r)\}$ conține o **contradicție**.



Exemplu

1. $\{\neg p, q\}$ Premisă
2. $\{\neg q, r\}$ Premisă
3. $\{p\}$ Concluzie negată
4. $\{\neg r\}$ Concluzie negată
5. $\{q\}$ Rezoluție 1, 3
6. $\{r\}$ Rezoluție 2, 5
7. $\{\}$ Rezoluție 4, 6 \rightarrow clauza vidă

T | **Teorema Rezoluției:** Rezoluția propozițională este consistentă și completă, i.e. $\Delta \models \phi \Leftrightarrow \Delta \vdash_{rez} \phi$.

- **Terminare garantată** a procedurii de aplicare a rezoluției: număr **finit** de clauze \rightarrow număr **finit** de concluzii.

- Utilizată pentru rezoluția în LPOI
- vezi și sinteza de tip în Haskell



cum știm dacă folosind ipoteza $om(Marcel)$ și propoziția $\forall om(x) \Rightarrow are_inima(x)$ putem demonstra că $are_inima(Marcel) \rightarrow$ unificând $om(Marcel)$ și $\forall om(x)$.

- reguli:
 - o propoziție unifică cu o propoziție de aceeași formă
 - două predicate unifică dacă au același nume și parametri care unifică (om cu om , x cu $Marcel$)
 - o constantă unifică cu o constantă cu același nume
 - o variabilă unifică cu un termen ce nu conține variabila (x cu $Marcel$)

- Problemă **NP-completă**;

- Posibile legări **ciclice**;

- Exemplu:

$prieten(x, coleg_banca(x))$ și

$prieten(coleg_banca(y), y)$

MGU: $S = \{x \leftarrow coleg_banca(y), y \leftarrow coleg_banca(x)\}$

$\Rightarrow x \leftarrow coleg_banca(coleg_banca(x)) \rightarrow$ **imposibil!**

- Soluție: verificarea apariției unei variabile în **valoarea** la care a fost legată (*occurrence check*);

- Rezoluția pentru clauze **Horn**:

$$A_1 \wedge \dots \wedge A_m \Rightarrow A$$

$$B_1 \wedge \dots \wedge A' \wedge \dots \wedge B_n \Rightarrow B$$

$$\text{unificare}(A, A') = S$$

$$\text{subst}(S, A_1 \wedge \dots \wedge A_m \wedge B_1 \wedge \dots \wedge B_n \Rightarrow B)$$

- $\text{unificare}(\alpha, \beta) \rightarrow$ **substituția** sub care unifică propozițiile α și β ;
- $\text{subst}(S, \alpha) \rightarrow$ propoziția rezultată în urma **aplicării** substituției S asupra propoziției α .



Exemplu

Horses and hounds

- 1 Horses are faster than dogs.
- 2 There is a greyhound that is faster than any rabbit.
- 3 Harry is a horse and Ralph is a rabbit.
- 4 Is Harry faster than Ralph?

Exemplu Horses and Hounds

- 1 $\forall x.\forall y.horse(x) \wedge dog(y) \Rightarrow faster(x, y)$
 $\rightarrow \neg horse(x) \vee \neg dog(y) \vee faster(x, y)$
- 2 $\exists x.greyhound(x) \wedge (\forall y.rabbit(y) \Rightarrow faster(x, y))$
 $\rightarrow greyhound(Greg) ; \neg rabbit(y) \vee faster(Greg, y)$
- 3 $horse(Harry) ; rabbit(Ralph)$
- 4 $\neg faster(Harry, Ralph)$ (concluzia negată)
- 5 $\neg greyhound(x) \vee dog(x)$ (common knowledge)
- 6 $\neg faster(x, y) \vee \neg faster(y, z) \vee faster(x, z)$ (tranzitivitate)
- 7 $1 + 3a \rightarrow \neg dog(y) \vee faster(Harry, y)$ (cu $\{Harry/x\}$)
- 8 $2a + 5 \rightarrow dog(Greg)$ (cu $\{Greg/x\}$)
- 9 $7 + 8 \rightarrow faster(Harry, Greg)$ (cu $\{Greg/y\}$)
- 10 $2b + 3b \rightarrow faster(Greg, Ralph)$ (cu $\{Ralph/y\}$)
- 11 $6 + 9 + 10 \rightarrow faster(Harry, Ralph) \{Harry/x, Greg/y, Ralph/z\}$
- 12 $11 + 4 \rightarrow \square$ q.e.d.

- sintaxa și semantica în LPOI
- Forme normale, Unificare, Rezoluție în LPOI