Logică și structuri discrete

Relații. Funcții parțiale

Marius Minea
marius@cs.upt.ro

http://www.cs.upt.ro/~marius/curs/lsd/

27 octombrie 2015

Relații în lumea reală și informatică

O relație (matematică) modelează *legătura* dintre două entități (posibil de *tip* diferit)

- relații subiect-obiect: un om a citit o carte
- relații umane: copil , părinte , prieten
- relații cantitative : egal, mai mic

Transpuse în informatică:

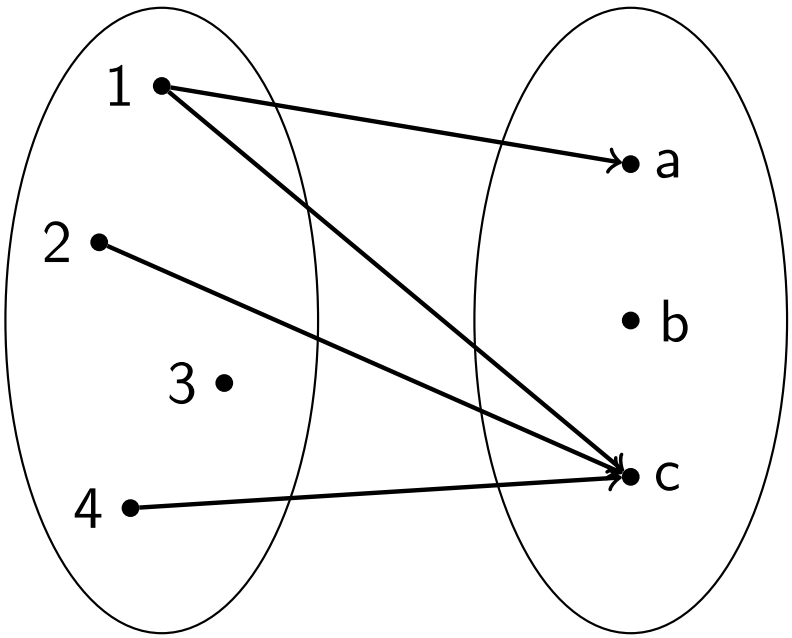
- rețele sociale : “prieten”, “follow”, “în cercuri”, etc.

O relație între elementele *aceleiași* mulțimi definește un *graf* (elementele sunt noduri, relația e reprezentată prin muchii)
$\Rightarrow$ relațiile sunt o noțiune cheie în teoria grafurilor

Relații: definiție și exemple

O *relație binară* R între două mulțimi A și B e o mulțime de perechi:
o *submulțime a produsului cartezian* $A \times B$: $\quad R \subseteq A \times B$

Notăm $(x, y) \in R$, sau $x \; R \; y$, sau $R(x, y)$ $\quad$ (x e în relație cu y)



$A = \{1, 2, 3, 4\}, \quad B = \{a, b, c\}$

$R = \{(1, a), (1, c), (2, c), (4, c)\}$

O relație e o noțiune *mai generală* decât o funcție: o funcție asociază *fiecărui* $x \in A$ *un singur* $y \in B$

Într-o relație putem avea:

1: are asociate *mai multe* elemente: a, c
2: are asociat *un singur* element: c
3: nu are asociat *niciun* element din B

Relații (cont.)

Generalizat, putem avea o relație *n-ară* care e o mulțime de n-tupluri (din produsul cartezian a n mulțimi).

Exemplu: $R \subseteq \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}$
$R(x, y, m)$ dacă m e un multiplu comun pentru a și b:

$R(2, 9, 18)$, $R(6, 9, 18)$, $R(2, 9, 36)$, etc.

În general, o relație nu e o noțiune simetrică:
produsul cartezian/perechea sunt noțiuni ordonate, $(x, y) \neq (y, x)$

Există, desigur, relații simetrice (în lumea reală și în matematică)

Reprezentarea unei relații

Explicit, prin mulțimea perechilor
dacă e finită
sau exprimabilă printr-o regulă care leagă elementele:

$$R = \{(x, x^2 + 1) \mid x \in \mathbb{Z}\}$$

.

În formă de matrice booleană / binară, pentru A, B finite,
cu liniile indexate după A, și coloanele după B
$m_{xy} = 1$ dacă $(x, y) \in R$, $m_{xy} = 0$ dacă $(x, y) \notin R$
reprezentare practică dacă A și B nu sunt foarte mari

Printr-o funcție de la A la $\mathcal{P}(B)$

Relația văzută ca funcție

O *relație* $R \subseteq A \times B$ poate fi văzută ca o *funcție* $f_R : A \to \mathcal{P}(B)$ de la A la mulțimea părților lui B:

$$f_R(x) = \{y \in B \mid (x, y) \in R\}$$

Asociază fiecărui x mulțimea (posibil vidă) a tuturor elementelor lui B cu care x e în relație.

Funcții parțiale

Dacă relaxăm condiția ca o funcție să asocieze o valoare *fiecărui* element, obținem o *funcție parțială* $f : A \nrightarrow B$.

O funcție parțială e un *caz particular de relație*:
asociază câte *un singur* element din B (ca funcția)
dar nu neapărat fiecărui element din A.

Ele sunt utile
– când domeniul exact al funcției nu e cunoscut
(funcții care nu sunt neapărat calculabile în orice punct).
în conjectura Collatz $(3 \cdot n + 1)$, numărul de pași până la 1
ar putea să nu existe (infinit) pentru anumiți n

– când domeniul de definiție al funcției e foarte mare sau nelimitat,
dar reprezentăm funcția explicit doar pentru valorile de interes
Exemplu: populația unei localități
posibil să nu știm populația pentru toate localitățile
dacă argumentul e un șir, nu orice șir e nume de localitate

Dicționare: funcții parțiale în ML

Un *dicționar* memorează *perechi* care asociază o *cheie* (primul element) cu o *valoare* (al doilea element)

În ML, folosim modulul `Map`, parametrizat (ca la mulțimi) cu un modul care definește tipul (ordonat) al cheilor

```
module M = Map.Make(String)
```
creează un modul care reprezintă asocieri între șiruri de caractere (cheia, de tip `string`) și un tip valoare încă neprecizat

Adăugând o pereche în asociere se instanțiază și tipul valorilor:

```
let m = M.add "x" 5 M.empty
```
adaugă la dicționarul vid o asociere de la șirul "x" la 5
echivalent: `let m = M.singleton "x" 5`
acum, `m` e un *dicționar* de la șiruri la întregi

Căutăm valoarea asociată unei chei în dicționar:

`M.find "x" m` returnează întregul 5
`M.find "y" m` generează *excepția* `Not_found`

Lucrul cu excepții

O *excepție* este o condiție specială care întrerupe calculul normal
dacă nu e *tratată*, se abandonează execuția programului
altfel, codul de *tratare a excepției* stabilește ce se face

Sintaxa:

`try` *expresie*

`with` *tipar*

unde *tipar* tratează una sau mai multe excepții și are forma

| *excepție-1* -> *expresie-exceptionala-1*
| *excepție-2* -> *expresie-exceptionala-2*

...

Dacă *expresie* se evaluează normal, ea dă rezultatul;
altfel, dacă apare *excepție-k* se evaluează ramura respectivă

```
fun x m -> try M.find x m
           with Not_found -> 0
```

Excepțiile se propagă, terminând fiecare funcție, până când sunt "prinse" de un bloc de tratare – altfel, programul e abandonat.

Relații cu ajutorul dicționarelor

Am văzut că o *relație* $R \subseteq A \times B$ poate fi privită ca o *funcție* $f_R : A \to \mathcal{P}(B)$ de la A la mulțimea părților lui B:

$$f_R(x) = \{y \in B \mid (x, y) \in R\}$$

Dicționarul va fi atunci de la A la *mulțimi* de elemente din B

```
module M = Map.Make(String) (* dictionar pe siruri *)
module S = Set.Make(String) (* multimea de valori *)

let addpair m (x, y) =
  let oldset = try M.find x m with Not_found -> S.empty
  in M.add x (S.add y oldset) m
(* creeaza dictionar din lista *)
let setmap_of_pairs = List.fold_left addpair M.empty

setmap_of_pairs [("tms", "arad");("tms", "lugoj")];;
```

asociază "tms" cu mulțimea {"arad", "lugoj"}

Relații binare pe o mulțime

Următoarele proprietăți sunt definite pentru relații binare pe *o* (aceeași) mulțime X: $R \subseteq X \times X$

reflexivă dacă pentru orice $x \in X$ avem $(x, x) \in R$

ireflexivă dacă pentru orice $x \in X$ avem $(x, x) \notin R$

simetrică dacă pentru orice $x, y \in X$,
dacă $(x, y) \in R$ atunci și $(y, x) \in R$

antisimetrică dacă pentru orice $x, y \in X$,
dacă $(x, y) \in R$ și $(y, x) \in R$, atunci $x = y$

tranzitivă, dacă pentru orice $x, y, z \in X$,
dacă $(x, y) \in R$ și $(y, z) \in R$, atunci $(x, z) \in R$

Relații de echivalență

O relație e *de echivalență* dacă e *reflexivă*, *simetrică* și *tranzitivă*

Relația de egalitate e (evident) o relație de echivalență.

Relația de congruență modulo un număr:
$a \equiv b \pmod{n}$ dacă $n \mid a - b$ (divide diferența)

O relație de echivalență pe X definește o *partiție* a lui X
în *clase de echivalență*

$$\hat{x} = \{y \mid (x, y) \in R\}$$

(clasa de echivalență a lui x e mulțimea elementelor aflate în relație cu x)

Relații de ordine stricte și totale

O relație $\prec$ e o *ordine strictă* dacă e *ireflexivă* și *tranzitivă*
nu există x cu $x \prec x$
dacă $x \prec y$ și $y \prec z$ atunci $x \prec z$

Exemple: relațiile $<$ și $>$ între numere (întregi, reale, etc.)
Relația "descendent" între persoane

O relație $\preceq$ e o *ordine totală* dacă e *reflexivă*, *antisimetrică* (dacă $x \preceq y$ și $y \preceq x$ atunci $x = y$), *tranzitivă*, și în plus
oricare două elemente sunt *comparabile*, adică pentru orice x, y avem $x \preceq y$ sau $y \preceq x$

Exemple: relațiile $\leq$ și $\geq$ între numere (întregi, reale, etc.)

Relații de ordine parțială

În practică apar adesea relații de ordine care nu sunt totale:

într-un campionat, după faza pe grupe, avem un clasament în fiecare grupă dar nu și între echipe din grupe diferite

avem o ordine între mesajele recepționate, dar nu neapărat între momentele trimiterii lor

în expresia $f(x) + g(x)$, f și g se apelează *înainte* de adunare, dar nu neapărat f înainte de g

O relație e o *ordine parțială* (non-strictă), dacă e *reflexivă*, *antisimetrică* și *tranzitivă*

Exemple din matematică:

relația de divizibilitate între întregi

relația de incluziune $\subseteq$ pe mulțimea părților

Orice ordine totală e și o ordine parțială (dar nu și reciproc).

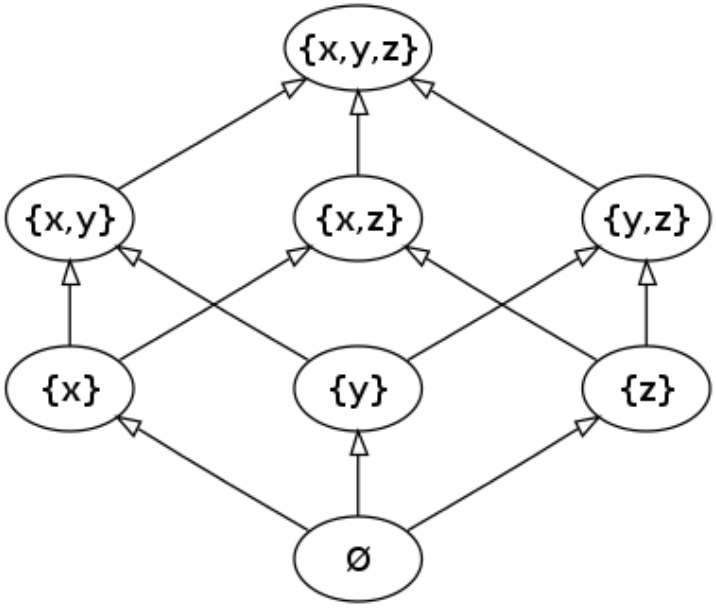
Orice ordine parțială induce o ordine strictă, și reciproc:

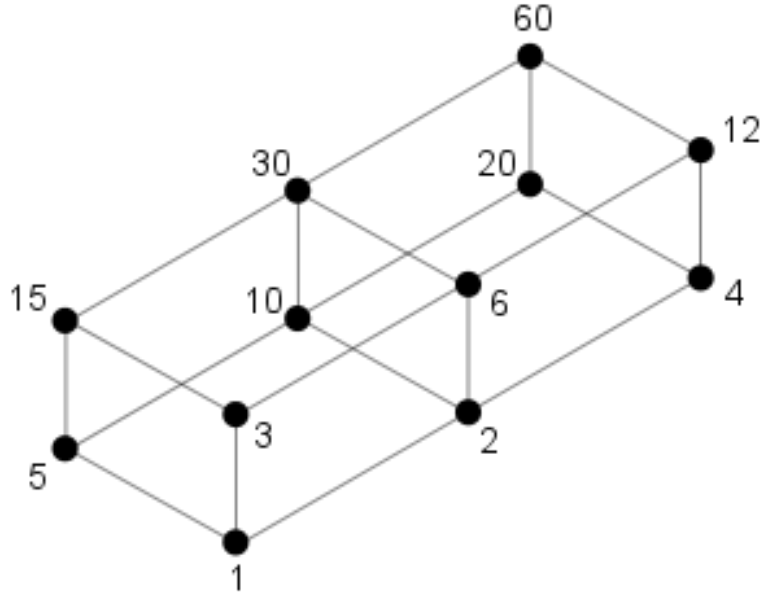
Definim: $a \prec b$ dacă $a \preceq b$ și $a \neq b$

Invers, definim $a \preceq b$ dacă $a \prec b$ sau $a = b$

Latice

O *latice* e o mulțime *parțial ordonată*, în care orice două elemente au un *minorant* și un *majorant*.
(elemente mai mici, respectiv mai mari în ordine decăt cele două).
Ex: mulțimea părților unei mulțimi (ordine: $\subseteq$; minor./maj.: $\cap$, $\cup$)
Ex: mulțimea divizorilor unui număr (c.m.m.d.c, c.m.m.m.c)





Imagine: http://en.wikipedia.org/wiki/File:Hasse_diagram_of_powerset_of_3.svg

http://en.wikipedia.org/wiki/File:Lattice_of_the_divisibility_of_60.svg

Latice completă

O latice L e *completă* dacă *orice* mulțime $S \subseteq L$ are un cel mai mic majorant (supremum) și un cel mai mare minorant (infimum).

Condiții mai puternice decât pentru o latice oarecare:

- nu doar un minorant și un majorant pentru orice două elemente
- *cel mai mic/mare* majorant/minorant pentru orice *submulțime* (chiar infinită)

$\Rightarrow$ Luând $S = L$, rezultă că L are un element minim și unul maxim

Exemplele din pagina anterioară sunt latice complete.

Noțiunea de punct fix

$x \in X$ e *punct fix* pentru funcția $f : X \to X$ dacă $f(x) = x$.
(privind f ca o transformare, ea nu îl modifică pe x)

Exemplu: fie un graf $G = (V, E)$, și pentru $X \subseteq V$ funcția

$$f(X) = X \cup \bigcup_{v \in X} \mathit{vecini}(v)$$

$f(U) = U \Rightarrow$ din nodurile U, urmărind vecinii nu găsim noduri noi

Pornind de la $s \in V$ și calculând $S_0 = \{s\}$, $S_1 = f(S_0)$, etc. când obținem un punct fix avem toate nodurile care pot fi atinse din S

Importanța: multe prelucrări (recursive/ciclice) pot fi definite ca transformări care se opresc când atingem un *punct fix*

care sunt toate configurațiile posibile într-un joc?
care sunt toate variabilele de care depinde o variabilă dată?
cu ce valoare poate ieși o variabilă dintr-un ciclu? etc.

Teorema de punct fix

Teoremă (Knaster-Tarski):

Fie f o funcție monotonă pe o latice completă.

Atunci mulțimea punctelor fixe a lui f e tot o latice completă.

Corolar:

O funcție monotonă pe o latice completă are un *cel mai mic punct fix* și *un cel mai mare punct fix*.

se obțin pornind de la 0, f(0), f(f(0)), ... resp. M, f(M), f(f(M)), ...

unde 0 și M sunt elementul cel mai mic respectiv cel mai mare

Punctul fix în ML

Dacă există limita șirului x, f(x), f(f(x)) ... putem scrie:

```
let rec fix f x =
  let y = f x in
  if y = x then x else fix f y
```

`fix f x` compară $f(x)$ cu x. Dacă sunt egale, x e punct fix,
și e returnat. Dacă nu, reapelăm recursiv cu valoarea $f(x)$.
Apelul al n-lea va avea argumentul $f^{n-1}(x)$ și îl compară cu $f^n(x)$.
Dacă există n cu $f^{n-1}(x) = f^n(x)$, va fi găsit, $f^{n-1}(x)$ e punct fix.

Putem rescrie, folosind o funcție ajutătoare cu doar un parametru
(nu mai trebuie repetat la apel parametrul `f`):

```
let fix f x =
  let rec fix1 x =
    let y = f x in if y = x then x else fix1 y
  in fix1
```

Inversa și compunerea de relații

Inversa unei relații $R \subseteq A \times B$ e relația $R^{-1} \subseteq B \times A$, cu $(y, x) \in R^{-1}$ dacă și numai dacă $(x, y) \in R$

$$R^{-1} = \{(y, x) \mid (x, y) \in R\}$$

Fie două relații $R_1 \subseteq A \times B$ și $R_2 \subseteq B \times C$.
Atunci compunerea $R_2 \circ R_1 \subseteq A \times C$ e relația

$$R_2 \circ R_1 = \{(x, z) \mid \text{ există } y \in B \mid (x, y) \in R_1 \text{ și } (y, z) \in R_2\}$$

La fel ca la funcții, scriem $R_2 \circ R_1$ și vedem că pentru $x \in A$ găsim întâi $y \in B$ și apoi $z \in C$.

Se poate vedea simplu că $(R \circ S)^{-1} = S^{-1} \circ R^{-1}$

Pentru o relație de echivalență, $R = R^{-1}$ (de ce? arătați!)

Dacă R e tranzitivă, $R \circ R \subseteq R$ (de ce? arătați!)

Pentru o relație binară $R \subseteq A \times A$, se notează $R^2 = R \circ R$, etc.

Închiderea tranzitivă a unei relații

Fie relația *copil(X, Y)* (X e copilul lui Y):
copil(ana, ion), *copil(lia, ion)*, *copil(ion, mara)*, *copil(mara, eva)*.

Definim relația *descendent*:
desc(X, Y) dacă *copil(X, Y)* (1)
desc(X, Z) dacă *desc(X, Y)* și *desc(Y, Z)* (2)

Relația *desc* include relația *copil* (1) și e tranzitivă (2).

Închiderea tranzitivă a unei relații $R \subseteq A \times A$ e relația *tranzitivă minimală* R^+ astfel ca $R \subseteq R^+$

Putem calcula $R^+ = \bigcup_{k=1}^{\infty} R^k = R \cup R^2 \cup \dots$

$R^2 = R \circ R$
$R^3 = R \circ R \circ R$

copil^2: *desc(ana, mara)*, *desc(lia, mara)*, *desc(ion, eva)*
copil^3: *desc(ana, eva)*, *desc(lia, eva)*.
Nu sunt descendenți de ordin > 3.

Sau, putem defini $f(X) = R \cup (X \circ R)$, atunci $f^n(R) = \bigcup_{k=1}^{n+1} R^k$.

Dar f e monotonă, deci are un punct fix minimal, care e chiar R^+ iar dacă X e finită, ajungem la limită într-un număr finit de pași