Logică și structuri discrete

Logica predicatelor

Casandra Holotescu

casandra@cs.upt.ro

https://tinyurl.com/lecturesLSD

Logică: recapitulare

Folosim logica pentru a exprima *riguros* (*formaliza*) raționamente.

Logică: recapitulare

Folosim logica pentru a exprima *riguros* (*formaliza*) raționamente.

Logica ne permite să facem *demonstrații (deducții)*
din *axiome* (totdeauna adevărate)
și *ipoteze* (considerate adevărate în problema dată)
folosind *reguli de inferență* (de deducție)

$$\frac{p \qquad p \to q}{q} \qquad \textit{modus ponens}$$

Modus ponens e suficient, dar sunt și alte reguli de deducție valide.

Logică: recapitulare

Folosim logica pentru a exprima *riguros* (*formaliza*) raționamente.

Logica ne permite să facem *demonstrații (deducții)*
din *axiome* (totdeauna adevărate)
și *ipoteze* (considerate adevărate în problema dată)
folosind *reguli de inferență* (de deducție)

$$\frac{p \qquad p \to q}{q} \qquad \textit{modus ponens}$$

Modus ponens e suficient, dar sunt și alte reguli de deducție valide.

Logica propozițională e
consistentă: orice formulă demonstrată (teoremă) e validă
completă: orice formulă validă (tautologie) poate fi demonstrată

Folosim logica

în *specificații* pentru programe: de exemplu, sortare

```
/*@ ensures
  @   (\forall int i; 0<=i && i<a.length - 1;
  @ a[i] <= a[i+1])
  @*/
```

în condiții (*predicate*) pentru datele de prelucrat

```
M.filter (fun k v -> k < "M" && v >= 5) stud_dict
```

exprimând riguros proprietăți: axioma mulțimii vide

$$\exists empty\ \forall x\ \neg contains(empty, x)$$

descriind structuri informatice: fișiere și cataloage

$$\forall x\ ((folder(x) \land x \neq root) \to contains(parent(x), x))$$

Logica propozițională nu poate exprima tot

Un exemplu clasic: (1) Toți oamenii sunt muritori.
(2) Socrate e om.
Deci, (3) Socrate e muritor.

Acesta e un *silogism* (tipar de regulă de inferență)
logica clasică: Aristotel, stoici

Logica propozițională nu poate exprima tot

Un exemplu clasic: (1) Toți oamenii sunt muritori.
(2) Socrate e om.
Deci, (3) Socrate e muritor.

Acesta e un *silogism* (tipar de regulă de inferență)
logica clasică: Aristotel, stoici

Seamănă cu *modus ponens*
dar premisa din (1) ("toți oamenii")
nu e la fel cu (2) (Socrate, un anumit om)

Logica propozițională nu poate exprima tot

Un exemplu clasic: (1) Toți oamenii sunt muritori.
(2) Socrate e om.
Deci, (3) Socrate e muritor.

Acesta e un *silogism* (tipar de regulă de inferență)
logica clasică: Aristotel, stoici

Seamănă cu *modus ponens*
dar premisa din (1) ("toți oamenii")
nu e la fel cu (2) (Socrate, un anumit om)

Am putea reformula (1): *Dacă X e om, atunci X e muritor*.
mai precis: *Pentru orice X, dacă X e om, atunci X e muritor*.

Logica modernă: *logica predicatelor* (logica de ordinul I)
Gottlob Frege, Charles Peirce (sec. 19)

Avem nevoie de formule mai expresive

Formulele sunt formate din *predicate* legate prin *conectori logici*

$$\forall x \; ((folder(x) \wedge x \neq root) \rightarrow contains(parent(x), x))$$

În loc de *propoziții* (a, p, q) avem *predicate*: $file(x)$, $contains(x, y)$

Un *predicat* = o afirmație relativ la una sau mai multe variabile, care, dând valori variabilelor, poate lua valoarea adevărat sau fals.

Avem nevoie de formule mai expresive

Formulele sunt formate din *predicate* legate prin *conectori logici*

$$\forall x \, ((\mathit{folder}(x) \wedge x \neq \mathit{root}) \rightarrow \mathit{contains}(\mathit{parent}(x), x))$$

În loc de *propoziții* (a, p, q) avem *predicate*: $\mathit{file}(x)$, $\mathit{contains}(x, y)$

Un *predicat* = o afirmație relativ la una sau mai multe variabile, care, dând valori variabilelor, poate lua valoarea adevărat sau fals.

Predicatele au argumente *termeni*: *variabile* x / *funcții*: $\mathit{parent}(x)$
intuitiv: reprezintă obiecte/noțiuni și funcții din univers

Nou: apar *cuantificatori* $\forall$ (orice), $\exists$ (există)

Definim *logica predicatelor* (*first-order logic*)
numită și *logica de ordinul I* (întâi)

Sintaxa logicii predicatelor: Termeni

Definim, structural recursiv, noțiunile de *termen* și *formulă*:

Termeni

variabilă v

Sintaxa logicii predicatelor: Termeni

Definim, structural recursiv, noțiunile de *termen* și *formulă*:

Termeni

variabilă v

$f(t_1, \cdots, t_n)$ cu f *funcție* n-ară și $t_1, \cdots, t_n$ *termeni*

Exemple: $parent(x)$, $cmmdc(x, y)$, $\max(\min(x, y), z)$

Sintaxa logicii predicatelor: Termeni

Definim, structural recursiv, noțiunile de *termen* și *formulă*:

Termeni

variabilă v

$f(t_1, \cdots, t_n)$ cu f *funcție* n-ară și $t_1, \cdots, t_n$ *termeni*

Exemple: $parent(x)$, $cmmdc(x, y)$, $\max(\min(x, y), z)$

constantă c: caz particular, funcție de zero argumente

Sintaxa logicii predicatelor: Formule

Formule (well-formed formulas, formule bine formate):

$P(t_1, \cdots, t_n)$ cu P *predicat* de n argum. și $t_1, \cdots, t_n$ *termeni*

Exemple: $contains(empty, x)$, $divide(cmmdc(x, y), x)$

propoziție p: caz particular, predicat de zero argumente

Sintaxa logicii predicatelor: Formule

Formule (well-formed formulas, formule bine formate):

$P(t_1, \cdots, t_n)$ cu P *predicat* de n argum. și $t_1, \cdots, t_n$ *termeni*

Exemple: $contains(empty, x)$, $divide(cmmdc(x, y), x)$

propoziție p: caz particular, predicat de zero argumente

$\neg\alpha$ unde α e o formulă

$\alpha \rightarrow \beta$ cu α, β formule

$\forall v\, \alpha$ cu v *variabilă*, α formulă: *cuantificare universală*

Exemple: $\forall x\ \neg contains(empty, x)$, $\forall x \forall y\ divide(cmmdc(x, y), x)$

Sintaxa logicii predicatelor: Formule

Formule (well-formed formulas, formule bine formate):

$P(t_1, \cdots, t_n)$ cu P *predicat* de n argum. și $t_1, \cdots, t_n$ *termeni*

Exemple: $contains(empty, x)$, $divide(cmmdc(x, y), x)$

propoziție p: caz particular, predicat de zero argumente

$\neg\alpha$ unde α e o formulă

$\alpha \to \beta$ cu α, β formule

$\forall v\, \alpha$ cu v *variabilă*, α formulă: *cuantificare universală*

Exemple: $\forall x\, \neg contains(empty, x)$, $\forall x \forall y\ divide(cmmdc(x, y), x)$

$t_1 = t_2$ cu t_1, t_2 termeni (în logica de ordinul I cu egalitate)

Exemplu: $\min(x, \min(y, z)) = \min(\min(x, y), z)$

Reprezentare în ML

Termenii și formulele se pot traduce direct în *tipuri recursive*

```
type term = V of string
          | F of string * term list

type predform = Pr of string * term list
              | Neg of predform
              | And of predform * predform
              | Or of predform * predform
              | Forall of string * predform
```

O formulă poate conține termeni. Termenii nu conțin formule!

Reprezentăm constantele ca funcții cu zero argumente.

Atât termenii cât și predicatele au argumente: listă de termeni.

Exemplu: $\forall x \, \neg \forall y \, P(x, f(y))$

```
Forall("x", Neg(Forall("y", Pr("P",[V "x"; F("f", [V "y"])]))))
```

Despre cuantificatori. Cuantificatorul existențial $\exists$

Notăm: $\boxed{\exists x\varphi \stackrel{def}{=} \neg\forall x(\neg\varphi)}$ φ formulă arbitrară

Există x pentru care φ e adevărată $\leftrightarrow$ nu pentru orice x φ e falsă.

Cei doi cuantificatori sunt *duali*. Putem scrie și $\forall x\varphi = \neg\exists x(\neg\varphi)$

Despre cuantificatori. Cuantificatorul existențial $\exists$

Notăm: $\boxed{\exists x\varphi \stackrel{def}{=} \neg\forall x(\neg\varphi)}$ φ formulă arbitrară

Există x pentru care φ e adevărată $\leftrightarrow$ nu pentru orice x φ e falsă.

Cei doi cuantificatori sunt *duali*. Putem scrie și $\forall x\varphi = \neg\exists x(\neg\varphi)$

Cuantificatorii au *precedență mai mare* decât conectorii $\neg$, $\wedge$, $\rightarrow$ ...
$\Rightarrow$ dacă formula cuantificată are $\wedge$, $\vee$, $\rightarrow$ folosim paranteze:
$\exists x(P(x) \rightarrow Q(x))$ $\forall y(Q(y) \wedge R(x, y))$

Despre cuantificatori. Cuantificatorul existențial $\exists$

Notăm: $\boxed{\exists x\varphi \stackrel{def}{=} \neg\forall x(\neg\varphi)}$ φ formulă arbitrară

Există x pentru care φ e adevărată $\leftrightarrow$ nu pentru orice x φ e falsă.

Cei doi cuantificatori sunt *duali*. Putem scrie și $\forall x\varphi = \neg\exists x(\neg\varphi)$

Cuantificatorii au *precedență mai mare* decât conectorii $\neg$, $\wedge$, $\rightarrow$ ...
$\Rightarrow$ dacă formula cuantificată are $\wedge$, $\vee$, $\rightarrow$ folosim paranteze:

$\exists x(P(x) \rightarrow Q(x))$ $\qquad$ $\forall y(Q(y) \wedge R(x, y))$

Altă notație: *punct .* cuantificatorul se aplică la tot restul formulei, până la sfârșit sau paranteză închisă

$P(x) \vee \forall y.Q(y) \wedge R(x, y)$ $\qquad$ $(R(y) \vee \exists x.P(x) \rightarrow Q(x)) \wedge S(x)$

În logica *de ordinul I* se pot cuantifica ($\forall$, $\exists$) doar variabile.
În logici *de ordin superior* (*higher-order*) se pot cuantifica și predicate.

Distributivitatea cuantificatorilor față de $\wedge$ și $\vee$

Cuantificatorul *universal* e *distributiv față de conjuncție*:

$$\forall x(P(x) \wedge Q(x)) \leftrightarrow \forall x\, P(x) \wedge \forall x\, Q(x)$$

dar cuantificatorul *existențial* NU ~~e distributiv~~ față de conjuncție:

$$\exists x(P(x) \wedge Q(x)) \not\leftrightarrow (\exists x\, P(x) \wedge \exists x\, Q(x))$$

avem implicație $\rightarrow$, dar nu și invers, poate să nu fie același x !

Distributivitatea cuantificatorilor față de $\wedge$ și $\vee$

Cuantificatorul *universal* e *distributiv față de conjuncție*:

$$\forall x(P(x) \wedge Q(x)) \leftrightarrow \forall x\, P(x) \wedge \forall x\, Q(x)$$

dar cuantificatorul *existențial* NU ~~e distributiv~~ față de conjuncție:

$$\exists x(P(x) \wedge Q(x)) \not\leftrightarrow (\exists x\, P(x) \wedge \exists x\, Q(x))$$

avem implicație $\rightarrow$, dar nu și invers, poate să nu fie același x !

Dual, $\exists$ e distributiv față de disjuncție:

$$\exists x\, P(x) \vee \exists x\, Q(x) \leftrightarrow \exists x. P(x) \vee Q(x)$$

$\forall$ nu e distributiv față de disjuncție. Avem doar:

$$\forall x\, P(x) \vee \forall x\, Q(x) \rightarrow \forall x. P(x) \vee Q(x)$$

Variabile legate și libere

În formula $\forall v \varphi$ (sau $\exists v \varphi$) variabila v se numește *legată*
Variabilele care nu sunt legate se numesc *libere*

Variabile legate și libere

În formula $\forall v \varphi$ (sau $\exists v \varphi$) variabila v se numește *legată*
Variabilele care nu sunt legate se numesc *libere*

O variabilă poate fi liberă și legată în aceeași formulă.

În $(\exists x.P(x) \rightarrow Q(x)) \wedge R(x)$, x e *legată* în $\exists x.P(x) \rightarrow Q(x)$
și e *liberă* în $R(x)$ (e în afara cuantificatorului)

Variabile legate și libere

În formula $\forall v \varphi$ (sau $\exists v \varphi$) variabila v se numește *legată*
Variabilele care nu sunt legate se numesc *libere*

O variabilă poate fi liberă și legată în aceeași formulă.

În $(\exists x . P(x) \to Q(x)) \wedge R(x)$, x e *legată* în $\exists x . P(x) \to Q(x)$
și e *liberă* în $R(x)$ (e în afara cuantificatorului)

Înțelesul unei formule *nu depinde* de variabilele legate
 înțelesul lor e "*legat*" de cuantificator ("pentru orice", "există")
 pot fi redenumite, fără a schimba înțelesul formulei
$(\exists x . P(x) \to Q(x)) \wedge R(x)$ la fel ca $(\exists y . P(y) \to Q(y)) \wedge R(x)$

O formulă *fără variabile libere* are înțeles de sine stătător.
(*closed formula*)

Analogie cu variabilele în program

Rol similar: parametrii formali la funcții în limbaje de programare
putem să îi redenumim fără a schimba efectul funcției
`fun x -> x + 3` și `fun y -> y + 3` sunt aceeași funcție

Interpretarea unei formule *depinde* de variabilele sale libere
(ce valoare din univers au; discutăm la semantica formulelor)

La fel și `fun x -> x + y`
înțelesul depinde de definiția lui `y` (presupus declarat anterior)

Formalizarea limbajului natural

Formulele conțin: *variabile*, *funcții*, *predicate*.

Formalizarea limbajului natural

Formulele conțin: *variabile*, *funcții*, *predicate*.

Verbele devin *predicate* (ca în limbajul natural):

$cumpără(X, Y)$, $scade(X)$,

Subiectul și *complementele* (in)directe: *argumentele* predicatului

Formalizarea limbajului natural

Formulele conțin: *variabile*, *funcții*, *predicate*.

Verbele devin *predicate* (ca în limbajul natural):
  cumpără(X, Y), *scade*(X),
Subiectul și *complementele* (in)directe: *argumentele* predicatului

Atributele (proprietăți) devin *predicate* despre valorile-argument
  bucuros(X), *de_aur*(Y)

Formalizarea limbajului natural

Formulele conțin: *variabile*, *funcții*, *predicate*.

Verbele devin *predicate* (ca în limbajul natural):
$\quad cumpără(X, Y)$, $scade(X)$,
Subiectul și *complementele* (in)directe: *argumentele* predicatului

Atributele (proprietăți) devin *predicate* despre valorile-argument
$\quad bucuros(X)$, $de\_aur(Y)$

Variabilele din formule pot lua valori *de orice fel* din *univers*
nu au un tip anume
$\Rightarrow$*Categoriile* devin tot *predicate*, cu argument obiectul de acel fel
$\quad copil(X)$, $caiet(X)$

Formalizarea limbajului natural

Formulele conțin: *variabile*, *funcții*, *predicate*.

Verbele devin *predicate* (ca în limbajul natural):
$cumpără(X, Y)$, $scade(X)$,
Subiectul și *complementele* (in)directe: *argumentele* predicatului

Atributele (proprietăți) devin *predicate* despre valorile-argument
$bucuros(X)$, $de\_aur(Y)$

Variabilele din formule pot lua valori *de orice fel* din *univers*
nu au un tip anume
$\Rightarrow$*Categoriile* devin tot *predicate*, cu argument obiectul de acel fel
$copil(X)$, $caiet(X)$

Entitățile *unice* devin *constante*:
ion, *emptyset*, *santaclaus*

Exemplu de formalizare (1)

1. *Fiecare investitor a cumpărat acțiuni sau obligațiuni.*

Exemplu de formalizare (1)

1. Fiecare investitor a cumpărat acțiuni sau obligațiuni.

Cuantificatorii introduc variabile cu valori *arbitrare* din univers
$\Rightarrow$ impunem categorii prin predicate suplimentare
$\Rightarrow$ introducem un predicat $inv(X)$ $\quad$ (X e investitor)

Pentru *orice* X, *dacă* X e investitor, a făcut ceva

$$\forall X . inv(X) \rightarrow \boxed{\text{ce face } X}$$

Exemplu de formalizare (1)

1. Fiecare investitor a cumpărat acțiuni sau obligațiuni.

Cuantificatorii introduc variabile cu valori *arbitrare* din univers
$\Rightarrow$ impunem categorii prin predicate suplimentare
$\Rightarrow$ introducem un predicat $inv(X)$ (X e investitor)

Pentru *orice* X, *dacă* X e investitor, a făcut ceva

$$\forall X.inv(X) \rightarrow \boxed{\textit{ce face } X}$$

Ce se spune despre investitor? *Există* ceva ce a cumpărat

$$\forall X.inv(X) \rightarrow \exists\, C.cumpără(X, C) \wedge \boxed{\textit{ce știm despre } C}$$

Exemplu de formalizare (1)

1. Fiecare investitor a cumpărat acțiuni sau obligațiuni.

Cuantificatorii introduc variabile cu valori *arbitrare* din univers
$\Rightarrow$ impunem categorii prin predicate suplimentare
$\Rightarrow$ introducem un predicat $inv(X)$ (X e investitor)

Pentru *orice* X, *dacă* X e investitor, a făcut ceva

$$\forall X.inv(X) \rightarrow \boxed{\textit{ce face } X}$$

Ce se spune despre investitor? *Există* ceva ce a cumpărat

$$\forall X.inv(X) \rightarrow \exists\, C.cumpără(X, C) \wedge \boxed{\textit{ce știm despre } C}$$

$$\forall X.inv(X) \rightarrow \exists\, C.cumpără(X, C) \wedge (acțiune(C) \vee oblig(C))$$

Sursa: `http://www.cs.utexas.edu/users/novak/reso.html`

Exemplu de formalizare (2)

2. Dacă indicele Dow Jones scade, toate acțiunile mai puțin aurul scad.

Indicele Dow Jones e o noțiune unică $\Rightarrow$ folosim o *constantă* dj
alternativ: puteam folosi și o *propoziție* $scadedj$

$$scade(dj) \rightarrow \boxed{\text{ce se întâmplă}}$$

$$scade(dj) \rightarrow \forall X.\ \boxed{\textit{condiții pentru } X} \rightarrow scade(X)$$

$$scade(dj) \rightarrow \forall X. acțiune(X) \wedge \neg aur(X) \rightarrow scade(X)$$

Exemplu de formalizare (3)

3. Dacă trezoreria crește dobânda, toate obligațiunile scad.

$$\mathit{creștedob} \rightarrow \forall X.\mathit{oblig}(X) \rightarrow \mathit{scade}(X)$$

Dobânda e unicul lucru din problemă care crește $\Rightarrow$ propoziție
alternativ: o constantă *dobânda* + predicat *crește*

Exemplu de formalizare (3)

3. Dacă trezoreria crește dobânda, toate obligațiunile scad.

$$crestedob \rightarrow \forall X.oblig(X) \rightarrow scade(X)$$

Dobânda e unicul lucru din problemă care crește $\Rightarrow$ propoziție
alternativ: o constantă *dobânda* + predicat *crește*

4. Orice investitor care a cumpărat ceva care scade nu e bucuros.

$$\forall X.inv(X) \rightarrow \boxed{\text{ce știm despre } X}$$

$$\forall X.inv(X) \rightarrow (\boxed{\text{condiție pentru } X} \rightarrow \neg bucuros(X))$$

$$\forall X.inv(X) \rightarrow (\exists\, C.cumpără(X, C) \wedge scade(C)) \rightarrow \neg bucuros(X)$$

Exemplu de formalizare (3)

3. Dacă trezoreria crește dobânda, toate obligațiunile scad.

$$\mathit{creștedob} \rightarrow \forall X.\mathit{oblig}(X) \rightarrow \mathit{scade}(X)$$

Dobânda e unicul lucru din problemă care crește $\Rightarrow$ propoziție
alternativ: o constantă *dobânda* + predicat *crește*

4. Orice investitor care a cumpărat ceva care scade nu e bucuros.

$$\forall X.\mathit{inv}(X) \rightarrow \boxed{\textit{ce știm despre } X}$$

$$\forall X.\mathit{inv}(X) \rightarrow (\boxed{\textit{condiție pentru } X} \rightarrow \neg \mathit{bucuros}(X))$$

$$\forall X.\mathit{inv}(X) \rightarrow (\exists\, C.\mathit{cumpără}(X, C) \land \mathit{scade}(C)) \rightarrow \neg \mathit{bucuros}(X)$$

$\rightarrow$ asociază la dreapta, $p \rightarrow q \rightarrow r = p \rightarrow (q \rightarrow r) = p \land q \rightarrow r$, echivalent:

$$\forall X.\mathit{inv}(X) \land (\exists\, C.\mathit{cumpără}(X, C) \land \mathit{scade}(C)) \rightarrow \neg \mathit{bucuros}(X)$$

Exemplu de formalizare (4)

5. Dacă indicele Dow Jones scade și trezoreria crește dobânda, toți investitorii bucuroși au cumpărat ceva acțiuni de aur.

$$scade(dj) \land creștedob \to \boxed{\textit{ce se întâmplă}}$$

$$scade(dj) \land creștedob \to \\ \forall X.inv(X) \land bucuros(X) \to \boxed{\textit{ce știm despre } X}$$

$$scade(dj) \land creștedob \to \\ \forall X.inv(X) \land bucuros(X) \to \exists C.cumpără(X, C) \land acțiune(C) \land aur(C)$$

Atenție la cuantificatori!

Cuantificatorul *universal* ("toți") cuantifică o *implicație*:

Toți studenții sunt tineri

Studenți $\subseteq$ *Tineri*

$\forall x.student(x) \rightarrow tânăr(x)$

Eroare frecventă: $\wedge$ în loc de $\rightarrow$: $\cancel{\forall x.student(x) \wedge tânăr(x)}$

Oricine/orice din univers e și student și tânăr!!!

Atenție la cuantificatori!

Cuantificatorul *universal* ("toți") cuantifică o *implicație*:

Toți studenții sunt tineri

Studenți $\subseteq$ *Tineri*

$$\forall x.student(x) \to tânăr(x)$$

Eroare frecventă: $\wedge$ în loc de $\to$: $\cancel{\forall x.student(x) \wedge tânăr(x)}$

Oricine/orice din univers e și student și tânăr!!!

Cuantificatorul *existențial* ("unii","există") cuantifică o *conjuncție*.

Există premianți studenți.

Premianți $\cap$ *Studenți* $\neq \emptyset$

$$\exists x.premiant(x) \wedge student(x)$$

Eroare frecventă: $\to$ în loc de $\wedge$: $\cancel{\exists x.premiant(x) \to student(x)}$

E adevărată dacă există un ne-premiant! (fals implică orice)

După traducerea în logică, putem demonstra!

Având o *infinitate de interpretări* (valori din univers, funcții, valori pentru relații/predicate), nu putem scrie tabele de adevăr.

După traducerea în logică, putem demonstra!

Având o *infinitate de interpretări* (valori din univers, funcții, valori pentru relații/predicate), nu putem scrie tabele de adevăr.

Putem face însă *demonstrații* (deducții) după *reguli de inferență* (pur sintactice), ca în logica propozițională.

Logica predicatelor e și ea *consistentă* și *completă*:

Orice teoremă e validă (adevărată în toate interpretările/atribuirile).

Orice formulă validă (tautologie) poate fi *demonstrată* (e teoremă).
dar dacă nu e validă, încercarea de a o demonstra (sau o refuta) poate continua la nesfârșit.

Demonstrația prin metoda rezoluției

O formulă e *validă* dacă și numai dacă *negația* ei e o *contradicție*.

Putem demonstra o teoremă prin *reducere la absurd*
arătând că *negația ei e o contradicție* (nerealizabilă).

Demonstrația prin metoda rezoluției

O formulă e *validă* dacă și numai dacă *negația* ei e o *contradicție*.

Putem demonstra o teoremă prin *reducere la absurd*
arătând că *negația ei e o contradicție* (nerealizabilă).

Fie ipotezele $A_1, A_2, \ldots, A_n$ și concluzia C.
Fie teorema

$$A_1 \wedge A_2 \ldots \wedge A_n \rightarrow C$$

adică: ipotezele $A_1, A_2, \ldots A_n$ implică împreună concluzia C

Demonstrația prin metoda rezoluției

O formulă e *validă* dacă și numai dacă *negația* ei e o *contradicție*.

Putem demonstra o teoremă prin *reducere la absurd*
arătând că *negația ei e o contradicție* (nerealizabilă).

Fie ipotezele $A_1, A_2, \ldots, A_n$ și concluzia C.
Fie teorema

$$A_1 \wedge A_2 \ldots \wedge A_n \rightarrow C$$

adică: ipotezele $A_1, A_2, \ldots A_n$ implică împreună concluzia C

Negația implicației: $\neg(H \rightarrow C) = \neg(\neg H \vee C) = H \wedge \neg C$

Deci arătăm că $A_1 \wedge A_2 \ldots \wedge A_n \wedge \neg C$ e o contradicție

(*reducere la absurd*: ipoteze adevărate+concluzia falsă e imposibil)

Arătăm că o formulă e o contradicție prin *metoda rezoluției*.

Rezoluția în calculul propozițional

Rezoluția e o *regulă de inferență* care produce *o nouă clauză* din două clauze cu *literali complementari* (p și $\neg p$).

$$\boxed{\frac{p \vee A \qquad \neg p \vee B}{A \vee B} \qquad \textit{rezoluție}}$$

"Din clauzele $p \vee A$ și $\neg p \vee B$ deducem/derivăm clauza $A \vee B$"

Rezoluția în calculul propozițional

Rezoluția e o *regulă de inferență* care produce *o nouă clauză* din două clauze cu *literali complementari* (p și $\neg p$).

$$\boxed{\frac{p \vee A \qquad \neg p \vee B}{A \vee B} \qquad \textit{rezoluție}}$$

"Din clauzele $p \vee A$ și $\neg p \vee B$ deducem/derivăm clauza $A \vee B$"

Reamintim: *clauză* = *disjuncție* $\vee$ de *literali* (propoziții sau negații)

Clauza obtinută = *rezolventul* celor două clauze în raport cu p

Exemplu: $rez_p(p \vee q \vee \neg r, \neg p \vee s) = q \vee \neg r \vee s$

Modus ponens poate fi privit ca un *caz particular de rezoluție*:

$$\frac{p \vee false \qquad \neg p \vee q}{false \vee q}$$

Rezoluția e o regulă validă

$$\frac{p \vee A \qquad \neg p \vee B}{A \vee B} \qquad \textit{rezoluție}$$

Rezoluția e o regulă de inferență *validă*:

$$\{p \vee A, \neg p \vee B\} \models A \vee B$$

orice atribuire care face premisele adevărate face și concluzia adevărată

Rezoluția e o regulă validă

$$\frac{p \vee A \qquad \neg p \vee B}{A \vee B} \qquad \textit{rezoluție}$$

Rezoluția e o regulă de inferență *validă*:

$$\{p \vee A, \neg p \vee B\} \models A \vee B$$

orice atribuire care face premisele adevărate face și concluzia adevărată

pentru $p = T$, trebuie să arătăm $B \models A \vee B$:

dacă $B = T$, atunci și $A \vee B = T$

simetric pentru $p = F$, deci regula e validă

Rezoluția e o regulă validă

$$\frac{p \vee A \qquad \neg p \vee B}{A \vee B} \qquad \textit{rezoluție}$$

Rezoluția e o regulă de inferență *validă*:

$$\{p \vee A, \neg p \vee B\} \models A \vee B$$

orice atribuire care face premisele adevărate face și concluzia adevărată

pentru $p = T$, trebuie să arătăm $B \models A \vee B$:

dacă $B = T$, atunci și $A \vee B = T$

simetric pentru $p = F$, deci regula e validă

Corolar: dacă $A \vee B$ e contradicție, la fel și $(p \vee A) \wedge (\neg p \vee B)$

dacă ajungem la contradicție, și formula inițială era contradicție

Exemplu de rezoluție (1)

$(a \vee \neg b \vee \neg d)$	b negat
$\wedge\ (\neg a \vee \neg b)$	b negat
$\wedge\ (\neg a \vee c \vee \neg d)$	
$\wedge\ (\neg a \vee b \vee c)$	b pozitiv

Exemplu de rezoluție (1)

$(a \vee \neg b \vee \neg d)$	b negat
$\wedge\ (\neg a \vee \neg b)$	b negat
$\wedge\ (\neg a \vee c \vee \neg d)$	
$\wedge\ (\neg a \vee b \vee c)$	b pozitiv

Luăm o propoziție cu ambele polarități (b) și construim rezolvenții

$rez_b(a \vee \neg b \vee \neg d,\ \neg a \vee b \vee c) = a \vee \neg d \vee \neg a \vee c = \mathrm{T}$

$rez_b(\neg a \vee \neg b,\ \neg a \vee b \vee c) = \neg a \vee \neg a \vee c = \neg a \vee c$

Exemplu de rezoluție (1)

$$\begin{array}{ll} \quad (a \vee \neg b \vee \neg d) & b \text{ negat} \\ \wedge\, (\neg a \vee \neg b) & b \text{ negat} \\ \wedge\, (\neg a \vee c \vee \neg d) & \\ \wedge\, (\neg a \vee b \vee c) & b \text{ pozitiv} \end{array}$$

Luăm o propoziție cu ambele polarități (b) și construim rezolvenții
$rez_b(a \vee \neg b \vee \neg d,\ \neg a \vee b \vee c) = a \vee \neg d \vee \neg a \vee c = \mathrm{T}$
$rez_b(\neg a \vee \neg b,\ \neg a \vee b \vee c) = \neg a \vee \neg a \vee c = \neg a \vee c$

Adăugăm noii rezolvenți (ignorăm T); eliminăm vechile clauze cu b

$$\begin{array}{l} \quad (\neg a \vee c \vee \neg d) \\ \wedge\, (\neg a \vee c) \end{array}$$

Exemplu de rezoluție (1)

$(a \vee \neg b \vee \neg d)$	b negat
$\wedge (\neg a \vee \neg b)$	b negat
$\wedge (\neg a \vee c \vee \neg d)$	
$\wedge (\neg a \vee b \vee c)$	b pozitiv

Luăm o propoziție cu ambele polarități (b) și construim rezolvenții
$rez_b(a \vee \neg b \vee \neg d,\ \neg a \vee b \vee c) = a \vee \neg d \vee \neg a \vee c = \mathrm{T}$
$rez_b(\neg a \vee \neg b,\ \neg a \vee b \vee c) = \neg a \vee \neg a \vee c = \neg a \vee c$

Adăugăm noii rezolvenți (ignorăm T); eliminăm vechile clauze cu b

$$(\neg a \vee c \vee \neg d)$$
$$\wedge (\neg a \vee c)$$

Nu mai putem crea rezolvenți. Nu avem clauza vidă.
$\Rightarrow$ formula e realizabilă, de exemplu cu $a = \mathrm{F}$. Sau cu $c = \mathrm{T}$.

Exemplu de rezoluție (1)

$$\begin{array}{ll} \quad (a \vee \neg b \vee \neg d) & b \text{ negat} \\ \wedge (\neg a \vee \neg b) & b \text{ negat} \\ \wedge (\neg a \vee c \vee \neg d) & \\ \wedge (\neg a \vee b \vee c) & b \text{ pozitiv} \end{array}$$

Luăm o propoziție cu ambele polarități (b) și construim rezolvenții
$rez_b(a \vee \neg b \vee \neg d,\ \neg a \vee b \vee c) = a \vee \neg d \vee \neg a \vee c = \mathrm{T}$
$rez_b(\neg a \vee \neg b,\ \neg a \vee b \vee c) = \neg a \vee \neg a \vee c = \neg a \vee c$

Adăugăm noii rezolvenți (ignorăm T); eliminăm vechile clauze cu b

$$\begin{array}{l} \quad (\neg a \vee c \vee \neg d) \\ \wedge (\neg a \vee c) \end{array}$$

Nu mai putem crea rezolvenți. Nu avem clauza vidă.
$\Rightarrow$ formula e realizabilă, de exemplu cu $a = \mathrm{F}$. Sau cu $c = \mathrm{T}$.

Pentru o atribuire suficientă ca să facă formula realizabilă, revenim la formula inițială, și dăm valori și lui b și/sau d.

Exemplu de rezoluție (2)

$$\begin{array}{ll} \quad a & \\ \wedge\ (\neg a \vee b) & \\ \wedge\ (\neg b \vee c) & \quad c \text{ pozitiv} \\ \wedge\ (\neg a \vee \neg b \vee \neg c) & \quad c \text{ negat} \end{array}$$

Exemplu de rezoluție (2)

$$\begin{array}{ll} \quad a & \\ \wedge (\neg a \vee b) & \\ \wedge (\neg b \vee c) & c \text{ pozitiv} \\ \wedge (\neg a \vee \neg b \vee \neg c) & c \text{ negat} \end{array}$$

Aplicăm rezoluția după c, avem o singură pereche de clauze:
$rez_c(\neg b \vee c,\ \neg a \vee \neg b \vee \neg c) = \neg b \vee \neg a \vee \neg b = \neg a \vee \neg b$

Exemplu de rezoluție (2)

$$
\begin{array}{ll}
\quad a & \\
\wedge (\neg a \vee b) & \\
\wedge (\neg b \vee c) & c \text{ pozitiv} \\
\wedge (\neg a \vee \neg b \vee \neg c) & c \text{ negat}
\end{array}
$$

Aplicăm rezoluția după c, avem o singură pereche de clauze:

$rez_c(\neg b \vee c,\ \neg a \vee \neg b \vee \neg c) = \neg b \vee \neg a \vee \neg b = \neg a \vee \neg b$

Eliminăm cele două clauze cu c și adăugăm clauza nouă:

$$
\begin{array}{l}
\quad a \\
\wedge (\neg a \vee b) \\
\wedge (\neg a \vee \neg b)
\end{array}
$$

Exemplu de rezoluție (2)

$$\begin{array}{ll} \quad a & \\ \wedge (\neg a \vee b) & \\ \wedge (\neg b \vee c) & \qquad c \text{ pozitiv} \\ \wedge (\neg a \vee \neg b \vee \neg c) & \qquad c \text{ negat} \end{array}$$

Aplicăm rezoluția după c, avem o singură pereche de clauze:
$rez_c(\neg b \vee c,\ \neg a \vee \neg b \vee \neg c) = \neg b \vee \neg a \vee \neg b = \neg a \vee \neg b$

Eliminăm cele două clauze cu c și adăugăm clauza nouă:

$$\begin{array}{l} \quad a \\ \wedge (\neg a \vee b) \\ \wedge (\neg a \vee \neg b) \end{array}$$

Aplicăm rezoluția după b:
$rez_b(\neg a \vee b, \neg a \vee \neg b) = \neg a \vee \neg a = \neg a$

Eliminăm cele două clauze cu b, adăugăm clauza nouă: $\begin{array}{l} \quad a \\ \wedge \neg a \end{array}$

Exemplu de rezoluție (2)

$$\begin{array}{ll} \quad a & \\ \wedge (\neg a \vee b) & \\ \wedge (\neg b \vee c) & \qquad c \text{ pozitiv} \\ \wedge (\neg a \vee \neg b \vee \neg c) & \qquad c \text{ negat} \end{array}$$

Aplicăm rezoluția după c, avem o singură pereche de clauze:
$rez_c(\neg b \vee c,\ \neg a \vee \neg b \vee \neg c) = \neg b \vee \neg a \vee \neg b = \neg a \vee \neg b$

Eliminăm cele două clauze cu c și adăugăm clauza nouă:

$$\begin{array}{l} \quad a \\ \wedge (\neg a \vee b) \\ \wedge (\neg a \vee \neg b) \end{array}$$

Aplicăm rezoluția după b:
$rez_b(\neg a \vee b, \neg a \vee \neg b) = \neg a \vee \neg a = \neg a$

Eliminăm cele două clauze cu b, adăugăm clauza nouă: $\begin{array}{l} \quad a \\ \wedge \neg a \end{array}$

Aplicăm rezoluția după a: $rez_a(a, \neg a) = \mathrm{F}$ (clauza vidă)
Deci formula inițială e o contradicție (e nerealizabilă).

Aplicarea rezoluției în calculul propozițional

Pornind de la o formulă în formă normală conjunctivă (CNF), *adăugăm rezolvenți*, încercând să *obținem clauza vidă:*

Alegem o propoziție p și adăugăm toți rezolvenții în raport cu p:
din m clauze cu p și n clauze cu $\neg p$, creăm $m \cdot n$ rezolvenți
am eliminat $p \Rightarrow$ ștergem cele $m+n$ clauze inițiale

Aplicarea rezoluției în calculul propozițional

Pornind de la o formulă în formă normală conjunctivă (CNF), *adăugăm rezolvenți*, încercând să *obținem clauza vidă:*

Alegem o propoziție p și adăugăm toți rezolvenții în raport cu p:
- din m clauze cu p și n clauze cu $\neg p$, creăm $m \cdot n$ rezolvenți
- am eliminat $p \Rightarrow$ ștergem cele $m+n$ clauze inițiale

Dacă vreun rezolvent e *clauza vidă*, formula e *nerealizabilă*

Dacă nu mai putem crea rezolvenți (literalii au polaritate unică), formula e *realizabilă* (facem T toți literalii rămași)

Numărul de clauze poate crește exponențial (problematic!)

Rezoluția: de la propoziții la predicate

În logica predicatelor, un *literal* nu e o propoziție, ci un *predicat*
nu doar p și $\neg p$, ci $P(arg1)$ și $\neg P(arg2)$ (argumente diferite)

Rezoluția: de la propoziții la predicate

În logica predicatelor, un *literal* nu e o propoziție, ci un *predicat*
nu doar p și $\neg p$, ci $P(arg1)$ și $\neg P(arg2)$ (argumente diferite)

Pentru a deriva o nouă clauză din $A \vee P(arg1)$ și $B \vee \neg P(arg2)$
trebuie să încercăm să aducem argumentele la o expresie comună.

Vom avea clauze cu variabile implicit cuantificate universal
pot lua orice valoare $\Rightarrow$ le putem *substitui* cu *termeni*

Rezoluția: de la propoziții la predicate

În logica predicatelor, un *literal* nu e o propoziție, ci un *predicat*
nu doar p și $\neg p$, ci $P(arg1)$ și $\neg P(arg2)$ (argumente diferite)

Pentru a deriva o nouă clauză din $A \vee P(arg1)$ și $B \vee \neg P(arg2)$ trebuie să încercăm să aducem argumentele la o expresie comună.

Vom avea clauze cu variabile implicit cuantificate universal
pot lua orice valoare $\Rightarrow$ le putem *substitui* cu *termeni*

Există o substituție care aduce predicatele la o formă comună?
ex. 1: $P(x, g(y))$ și $P(a, z)$
ex. 2: $P(x, g(y))$ și $P(z, a)$

În exemplul 1, substituind $x \mapsto a$, $z \mapsto g(y)$ obținem $P(a, g(y))$ și $P(a, g(y)) \Rightarrow$ am găsit o formă comună

În ex. 2 nu putem substitui *constanta* a cu $g(y)$ (a nu e variabilă)
g e funcție arbitrară, nu știm dacă există un y cu $g(y) = a$

Substituții și unificări de termeni

O *substituție* e o *funcție* care asociază unor *variabile* niște *termeni*:

$$\{x_1 \mapsto t_1,\ \ldots, x_n \mapsto t_n\}$$

Substituții și unificări de termeni

O *substituție* e o *funcție* care asociază unor *variabile* niște *termeni*:

$$\{x_1 \mapsto t_1,\ \ldots, x_n \mapsto t_n\}$$

Doi termeni se pot *unifica* dacă există o substituție care îi face egali

$f(x, g(y, z), t)\{x \mapsto h(z), y \mapsto h(b), t \mapsto u\} = f(h(z), g(h(b), z), u)$

Substituții și unificări de termeni

O *substituție* e o *funcție* care asociază unor *variabile* niște *termeni*:

$$\{x_1 \mapsto t_1,\ \ldots, x_n \mapsto t_n\}$$

Doi termeni se pot *unifica* dacă există o substituție care îi face egali
$f(x, g(y, z), t)\{x \mapsto h(z), y \mapsto h(b), t \mapsto u\} = f(h(z), g(h(b), z), u)$

Reguli de unificare

O *variabilă* x poate fi unificată cu orice *termen* t (substituție)
dacă x *nu apare* în t (altfel, substituind obținem un termen infinit)
deci nu: x cu $f(h(y), g(x, z))$

Doi *termeni* $f(\ldots)$ pot fi unificați doar dacă au aceeași funcție,
și *argumentele* (termeni) pot fi unificate (poziție cu poziție)

Două *constante* (funcții cu 0 arg.) $\Rightarrow$ unificate dacă sunt identice

Rezoluția în calculul predicatelor

Fie clauzele: A cu $P(\ldots)$ *pozitiv* și B, cu $\neg P(\ldots)$ (*negat*) Exemplu:

A: $P(x, g(y)) \vee P(h(a), z) \vee Q(z)$
B: $\neg P(h(z), t) \vee R(t, z)$

Rezoluția în calculul predicatelor

Fie clauzele: A cu $P(\ldots)$ *pozitiv* și B, cu $\neg P(\ldots)$ (*negat*) Exemplu:

A: $P(x, g(y)) \vee P(h(a), z) \vee Q(z)$
B: $\neg P(h(z), t) \vee R(t, z)$

Alegem niște (≥ 1) $P(\ldots)$ din A și niște $\neg P(\ldots)$ din B. aici: toți

Rezoluția în calculul predicatelor

Fie clauzele: A cu $P(...)$ *pozitiv* și B, cu $\neg P(...)$ (*negat*) Exemplu:

A: $P(x, g(y)) \vee P(h(a), z) \vee Q(z)$
B: $\neg P(h(z), t) \vee R(t, z)$

Alegem niște (≥ 1) $P(...)$ din A și niște $\neg P(...)$ din B. aici: toți

Redenumim variabilele comune (nu au legătură între A și B)
A: $P(x, g(y)) \vee P(h(a), z) \vee Q(z)$ B: $\neg P(h(z_2), t) \vee R(t, z_2)$

Rezoluția în calculul predicatelor

Fie clauzele: A cu $P(\ldots)$ *pozitiv* și B, cu $\neg P(\ldots)$ (*negat*) Exemplu:

A: $P(x, g(y)) \vee P(h(a), z) \vee Q(z)$
B: $\neg P(h(z), t) \vee R(t, z)$

Alegem niște (≥ 1) $P(\ldots)$ din A și niște $\neg P(\ldots)$ din B. aici: toți

Redenumim variabilele comune (nu au legătură între A și B)
A: $P(x, g(y)) \vee P(h(a), z) \vee Q(z)$ B: $\neg P(h(z_2), t) \vee R(t, z_2)$

Unificăm (toți odată) doar acei $P(\ldots)$ din A și $\neg P(\ldots)$ din B aleși

$\{P(x, g(y)), P(h(a), z), P(h(z_2), t)\}$ $x \mapsto h(a); z_2 \mapsto a; z, t \mapsto g(y)$

Rezoluția în calculul predicatelor

Fie clauzele: A cu $P(\ldots)$ *pozitiv* și B, cu $\neg P(\ldots)$ (*negat*) Exemplu:

A: $P(x, g(y)) \vee P(h(a), z) \vee Q(z)$
B: $\neg P(h(z), t) \vee R(t, z)$

Alegem niște (≥ 1) $P(\ldots)$ din A și niște $\neg P(\ldots)$ din B. aici: toți

Redenumim variabilele comune (nu au legătură între A și B)
A: $P(x, g(y)) \vee P(h(a), z) \vee Q(z)$ B: $\neg P(h(z_2), t) \vee R(t, z_2)$

Unificăm (toți odată) doar acei $P(\ldots)$ din A și $\neg P(\ldots)$ din B aleși
$\{P(x, g(y)), P(h(a), z), P(h(z_2), t)\}$ $x \mapsto h(a); z_2 \mapsto a; z, t \mapsto g(y)$

Eliminăm pe $P(\ldots)$ și $\neg P(\ldots)$ aleși din $A \vee B$. *Aplicăm substituția* rezultată *din unificare* și *adăugăm noua clauză* la lista clauzelor.

$$Q(g(y)) \vee R(g(y), a)$$

Păstrăm clauzele inițiale, se pot folosi cu alte alegeri de predicate.

Rezoluția: în concluzie

Generăm repetat clauze noi (*rezolvenți*) prin **rezoluție cu unificare**

Dacă repetând obținem *clauza vidă*, formula inițială e *nerealizabilă*.
Dacă *nu mai găsim rezolvenți noi*, formula inițială e *realizabilă*.

Rezoluția: în concluzie

Generăm repetat clauze noi (*rezolvenți*) prin **rezoluție cu unificare**

Dacă repetând obținem *clauza vidă*, formula inițială e *nerealizabilă*.
Dacă *nu mai găsim rezolvenți noi*, formula inițială e *realizabilă*.

Reamintim: am pornit încercând să demonstrăm

$$A_1 \wedge A_2 \wedge \ldots \wedge A_n \rightarrow C$$

prin *reducere la absurd*, negând concluzia și arătând că

$$A_1 \wedge A_2 \wedge \ldots \wedge A_n \wedge \neg C \qquad \text{e } \textit{contradicție}$$

Rezoluția: în concluzie

Generăm repetat clauze noi (*rezolvenți*) prin **rezoluție cu unificare**

Dacă repetând obținem *clauza vidă*, formula inițială e *nerealizabilă*.
Dacă *nu mai găsim rezolvenți noi*, formula inițială e *realizabilă*.

Reamintim: am pornit încercând să demonstrăm

$$A_1 \wedge A_2 \wedge \ldots \wedge A_n \rightarrow C$$

prin *reducere la absurd*, negând concluzia și arătând că

$$A_1 \wedge A_2 \wedge \ldots \wedge A_n \wedge \neg C \qquad \text{e } \textit{contradicție}$$

Metoda rezoluției e *completă* relativ la refutație
pentru orice formulă nerealizabilă, va ajunge la clauza vidă
dar nu poate *determina* realizabilitatea *oricărei formule*
(există formule pentru care rulează la infinit)

Exemplu de aplicare a rezoluției

Reluăm exercițiul formalizat anterior.
Folosim () și nu . pentru a evita greșeli la aplicarea cuantificării.

A_1: $\forall X(inv(X) \rightarrow \exists\, C(cump(X, C) \wedge (act(C) \vee oblig(C))))$

A_2: $scadedj \rightarrow \forall\, X(act(X) \wedge \neg aur(X) \rightarrow scade(X))$

A_3: $creștedob \rightarrow \forall\, X(oblig(X) \rightarrow scade(X))$

A_4: $\forall X(inv(X) \rightarrow (\exists\, C(cump(X, C) \wedge scade(C)) \rightarrow \neg bucur(X)))$

C: $scadedj \wedge creștedob \rightarrow$
$\forall\, X(inv(X) \wedge bucur(X) \rightarrow \exists\, C(cump(X, C) \wedge act(C) \wedge aur(C)))$

Exemplu de aplicare a rezoluției

Reluăm exercițiul formalizat anterior.
Folosim () și nu . pentru a evita greșeli la aplicarea cuantificării.

A_1: $\forall X(inv(X) \to \exists\, C(cump(X, C) \land (act(C) \lor oblig(C))))$

A_2: $scadedj \to \forall\, X(act(X) \land \neg aur(X) \to scade(X))$

A_3: $creștedob \to \forall\, X(oblig(X) \to scade(X))$

A_4: $\forall X(inv(X) \to (\exists\, C(cump(X, C) \land scade(C)) \to \neg bucur(X)))$

C: $scadedj \land creștedob \to$
$\forall\, X(inv(X) \land bucur(X) \to \exists\, C(cump(X, C) \land act(C) \land aur(C)))$

$\boxed{\text{Negăm concluzia \textit{la început}, înainte de a transforma cuantificatorii!}}$

$\neg C$: $\neg(scadedj \land creștedob \to$
$\forall\, X(inv(X) \land bucur(X) \to \exists\, C(cump(X, C) \land act(C) \land aur(C))))$

Eliminăm implicația, ducem negația până la predicate

1. *Eliminăm implicația*: $A \to B = \neg A \vee B$, $\neg(A \to B) = A \wedge \neg B$

Orice transformare *într-o formulă* NU afectează ce e în afara ei!
În $\forall x\ A$, transformând oricum *pe A* ($\to$, $\neg, \ldots$) NU se schimbă $\forall x$

Eliminăm implicația, ducem negația până la predicate

1. *Eliminăm implicația*: $A \to B = \neg A \vee B$, $\neg(A \to B) = A \wedge \neg B$

Orice transformare *într-o formulă* NU afectează ce e în afara ei!
În $\forall x\ A$, transformând oricum *pe A* ($\to$, $\neg$, ...) NU se schimbă $\forall x$

2. Ducem *$\neg$ înăuntru*: $\neg \forall x P(x) = \exists x \neg P(x) \quad \neg \exists x P(x) = \forall x \neg P(x)$

Eliminăm implicația, ducem negația până la predicate

1. *Eliminăm implicația*: $A \rightarrow B = \neg A \vee B$, $\neg(A \rightarrow B) = A \wedge \neg B$

Orice transformare *într-o formulă* NU afectează ce e în afara ei!
În $\forall x\ A$, transformând oricum *pe A* ($\rightarrow$, $\neg$, ...) NU se schimbă $\forall x$

2. Ducem *$\neg$ înăuntru*: $\neg \forall x P(x) = \exists x \neg P(x) \quad \neg \exists x P(x) = \forall x \neg P(x)$

A_1: $\forall X(inv(X) \rightarrow \exists\, C(cump(X, C) \wedge (act(C) \vee oblig(C))))$
$\quad \forall X(\neg inv(X) \vee \exists\, C(cump(X, C) \wedge (act(C) \vee oblig(C))))$

A_2: $scadedj \rightarrow \forall\, X(act(X) \wedge \neg aur(X) \rightarrow scade(X))$
$\quad \neg scadedj \vee \forall\, X(\neg act(X) \vee aur(X) \vee scade(X))$

A_3: $creștedob \rightarrow \forall\, X(oblig(X) \rightarrow scade(X))$
$\quad \neg creștedob \vee \forall\, X(\neg oblig(X) \vee scade(X))$

A_4: $\forall X(inv(X) \rightarrow (\exists\, C(cump(X, C) \wedge scade(C)) \rightarrow \neg bucur(X)))$
$\quad \forall X(\neg inv(X) \vee \neg \exists\, C(cump(X, C) \wedge scade(C)) \vee \neg bucur(X))$
$\quad \forall X(\neg inv(X) \vee \forall\, C(\neg cump(X, C) \vee \neg scade(C)) \vee \neg bucur(X))$

Eliminăm implicația, ducem negația înăuntru (cont.)

$\neg C$: $\neg(scadedj \wedge creștedob \rightarrow$
$\forall X(inv(X) \wedge bucur(X) \rightarrow \exists C(cump(X,C) \wedge act(C) \wedge aur(C))))$

$\neg C$: $scadedj \wedge creștedob \wedge$
$\neg\forall X(inv(X) \wedge bucur(X) \rightarrow \exists C(cump(X,C) \wedge act(C) \wedge aur(C)))$

$scadedj \wedge creștedob \wedge$
$\exists X(inv(X) \wedge bucur(X) \wedge \neg\exists C(cump(X,C) \wedge act(C) \wedge aur(C)))$

$scadedj \wedge creștedob \wedge$
$\exists X(inv(X) \wedge bucur(X) \wedge \forall C(\neg cump(X,C) \vee \neg act(C) \vee \neg aur(C)))$

Redenumim: nume unice la variabile cuantificate

3. Dăm *nume unice* variabilelor cuantificate în fiecare formulă, pentru a putea elimina ulterior cuantificatorii. De exemplu:

$\forall x\, P(x) \vee \forall x\, \exists y\, Q(x, y)$ devine $\forall x\, P(x) \vee \forall z\, \exists y\, Q(z, y)$

Redenumim: nume unice la variabile cuantificate

3. Dăm *nume unice* variabilelor cuantificate în fiecare formulă, pentru a putea elimina ulterior cuantificatorii. De exemplu:

$\forall x\, P(x) \vee \forall x\, \exists y\, Q(x, y)$ devine $\forall x\, P(x) \vee \forall z\, \exists y\, Q(z, y)$

Nu e nevoie în exemplul nostru:

A_1: $\forall X(\neg inv(X) \vee \exists\, C(cump(X, C) \wedge (act(C) \vee oblig(C))))$

A_2: $\neg scadedj \vee \forall\, X(\neg act(X) \vee aur(X) \vee scade(X))$

A_3: $\neg creștedob \vee \forall\, X(\neg oblig(X) \vee scade(X))$

A_4: $\forall X(\neg inv(X) \vee \forall\, C(\neg cump(X, C) \vee \neg scade(C)) \vee \neg bucur(X))$

$\neg C$: $scadedj \wedge creștedob \wedge$
$\exists\, X(inv(X) \wedge bucur(X) \wedge \forall\, C(\neg cump(X, C) \vee \neg act(C) \vee \neg aur(C)))$

Skolemizare: eliminăm cuantificatorii existențiali

4. *Skolemizare*: În $\forall x_1 ... \forall x_n \exists y$, alegerea lui y *depinde* de $x_1, \ldots x_n$; introducem o nouă *funcție Skolem* $y = g(x_1, \ldots, x_n)$, $\quad \exists y$ dispare

Skolemizare: eliminăm cuantificatorii existențiali

4. *Skolemizare*: În $\forall x_1 ... \forall x_n \exists y$, alegerea lui y *depinde* de $x_1, \ldots x_n$; introducem o nouă *funcție Skolem* $y = g(x_1, \ldots, x_n)$, $\quad \exists y$ dispare

A_1: $\forall X (\neg inv(X) \vee \exists\, C (cump(X, C) \wedge (act(C) \vee oblig(C))))$

Skolemizare: eliminăm cuantificatorii existențiali

4. *Skolemizare*: În $\forall x_1 ... \forall x_n \exists y$, alegerea lui y *depinde* de $x_1, \ldots x_n$; introducem o nouă *funcție Skolem* $y = g(x_1, \ldots, x_n)$, $\exists y$ dispare

A_1: $\forall X(\neg inv(X) \vee \exists\, C(cump(X, C) \wedge (act(C) \vee oblig(C))))$

C din $\exists$ depinde de $X \Rightarrow C$ devine *o nouă funcție* $f(X)$, $\exists C$ dispare

$\forall X(\neg inv(X) \vee (cump(X, f(X)) \wedge (act(f(X)) \vee oblig(f(X)))))$

Atenție! fiecare cuantificator $\exists$ primește o *nouă funcție* Skolem!

Skolemizare: eliminăm cuantificatorii existențiali

4. *Skolemizare*: În $\forall x_1 ... \forall x_n \exists y$, alegerea lui y *depinde* de $x_1, \ldots x_n$; introducem o nouă *funcție Skolem* $y = g(x_1, \ldots, x_n)$, $\exists y$ dispare

A_1: $\forall X(\neg inv(X) \vee \exists\, C(cump(X, C) \wedge (act(C) \vee oblig(C))))$

C din $\exists$ depinde de $X \Rightarrow C$ devine *o nouă funcție* $f(X)$, $\exists C$ dispare

$\forall X(\neg inv(X) \vee (cump(X, f(X)) \wedge (act(f(X)) \vee oblig(f(X)))))$

Atenție! fiecare cuantificator $\exists$ primește o *nouă funcție* Skolem!

Pentru $\exists y$ în *exteriorul* oricărui $\forall$, alegem o nouă *constantă Skolem*

Skolemizare: eliminăm cuantificatorii existențiali

4. *Skolemizare*: În $\forall x_1 ... \forall x_n \exists y$, alegerea lui y *depinde* de $x_1, \ldots x_n$; introducem o nouă *funcție Skolem* $y = g(x_1, \ldots, x_n)$, $\quad \exists y$ dispare

A_1: $\forall X(\neg inv(X) \vee \exists\, C(cump(X, C) \wedge (act(C) \vee oblig(C))))$

C din $\exists$ depinde de $X \Rightarrow C$ devine *o nouă funcție* $f(X)$, $\exists C$ dispare

$\forall X(\neg inv(X) \vee (cump(X, f(X)) \wedge (act(f(X)) \vee oblig(f(X)))))$

Atenție! fiecare cuantificator $\exists$ primește o *nouă funcție* Skolem!

Pentru $\exists y$ în *exteriorul* oricărui $\forall$, alegem o nouă *constantă Skolem*

$\neg C$: $scadedj \wedge creștedob \wedge \exists\, X(inv(X) \wedge bucur(X)$
$\qquad \wedge \forall\, C(\neg cump(X, C) \vee \neg act(C) \vee \neg aur(C)))$

Skolemizare: eliminăm cuantificatorii existențiali

4. *Skolemizare*: În $\forall x_1 ... \forall x_n \exists y$, alegerea lui y *depinde* de $x_1, \ldots x_n$; introducem o nouă *funcție Skolem* $y = g(x_1, \ldots, x_n)$, $\exists y$ dispare

A_1: $\forall X(\neg inv(X) \vee \exists\, C(cump(X, C) \wedge (act(C) \vee oblig(C))))$

C din $\exists$ depinde de $X \Rightarrow C$ devine *o nouă funcție* $f(X)$, $\exists C$ dispare

$\forall X(\neg inv(X) \vee (cump(X, f(X)) \wedge (act(f(X)) \vee oblig(f(X)))))$

Atenție! fiecare cuantificator $\exists$ primește o *nouă funcție* Skolem!

Pentru $\exists y$ în *exteriorul* oricărui $\forall$, alegem o nouă *constantă Skolem*

$\neg C$: $scadedj \wedge creștedob \wedge \exists\, X(inv(X) \wedge bucur(X)$
$\wedge \forall\, C(\neg cump(X, C) \vee \neg act(C) \vee \neg aur(C)))$

X devine o nouă *constantă* b (nu depinde de nimic), $\exists X$ dispare

$scadedj \wedge creștedob \wedge inv(b) \wedge bucur(b)$
$\wedge \forall C(\neg cump(b, C) \vee \neg act(C) \vee \neg aur(C))$

Forma normală prenex. Eliminăm cuantificatorii universali

5. Ducem *cuantificatorii universali în față*: *forma normală prenex*

A_4: $\forall X(\neg inv(X) \vee \forall C(\neg cump(X, C) \vee \neg scade(C)) \vee \neg bucur(X))$

$\forall X \forall C(\neg inv(X) \vee \neg cump(X, C) \vee \neg scade(C) \vee \neg bucur(X))$

Forma normală prenex. Eliminăm cuantificatorii universali

5. Ducem *cuantificatorii universali în față*: *forma normală prenex*

A_4: $\forall X(\neg inv(X) \vee \forall C(\neg cump(X, C) \vee \neg scade(C)) \vee \neg bucur(X))$

$\forall X \forall C(\neg inv(X) \vee \neg cump(X, C) \vee \neg scade(C) \vee \neg bucur(X))$

6. *Eliminăm cuantificatorii universali*
(devin impliciți, o variabilă poate fi înlocuită cu orice termen).

A_1: $\neg inv(X) \vee (cump(X, f(X)) \wedge (act(f(X)) \vee oblig(f(X))))$

A_2: $\neg scadedj \vee \neg act(X) \vee aur(X) \vee scade(X)$

A_3: $\neg creștedob \vee \neg oblig(X) \vee scade(X)$

A_4: $\neg inv(X) \vee \neg cump(X, C) \vee \neg scade(C) \vee \neg bucur(X)$

$\neg C$: $scadedj \wedge creștedob \wedge inv(b) \wedge bucur(b)$
$\wedge(\neg cump(b, C) \vee \neg act(C) \vee \neg aur(C))$

Forma clauzală

7. Ducem *conjuncția în exterior*ul disjuncției (distributivitate) și scriem fiecare clauză separat (*formă clauzală*, CNF)

(1) $\neg inv(X) \vee cump(X, f(X))$

(2) $\neg inv(X) \vee act(f(X)) \vee oblig(f(X)))$

(3) $\neg scadedj \vee \neg act(X) \vee aur(X) \vee scade(X)$

(4) $\neg creștedob \vee \neg oblig(X) \vee scade(X)$

(5) $\neg inv(X) \vee \neg cump(X, C) \vee \neg scade(C) \vee \neg bucur(X)$

(6) $scadedj$

(7) $creștedob$

(8) $inv(b)$

(9) $bucur(b)$

(10) $\neg cump(b, C) \vee \neg act(C) \vee \neg aur(C)$

Generăm rezolvenți până la clauza vidă

Căutăm predicate $P(...)$ și $\neg P(...)$ și unificăm, obținând rezolvenții:

(11) $\neg act(X) \vee aur(X) \vee scade(X)$ (3, 6)

(12) $\neg cump(b, C) \vee \neg act(C) \vee scade(C)$ (10, 11, $X = C$)

(13) $\neg oblig(X) \vee scade(X)$ (4, 7)

Când unificăm, redenumim clauzele să nu aibă variabile comune:

(13) $\neg oblig(Y) \vee scade(Y)$ vom unifica cu (2), redenumim X

(14) $\neg inv(X) \vee act(f(X)) \vee scade(f(X))$ (2, 13, $Y = X$)

(15) $\neg cump(b, f(X)) \vee \neg inv(X) \vee scade(f(X))$ (12, 14, $C = f(X)$)

(16) $\neg cump(b, C) \vee \neg scade(C) \vee \neg bucur(b)$ (5, 8, $X = b$)

(17) $\neg cump(b, C) \vee \neg scade(C)$ (9, 16)

(18) $\neg cump(b, f(X)) \vee \neg inv(X)$ (15, 17, $C = f(X)$)

(19) $\neg inv(b)$ (1, 18, $X = b$)

(20) $\emptyset$ (contradicție = succes în reducerea la absurd) (8, 19)

Rezumat

Putem traduce (*formaliza*) din limbaj natural în logica predicatelor

Putem *demonstra teoreme* prin reducere la absurd:

- negăm concluzia
- transformăm în *formă clauzală* (conjuncție $\wedge$ de disjuncții $\vee$)
- prin metoda *rezoluției* găsim o *contradicție* (clauza vidă)