CHAPITRE : Relations binaires

Michaël PÉRIN – mises à jour Patrick LOISEAU

February 25, 2018

Contents

1 Motivation pour l'étude des relations

1. On peut redéfinir tous les concepts mathématiques à partir des ensembles ou bien à partir des relations.

Les relations sont donc une notion de base.

1. Les relations sont très utilisées en informatique pour organiser et structurer des connaissances.

Par exemple, les base de données relationnelles (cf. L3) sont au coeur de nombreux logiciels.

Amazon et Google utilise les relations et un algorithme de déduction pour faire de la publicité ciblée : ils étudient vos goûts et vous proposent des produits qui peuvent vous intéressés.

À partir de

- achete(VOUS , "DVD 1") qui représente « VOUS avez acheté DVD 1 »
- aussi("DVD 1","DVD 2") représente « ceux qui ont acheté X ont aussi acheté Y »

ils déduisent aimera(VOUS,"DVD 2") et vous suggère le DVD 2.

1. Les relations permettent de modéliser de nombreux problèmes de manière élégante (voir Projet de compilation INF124).

2 Relations binaires : définitions

Il y a plusieurs façons de parler d'une relation binaire R entre des éléments d'un ensemble A et les éléments d'un ensemble B

1. en mathématique, « a est en relation R avec b » se note a R b

   Exemple : a R b ssi a est un multiple de b

2. Définition par prédicat : une relation binaire R peut-être définie par un prédicat R : A x B –> Bool

   let R(a,b) = (a mod b = 0)

3. Définition ensembliste : Une relation binaire R est un sous-ensemble de l'ensemble produit AxB

   Rappel : l'ensemble A x B = { (a,b) | a∈ A, b∈ B } est l'ensemble de tous les couples (a,b) possibles.

   Une relation R sur AxB est une selection des certains de ces couples :

   R = { (a,b) | a∈ A, b∈ B, a mod b = 0 }

   Ici « a mod b = 0 » est le critère de sélection.

4. Représenation graphique : une relation binaire R est un graphe qui relie des points de A à des points de B par des arcs orientés

- L'arc a –> b est dans le graphe R
- équivaut à a R b
- équivaut à R(a,b) =true
- équivaut à (a,b)$\in$ R

5. Représentation informatique d'une relation binaire R (finie) :

   On représente R par un tableau (ou matrice) à deux dimensions de booléens

   x -R-> y ssi R[x][y] = true

   Remarque : c'est la représentation du prédicat (1) sous la forme d'un tableau

3 Propriétés classiques des relations binaires et interpétation sur les différentes représentations

Considérons une relation R sur ExE

3.0.1 R est réflexive si QQ x:E. x R x

- ie. le graphe de R contient de bouclettes sur chaque noeud
- ie. la diagonale du tableau R contient des 1

3.0.2 R est symétrique si QQ x,y:E. x R y ==> y R x

- ie. si le graphe de R contient un arc x–>y, il contient aussi l'arc x<–y
- ie. le tableau R est symétrique par rapport à la diagonale : R[x][y] = R[y][x]

3.0.3 R est transitive si QQ x,y,z:E. x R y /\ y R Z ==> x R z

3.0.4 R est anti-symétrique si QQ x,y:E. x R y /\ y R x ==> x=y

- ie. des noeuds x et y reliés par un double arc x <–> y doivent fusionner en un seul noeud (xy)

4 Relation d’ordre

4.0.1 Une relation R sur ExE est une relation d’ordre si R est réflexive, anti-symétrique, transitive

Exemple : prenez pour R la relation <= sur NxN

- réflexive: QQ n:N. n<=n
- anti-symétrique : QQ n,m:N, n<=m /\ m<=n ==> n=m
- transitive : QQ i,j,k:N. i<=j /\ j<=k => i < k

4.0.2 Une relation d’ordre sur ExE est totale si QQ x,y:E. x R y \/ y R x

Autrement dit, si on prend deux éléments de E on peut les comparer et dire lequels des deux est le plus grand.

4.0.3 Une relation d’ordre n’est pas forcément totale dans ce cas on dit qu’elle est partielle

Exemple : considérez la relation d’inclusion sur les sous-ensemble de {1,2,3}

1. Montrez que c’est une relation d’ordre
2. Montrez qu’elle est partielle solution: {1} inclus {1,2,3} mais {1,2} et {2,3} ne sont pas comparables : pas d’inclusion ni dans un sens, ni dans l’autre.
3. Construire le graphe complet de la relation d’ordre « est inclus dans » sur les sous-ensembles de {1,2,3}

4.0.4 Applications :

- projet de compilation INF124
- il existe des algorithmes pour construire un relation d’ordre à partir d’une relation donnée
- il sont utilisés dans de nombreux protocoles réseaux

5 Composition de relations binaires

Considérons deux relations binaires

- R sur A x B représentée par un tableau de booléens de dimension A x B tel que R[a][b] = 1 ssi a R b
- S sur B x C représentée par un tableau de booléens de dimension B x C tel que S[b][c]=1 ssi b S c

5.0.1 Définition intuitive de la composition

R relie des éléments de A a des élements de B et S relie des éléments de B à des éléments de C. On peut alors constuire la relation sur A x C qui relie directement des éléments de A à des éléments de C en composant les arcs de R et S.

A —R—> B —S—> C _ _ _ _ _ _ _ _ S o R _ _ _ _ _ _ _/ ou R ; S

La composition des relations binaires R:AxB et S:BxC est notée « S o R » On trouve aussi la notation « R;S » en informatique.

5.0.2 Définition mathématique : S o R = { (a,c) | il existe b:B. R[a][b] /\ S[b][c] }

Autrement dit, S o R est un l'ensemble des arcs a–>c tel qu'il existe un arc a-R->b et b-S->c

5.0.3 Construction de la composition de relations

La relation (S o R) est un tableau de dimension AxC

Pour le constuire il faut remplir le tableau case par case en appliquant la définition mathématique :

- (S o R)[a][c] = 1
- si et seulement si l'arc (a,c) apparient à SoR
- si et seulement si il existe b:B. R[a][b] /\ S[b][c]

Remarque:

- il existe x:{1,2,3} équivaut à x=1 \/ x=2 \/ x=3 équivaut à $\backslash/_{x}$:{1,2,3} qui se lit « disjonction sur tous les x appartenant à {1,2,3} »
- Le symbole « il existe » est la généralisation du \/ aux ensembles infinis

- En exploitant cette remarque on peut remplacer « il existe b:B » par « $\backslash/_{\mathrm{b}}$:B » puisque l'ensemble B est fini.

Finalement on obtient la formule [1] (S o R)[a][c] = $\backslash/_{\mathrm{b}}$:B (R[a][b] /\ S[b][c])

5.0.4 La composition de relation correspond à un produit de matrice

Regardons R et S comme de matrices et notons M = R*S = le produit des matrices R et S. Le coefficient m_{ac} de la matrice m est défini par

m_{ac} = Somme_{b}:B (r_{ab} * s_{bc})

Si, au lieu de la notation mathématique, m_{ac}, on reprend les notations informatiques, on obtient la formule

M[a][c] = Somme_{b}:B (R[a][b] * S[b][c])

Si on remplace « Somme par \/ » et « * par /\ » on retrouve la formule [1]

Conclusion S o R (composition de relation) = R*S (produit de matrices) où la somme est la disjonction et le produit est la conjonction.

5.0.5 Le produit de matrice compte le nombre de façons d'aller de «a» à «c» par SoR

Le produit de matrice est plus générale que la composition de relation binaire au sens où le produit de matrice considère systématiquement **tous les «b»** et l'influence qu'ils ont sur le résultat alors que pour la composition de relation il suffit de trouver **un «b»** tel que *a -R-> b -S-> c* pour conclure que le lien *a-SoR->c* existe.

Si on représente FAUX par 0 et VRAI par 1 (en Python, toute valeur différente de 0 représente VRAI) et Si on effectue le produit de matrices standart (où somme est l'addition et produit est la multiplication) entre deux relations binaires R et S représentée par des tableaux remplis de 0 et de 1

Alors on obtient un tableau SoR qui contient des entiers positifs

- 0 dans la case SoR[a][c] représente l'absence d'arc a -SoR-> c
- n dans la case SoR[a][c] représente l'existence d'arc a -SoR-> c et indique en plus qu'il existe *n* façons d'aller de «a» à «c»

[1] DEFINITION NOT FOUND.

autrement dit il y a *n* «b» possibles tels que *a* -*R*-> *b* -*S*->*c*

Ainsi, le produit de matrice ne se contente pas de savoir s'il existe une façon d'aller de «a» à «c» ; il compte le nombre de façon de le faire.

5.0.6 Généralisation du programme de composition

Au lieu d'écrire un **programme de composition de relations** spécialisé pour des tableaux de booléens, c'est-à-dire en utilisant (BOOL, FALSE, \/, TRUE, /\) on peut écrire un programme de produit de matrices qui utilise un semi-anneau générique défini par (RING,ZERO,somme,UNIT,produit) où :

- RING est un ensemble de valeurs (par exemple: RING = BOOL,N,Z,Q, ou bien R)
- ZERO est une constante qui représente l'élément neutre de l'opérateur somme ie. somme(ZERO,x) = x = somme(x,ZERO) pour tout x:RING
- UNIT est l'élément neutre de l'opérateur produit ie. produit(UNIT,x) = x = produit(x,UNIT) pour tout x:RING
- ZERO doit aussi être l'élément absorbant de l'opérateur produit ie. produit(ZERO,x) = ZERO = produit(x,ZERO) pour tout x:RING
- l'opération produit se distribue sur l'opération somme ie. produit(x,somme(y,z)) = somme(produit(x,y),produit(x,z)) pour tout x:RING

Si ces contraintes sont respectées alors (RING,ZERO,somme,UNIT,produit) forme un semi-anneau et le produit de matrices (qui s'appuie sur ces propriétés) est bien défini et le produit de matrices R*S correspond à la composition des effets de R et de S

5.0.7 Implantation générique en Python du produit de matrices

import numpy as np

def multiply(R, S): SoR = np.zeros(shape=(R.shape[2],S.shape[1])) for a in range(0,R.shape[2]): for c in range(0,S.shape[1]): s = ZERO for b in range(0,R.shape[1]): s = somme(s, produit(R[a][b],S[b][c])) SoR[a][c] = s return SoR

pour obtenir la composition de relations binaires il ne reste plus qu'à définir les constantes :

[2] DEFINITION NOT FOUND.

- ZERO = False
- UNIT = True

et les opérateurs :

- def somme(x, y): return x and y
- def produit(x, y): return x or y

6 Exercices sur les relations binaires

6.1 Exercice 1 :

Définir @p, @e, @g pour la relation @ sur l'ensemble {1,2,3,4,5}
définie comme suit n @ m `def` n<m et m est pair

6.2 Exercice 2 :

6.2.1 Construire le graphe de la relation d'ordre < sur {1,2,3,4}

6.2.2 Quelles sont les propriétés de cette relation et rappel des définitions et interprétations graphiques

- rélexive (bouclettes)
- transitive (chemin ==> racourci)
- symétrique (aller => retour)
- anti-symétrique (Piège : anti-symétrique / non symétrique) : forall x,y. xRy /\ yRx ==> x=y
- relation d'ordre (les noeuds du graphe sont séparables en couche) : reflexive, transitive, anti-symétrique
- relation d'ordre totale (un noeud par couche) : relation d'ordre, forall x,y. xRy \/ yRx

6.3 Exercice 3 :

Dessinez un relation sur {a,b,c,d} puis la compléter pour la rendre transitive, rélexive, symétrique

6.4 Exercice 4 :

Donnez tous les relations symétriques et anti-symétriques et transitives sur {a,b,c,d}

Combien y'en-a-t'il ?

6.5 Exercice

Dessinez une relation sur {a,b,c,d,e}

- reflexive,
- transitive
- symetrique,
- anti-symetrique

(variante : la plus petite relation)

6.6 Exercice

6.6.1 Comptez le nombre de relations possibles sur {e1,...,en}

indication : c'est choisir V/F pour chaque du tableau donc $2^{(n^2)}$

6.6.2 Est-il raisonnable de demander en examen de construire toutes les relations sur {a,b,...,z} ?

solution : à 1 sec le diagramme, on trouve $2^{(26^2)}$ sec c'est plus que l'age de l'univers !

6.6.3 Comptez les relations relfexives sur {e1,...,en}

(la diagonale est fixee donc n choix de moins a faire)

6.7 Exercice

Dessinez sur quelques mots la relation x–>y si et seulement si x est plus petit que y pour l'ordre alphabetique.

- est-ce une relation d'ordre strict ?

- est-ce une relation d'ordre totale ? (rappel de la definition) Remarque : placer les elements de haut en bas en fonction du nombre de fleches arrivant sur chaque mot. Ils sont classés !
- Trouver une representation simplifiee d'une relation d'ordre totale solution : tableau de dimension 1

6.8 Exercice

6.8.1 Modifier la relation suivante pour en faire une relation d'ordre (non strict) sur {a,b,c,d,e,f} avec R = { b>c ; b>a; d>c ; b >c ; c>f ; f>b }

on complete R pour la rendre reflexive, transitive et on fait attention a preserver l'anti symetrie. (R contient un cycle) donc il faut supprimer un arc.

6.8.2 Compléter R pour en faire une relation d'ordre totale.

on place les nombres de haut en bas en comptant le nombre de fleches entrantes.

on complete en mettant uniquement des fleches qui descendent (ainsi on est sur de ne pas creer de cycle)

pour certains elements on a le choix du sens de la fleche.

6.9 Exercice 5 :

Démonstration en DN de propriétés des relations :

INDICATION :

Commencez à faire la preuve en appliquant les définitions, quand vous êtes bloqués

faîtes des dessins pour déterminer quel candidat choisir pour le $\text{il}_{\text{existe}}$

1. Démontrez que R reflexive et S reflexive entraine RoS reflexive
2. formalisation en logique de l'énoncé
3. preuve en DN (arbre petit, faisable)

INDICATION : définition nécessaires pour faire l'exercice
R refl ———— (def refl) QQs x. xRx
IEx z xSz /\ zRy ——————- (def composition) x RoS y

1. Démontrez que R symétrique, S symétrique, RoS=SoR entrainent RoS symétrique

2. formalisation en logique de l'énoncé

3. preuve en DN (arbre de taille raisonnable)

INDICATION : définition nécessaires pour faire l'exercice
R1 = R2 ——————————- (def relation égales) QQs x,y. xR1y <==> xR2y

1. même exercice avec : R anti-symétrique, R transitive => RoR anti-symétrique