# Curs 6

## Clauze Horn

O anume clasă de clauze a devenit importantă deoarece ea se bucură de propeietăți teoretice care ne permit să vedem deducția ca și o execuție a unei proceduri. Aceasta este așa numita clasă de clauze Horn.

Clauzele care conțin numai un literal se numesc clauze unitate. Clauzele care conțin cel mult un literal pozitiv se numesc clauze Horn 8.

În loc a se scrie clauzele Horn așa {L1, ¬L2, … , ¬Ln}, adesea le scriem L←L1 ∧ … ∧Ln

Pe când clauzele Horn care conțin un literal pozitiv se numesc clauze definite/program.

O clauză definită care este o clauză unitate se numește fapt și se scrie sub forma: L

Clauza vida este notată cu []. De ex formula ←q (clauza de interogare), q←p (clauza definite, iar p este un fapt)

Clauzele de forma ←L1 ∧ … ∧Ln, pot fi interpretate atat declarativ, cât și procedural. În mod declarativ, ele se citesc astfel:

* Dacă L1 și L2 sunt valabile, atunci L este de asemenea, valabilă, pe de altă parte, in mod procedural ele se interpretează asa: pentru a rezolva L, avem de rezolvat L1, L1, ... Ln
* În interpretarea procedurală, fiecare clauză definită sub forma L←L1∧ ... ∧ Ln, este văzută ca o definiție de procedură cu antetul L și corpul L1∧ ... ∧Ln. Procedurile sunt apelate prin interogări, care au forma: L←L1∧ ... ∧ Lm
* Și se compun din apelurile procedurilor L1 și ... și Lm.

Interpretarea procedurală este rațiunea prin care o mulțime de clauze se numește program (definit sau logic propozițional)

Programele definite logic au mai multe proprietăți. Una dintre acestea este așa numita proprietate de intersecție a modelelor.

Propoziție:

Fie F un program logic definit. Dacă I1 și I2 sunt modele ale uni F, atunci și I1 intersectat cu I2 este un model al unui F. (proprietate de intersecție a modelelor)

O consecință imediată a acestei teoreme este ca, pentru orice program logic definit F, există un cel mai mic model care este format din intersectia tuturor modelelor. Notăm acest model cu MF. (M indice F)

Teorema:

Fie F un program logic definit. MF este mulțimea formată din acei p aparțin FBp

Cu alte cuvinte, cel mai mic model al unui program definit este în mod sigur mulțimea variabilelor propoziționale care sunt consecințe logice ale programului. Mai mult, pentru a verifica dacă o formulă G provine dintr-un program logic propozițional F, este suficient numai să considerăm cel mai mic model MF al lui F.

Dacă MF este bine definit în G, atunci G este o consecință logică a lui F.

Întorcându-ne la exemplul de mai sus, fie:

F = {q←p,p} si AR = {p,q,r,s}

I1 = {p,q,r} si I2 = {p,q,s} (I1,I2 sunt 2 modele ale lui F, iar I1 intersectat cu I2 = {p,q})

Este usor sa vedem ca fiecare model al lui F poate contine atat p si q deoarece orice clauza intalnita in F ar fi falsă sub contradicția modelului cu presupunerea că I1 si I2 sunt modele, în consecință, MF={p,q} este cel mai mic model al lui F.

Cel mai mic model al unui program poate fi calculat prin intermediul functiei semnificative TF mapează prin interpretări pe interpretări și este definită astfel:

TF(I)={p apartine p←p1∧ … ∧pn apartin F si {p1, … , pn} inclus in I}

TF este continua pe laticea completa definite prin multimea de interpretari 2AR si pe submultimea de relații.

Fie TF↑0=fi și TF↑(n+1)=TF(TF↑n), oricare ar fi n >sau egal 0.

## Deducția naturală

Considerăm formula: (p∨(q∧r)→(p∨q)∧(p∨r) a cărei veridicitate trebuie stabilită.

O demonstratie matematică ar arăta astfel:

* Presupunem că p∨(q∧r) este o formulă valabilă. În plus, presupunem că p nu este valabilă. Atunci q∧r este valabilă iar, prin consecință, atât q, cât și r sunt valabile.
* Distribuind p în paranteză (eliminând acel p) conducem ca (p sau q) si (p sau r) sunt valabile
* Urmează faptul că și formula (p sau q) si (p sau r) este de asemenea valabilă.
* Ultima formulă a fost derivată din formula (p sau q) si (p sau r) care este de asemenea valabilă
* Acum putem conclude că formula (p sau (q si r)) implica (p sau q) si (p sau r), este valabilă
* prima regulă de invarianță spune că dacă este dată falsitatea atunci poate fi derivată orice formulă
* prima regulă de negație este legea contradicției
* a doua și a 3-a regulă de negație codifică așa numita reducere la absurd: dacă falsitatea poate fi derivată din presupunerea non F implica F, F trebuie să fie validă. Invers, dacă falsitatea poate fi derivată din presupunerea lui F, atunci non F trebuie să fie validă
* regula a doua de introducere pentru implicații de cod, codifică legea contrapoziției. Dacă non F poate fi derivată din presupunerea non G, atunci F implică G trebuie să fie validă
  + Prima regulă de eliminare pentru implicatii codifică modus pones
  + A 2-a regulă de eliminare pt implicatii codifică modus tollens
* Aceste reguli de inferență se utilizează pentru a genera deducții și demonstrații

O deducție din calculul deducției naturale este o secvență de formule posibil închise in casete închise/deschise, în așa fel încât fiecare element să fie:

* Axiomă
* O formulă care rezultă din elementele întâlnite anterior din casetele deschise în această etapă prin una din regulile de inferență
* O formulă diferită din axiomă ce nu rezultă dintr-una din regulile de inferență, caz în care formula se numește prezumție

Teoremă

Calculul propozițional al deducției naturale este robust și complet.

## Calculul secvențial

Teoremă

Calculul secvențial propozițional este robust și complet.

Calculul secvențial este punctul de plecare într-o arie de cercetare care se numește teoria demonstrației. De exemplu, Prolog-ul poate fi privit ca implementarea unui fragment al calculului secvențial, deși el nu a fost dezvoltat în acest fel.

## Rezoluția

* Descoperită în 1965 de Alan Robinson

Fie două clauze

C1={L1,L2, … ,Ln} și C2={non L, K1, … , Km}

Atunci multimea formata din {K1, … , Km, L1, … , Ln} se numeste rezolvanta lui C1 si C2.

Se poate observa că aceste clauze sunt mulțimi de literali. Deci, rezolvanta lui {p,q} si {non p,q} este clauza {q}.

Rezolvanta celor două clauze C1 și C2 este o consecință logică a lui {C1,C2}.

Principiul rezoluției poate fi utilizat pt a răta că o multime de clauze F este nesatisfiabilă prin rezolvante generate repetat până când se obține o clauză vidă.

Formalizat, o deducție a clauzei C din F este o secvență finită de clauze C1, ... , Cn=C, astfel incât, fiecare Ci (i de la 1 la k) sa fie o clauză din F sau o rezolvantă de clauze precedente Ci.

O deducție a clauzei vide [] din F rond (f de mana) este numită respingere.

Dacă o clauză vidă este derivată din F, atunci clauza vidă este o consecință logică a lui F.

Așadar, clauza vidă poate fi numai o consecință logică a unei multimi de clauze nesatisfiabile și, prin urmare, F însăși trebuie să fie nesatisfiabilă.

Teoremă

Calculul rezoluției pentru logica propozițiilor este robust și complet.

* Calculul rezoluției propozițiilor poate fi implementat direct în prolog
* Vom utiliza reprezentarea formulelor discutate și presupunem că formulele sunt în forma clauză
* În cele din urmă poate fi realizat, de ex, programul clause\_form prin utilizarea Prolog
* Așadar, o clauză este reprezentată printr-o listă de literali, iar o mulțime de clauze este reprezentată printr-o listă de liste