École normale supérieure de Lyon
Master 1 d'informatique fondamentale

Rapport de cours de recherche
Pierre Boutillier
17 avril 2009

Une classe d'ordres partiels : les treillis

Cours de Graphe

Table des matières

Introduction

Commençons au début, sans réinventer la roue, rappelons qu'une relation R sur un ensemble E est une partie de $E \times E$ et que l'on note couramment xRy pour signifier $(x, y) \in R$.

De même pour rafraîchir la mémoire, un ensemble munie d'une relation d'ordre est un couple $(E, \leq)$ tels que $\leq$ soit :

- réflexive : $\forall x \in E, x \leq x$
- transitive : $\forall (x, y, z) \in E^3, x \leq y \wedge y \leq z \rightarrow x \leq z$
- antisymétrique : $\forall (x, y) \in E^3, x \leq y \wedge y \leq x \rightarrow x = y$

Les relations d'ordres sont bien connues et largement étudiées en mathématiques. Par exemple, les relations d'ordres sur les ensembles de nombres sont centrales en analyse. Pourtant celles-ci sont bien spécifiques. Elles sont en effet définies sur des ensembles infinis. Cependant, elles sont surtout totales ($\forall (x, y) \in E^2, x \leq y \vee y \leq x$).

L'informatique a elle introduit des relations d'ordres partiels sur des ensembles finis dont l'étude s'avère tout aussi instructive puisqu'elles possèdent des propriétés ayant des conséquences pratiques.

En ordonnancement par exemple, les relations d'ordres entre processus sont massivement utilisées pour signifier lesquelles doivent être exécutées avant d'autres. Si cette relation d'ordre était totale, les machines parallèles auraient un avenir bien réduit. Mais des relations d'ordres peuvent s'avérer de très puissants outils dans des domaines plus inattendus.

Afin de présenter l'intérêt des études réalisées sur ces objets, nous regarderons dans un premier temps comment les manipuler. Nous verrons ensuite qu'une classe d'entre eux se révèle à la fois riche en propriétés et en applications concrètes. Nous présenterons ces cas. Enfin, nous étudierons comment dans le cas d'un exemple il est possible, à partir des définitions, d'obtenir les caractéristiques de ces objets par des manipulations efficaces.

1 Représentation et transformation des relations d'ordres [DP02]

Pour commencer, les relations d'ordres telles qu'elles sont définies dans leur généralité, ne sont ni propice à une représentation en machine ni optimisées pour en extraire les caractéristiques. Nous nous attellerons donc dans un premier temps à fournir une boite à outils pour les manipuler.

1.1 Stockage et visualisation d'une relation d'ordre

Une relation d'ordre est transitive. Cette propriété bien que fondamentale rend extrêmement redondante la manière naïve de représenter ou de stocker une de ces relations. Pour alléger cela, introduisons une nouvelle relation :

Définition 1 (Relation de couverture) *Soit P un ensemble partiellement ordonné, pour $(x, y) \in P^2$, x est couvert par y (noté $x \prec y$) ssi*

- $x \neq y$ *et* $x \leq y$
- $\forall z, x \leq z \leq y \rightarrow x = z \vee z = y$

Cela signifie intuitivement qu'il n'existe pas d'élément de P entre x et y.

(Dans $(\mathbf{N}, \leq)$, $n \prec m$ ssi $m = n + 1$, ou dans $(\mathcal{P}(E), \subseteq)$, $a \prec b$ ssi $b = a \cup \{x\}$ pour E fini et $x \in E$.)

La relation d'ordre d'un ensemble partiellement ordonné fini peut alors se représenter par le graphe orienté et acyclique (par anti-symétrie) dont les sommets sont les éléments de l'ensemble et les arêtes la relation de couverture associée à la relation d'ordre.

Remarquons au passage que le graphe ne contient pas de sous graphe égal à K_3 d'après la seconde caractéristique de $\prec$.

Ce graphe possède une représentation graphique relativement claire définie comme suit :

Propriété 1 (Diagramme de Hasse) *On associe aux sommets un point du plan et aux arêtes un segment entre les sommets concernés de telle manière que*

- *si $x \prec y$, le sommet x est en dessous de celui y.*
- *aucun sommet n'intersecte d'arête dont il n'est pas l'extrémité.*

Démonstration : Par induction facile sur la taille de l'ensemble en écartant les points correctement. □

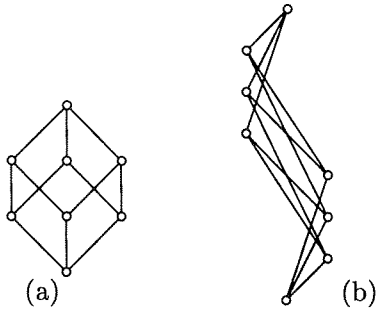

FIG. 1 – Deux diagramme de Hasse d'un même ordre

1.2 Transformations d'ensembles partiellement ordonnée

Afin de pouvoir plus facilement manipuler les ordres, il est parfois intéressant d'utiliser des fonctions dont on maîtrise mieux l'image tout en garantissant la conservation des propriétés qui nous importent.

Voici deux exemples de type de fonction qui peuvent nous être utiles dans ce but : Pour P et Q des ensembles munis d'une relation d'ordre et $\phi : P \longmapsto Q$ une application. ϕ est une

Définition 2 (Application croissante) *si $x \leq_P y$ implique que $\phi(x) \leq_Q \phi(y)$.*

Définition 3 (Plongement) *si $x \leq_P y$ équivaut à $\phi(x) \leq_Q \phi(y)$.*

Les plongements sont particulièrement intéressants car

– ce sont des applications injectives par antisymétrie donc la surjectivité équivaut à la bijectivité.
– elles sont stables par composition.
– ce sont les injections telles que $x \prec_P y$ ssi $\phi(x) \prec_Q \phi(y)$.

En informatique, les ensembles totalement ordonnés ont des représentations machine bien plus efficaces. L'usage de plongement peut donc se révéler particulièrement utile notamment pour stocker des objets partiellement ordonnés efficacement et en respectant leur ordre.

1.3 Éléments extremums

Avant de pouvoir se concentrer sur la sous-classe des relations d'ordres qui va nous intéresser, il faut introduire une dernière collection d'ensembles.

Avant cela, posons une définition qui va nous permettre de factoriser les notions :

Propriété 2 (Ordre dual) *On appelle ordre dual de $(E, \leq)$, $(E, \geq)$ qui est trivialement une relation d'ordre.*

De ce fait, toute propriété sur des éléments minorants de $\leq$ se transporte par passage au dual sur des éléments majorants.

Définition 4 (Idéal d'un ensemble partiellement ordonné) *Q est un idéal d'un ensemble partiellement ordonné ssi*

$$\forall x \in Q, \forall y \in P, y \leq x \rightarrow y \in Q$$

Pour A sous ensemble de P, l'idéal engendré par A sera noté $\downarrow A = \{y \in P | \exists x \in A, y \leq x\}$ et $\downarrow x$ représente $\downarrow \{x\}$.

Définition 5 (Filtre d'un ensemble partiellement ordonné) *Il s'agit de la notion dual de l'idéal.*

Q est un filtre d'un ensemble partiellement ordonné ssi

$$\forall x \in Q, \forall y \in P, x \leq y \rightarrow y \in Q$$

Pour A sous ensemble de P, le filtre engendré par A sera noté $\uparrow A = \{y \in P | \exists x \in A, x \leq y\}$ et $\uparrow x$ représente $\uparrow \{x\}$.

Comme leur nom le suggère les filtres et les idéaux d'un ensemble ordonné forment des structures algébriques étudiables en tant que telles. Dans ce rapport nous nous limiterons à l'utilisation des cas particuliers, centraux pour l'étude ultérieure.

Avant cela, soit

Définition 6 (Element maximal et plus grand élément) *x est un élément maximal ssi $\uparrow x = \{x\}$*

P admet un plus grand élément x ssi pour tout $y \in P$, $y \leq x$. x est dans ce cas l'unique élément maximal de P, Il pourra être noté $\top$.

Encore une fois, élément minimal et plus petit élément sont définis par dualité.

On défini donc des idéaux (respectivement des filtres) particuliers par :

Définition 7 (Majorants de $S \subset P$) *L'ensemble des majorants noté S^u est l'ensemble des éléments $x \in P$ tels que pour tout $y \in S$ $y \leq x$.*

Si S^u admet un plus petit élément, il est noté $\sup(S)$ *et appelé le plus petit majorant de S.*

Définition 8 (Minorants de $S \subset P$) *Par dualité toujours, $S^l = \{x \in P | \forall y \in S, x \leq x\}$*

Si S^l admet un plus grand élément, il est noté $\inf(S)$ *et appelé le plus grand minorant de S.*

2 Les treillis et treillis complets

Maintenant que les définitions sur les ordres ont été données, intéressons nous plus spécifiquement au cas d'une classe d'ordre qui a particulièrement intéressé les informaticiens. Cette classe se retrouve aussi bien pour obtenir des résultats en sémantique que pour extraire de l'information en fouille de données.

Regardons successivement deux manières équivalentes de les introduire puis analysons les résultats qui découlent de leur caractéristiques.

2.1 Définition par relation d'ordre

Définition 9 (Treilli) $(E, \leq)$ *est un treillis ssi pour tout* $(x, y) \in E^2$, $\sup(\{x, y\})$ *et* $\inf(\{x, y\})$ *existent*

Définition 10 (Treilli complet) $(E, \leq)$ *est un treillis complet ssi pour tout* $D \subset E$, $\sup(D)$ *et* $\inf(D)$ *existent*

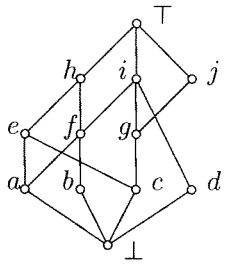

FIG. 2 – Le diagramme de Hasse d'un treillis complet non trivial

Tout de suite, il est remarquable que

Propriété 3 *Tout treillis sur un ensemble fini est en treillis complet*

Démonstration : Par induction sur la taille de l'ensemble grâce au lemme : soit P un treillis tel que pour $S \subset P$, $T \subset P$ tel que $\sup(P)$, $\sup(T)$, $\inf(P)$ et $\inf(T)$ existent alors $\sup(S \cup T) = \sup(P) \cup \sup(T)$ et $\inf(S \cup T) = \inf(P) \cup \inf(T)$ □

2.2 Définition algébrique

Plus original, il est possible de définir un treillis uniquement par des considération algébrique :

Définition 11 *Treillis un treillis est un ensemble E muni des deux lois internes $\vee$ et $\wedge$ telles qu'elles soient*

- *associatives :* $\forall(x,y,z) \in E^3, x \wedge (y \wedge z) = (x \wedge y) \wedge z$ *et* $\forall(x,y,z) \in E^3, x \vee (y \vee z) = (x \vee y) \vee z$
- *commutatives :* $\forall(x,y) \in E^2, x \wedge y = y \wedge x$ *et* $\forall(x,y) \in E^2, x \vee y = y \vee x$
- *idempotente :* $\forall x \in E, x \wedge x = x \vee x = x$
- *absorbantes :* $\forall(x,y) \in E^2, x \wedge (x \vee y) = x \vee (x \wedge y) = x$

Propriété 4 *Équivalence des définitions A partir de ces définitions, E en tant qu'ensemble ordonné est défini par $a \leq b$ ssi $a \wedge b = a$. Par transformation $a \leq b$ ssi $a \vee (a \wedge b) = b = a \vee b$.*

Les caractéristiques de relation d'ordre de /le se déduisent de celle de $\vee$ ou $\wedge$ puis et définissant $\inf(a,b) = a \wedge b$ *et* $\sup(a,b) = a \vee b$ *l'équivalence des définitions apparaît.*

Pour sortir des définitions formelles, prenons un cas précis. Pour E un ensemble fini, $(\mathcal{P}(E), \subseteq)$ est un treillis avec

- $\forall(A,B) \in \mathcal{P}(E)^2, A \vee B = A \cup B$
- $\forall(A,B) \in \mathcal{P}(E)^2, A \wedge B = A \cap B$

2.3 Propriétés fondamentales

Cette classe d'ordre se retrouve dans de nombreux cas concret mais surtout possède des caractéristiques particulièrement puissantes :

2.3.1 Diagramme de Hasse d'un treillis

Il y a deux éléments dans la définition d'un treillis qui permettent d'en caractériser le diagramme de Hasse. L'existence de $\sup(\{a,b\})$ (resp.$\inf(\{a,b\})$ impose :

1. que $\{a,b\}^u$ soit non vide. C'est pour cela que le diagramme a la forme de "diamant" qui correspond à l'intuition.
2. mais aussi et c'est plus subtil que $\{a,b\}^u$ est un plus petit élément. Ce qui signifie que $\{a,b\}^u$ à son tour est un treillis. C'est d'ailleurs pour cela que ces objets ont ce nom là.

2.3.2 Plongement sur un treillis

Propriété 5 (Théoreme du point fixe de Knaster-Tarski) *Soit L un treillis complet et Φ un plongement de L dans L :*

$$\Phi \text{ admet un point fixe défini par } \sup(\{x \in L | x \leq \Phi(x)\})$$

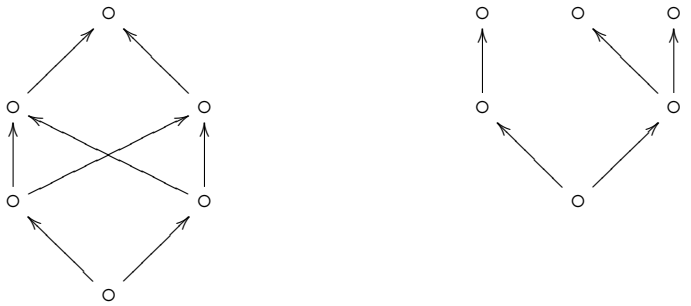

FIG. 3 – Sous-structure interdite d'un diagramme de Hasse d'un treillis

Démonstration : Soit $H = \{x \in L | x \leq \Phi(x)\}$ et $\alpha = \sup(H)$

1. Pour tout $x \in H$, par définition $x \leq \alpha$ puis comme Φ est un plongement et $x \in H$ $x \leq \Phi(x) \leq \Phi(\alpha)$ donc $\Phi(\alpha) \in H^u$ d'où α et $\Phi(\alpha)$ sont comparable et $\alpha \leq \Phi(\alpha)$.
2. Comme Φ conserve l'ordre, $\alpha \leq \Phi(\alpha)$ donc $\Phi(\alpha) \leq \Phi(\Phi(\alpha))$ ainsi $\Phi(\alpha)/inH$. C'est pourquoi $\Phi(\alpha) \leq \alpha$.

Par conséquent $\Phi(\alpha) = \alpha$, Φ admet α comme point fixe. □

Ce théorème est fondamental car il donne l'existence et même la caractérisation des fonctions récursives en programmation.

L'ensemble des fonctions définies sur une sous partie d'un ensemble fini est un treillis pour la relation suivante : "avoir le même graphe sur l'ensemble de définition commun". Il est donc par exemple possible de définir une chaîne de fonction sur les entiers par $f_0(0) = 1$ puis $f_k = f_{k-1}$ sur $[0, k-1]$, $f_k(k) = k \times f_{k-1}(k-1)$ et alors le théorème de Knaster-Tarski nous dit que la fonction factorielle appartient au langage et que son expression sous forme récursive est valide.

3 Les treillis de concepts

3.1 Fouille de donnée et concept

Informellement mais mathématiquement, un concept est un regroupement de protagonistes et un groupe de caractères partagés. Concrètement, "couleur des yeux des humains" est un concept de même que "type de processeur pour des ordinateurs" ou "nombre de satellites pour des planètes".

Formellement cette fois, l'ensemble des concepts est représenté par les couples "ensemble de protagonistes", "ensemble de caractères" qu'ils partagent. Il est possible de le munir de la relation d'ordre "être plus générale" comme suit :

$$(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2 \Leftrightarrow B_2 \subseteq B_1$$

Un concept peut enfin s'exprimer par le graphe bipartie (G, M, I) avec G les objets, M les attributs et $I \subset G \times M$ représentant la possession de l'attribut pour l'objet.

Pour $A \subset G$ et $B \subset M$, soient $A' = \{m \in M | \forall g \in A, (g, m) \in I\}$ et $B' = \{g \in G | \forall m \in A, (g, m) \in I\}$. Les concepts sont les couples (A, B) tels que $A' = B$ et $B' = A$.

Ils forment une classe particulière de treillis complets appelée *treillis de Galois* où les protagonistes forment les éléments médiants et sont d'autant plus similaires que leur plus petit minorant est loin de $\top$ (respectivement proche d'eux-mêmes).

Construire le treillis à partir des données des caractéristiques de chacun des protagonistes permet donc de détecter les agents similaires en fouille de données. Présentons un moyen efficace de l'obtenir.

3.2 Les construire et en extraire les caractéristiques efficacement [NR99]

La première étape est d'obtenir tous les concepts à partir des caractéristiques de chacun des protagonistes. Pour cela on réalise ce qui s'appelle l'arbre lexicographique de l'ensemble.

La seconde est de déduire de l'arbre lexicographique le graphe recouvrant représentant le treillis des sous-concepts.

Afin de fixer les notations,

- l'ensemble des objets est noté X
- l'ensemble des caractéristiques est noté B
- la relation "avoir la caractéristique" est elle notée E.

Au départ, le principe est de construire F l'ensemble des caractéristiques partageables et $\gamma(f)$ pour tout $f \in F$ l'ensemble des ensembles de caractéristiques d'objets qui constitue f. Pour ce faire, on construit itérativement les F_i ensembles des ensembles de caractéristiques partageables par les i premiers objets. Voici le détail :

Algorithm 1 Tree B

Require: B ensemble des caractéristiques
Ensure: l'arbre lexicographique de B

```
F ⇐ {∅}
for all b ∈ B do
  for all f ∈ F do
    f' ⇐ f ∪ b
    if f' ∉ F then
      F ⇐ F ∪ f
    end if
    γ(f') ⇐ γ(f) ∪ {b}
  end for
end for
```

Par décompte du nombre de boucles, cette étape a un coût $O((|X| + |B|)|B| \cdot |F|)$ en temps et $O((|X| + |B|)|F|)$ en espace.

La seconde partie de l'algorithme s'appuie sur une caractérisation de $\prec$ dans notre cas :

Propriété 6 *Soit f et f' dans F tels que $f \subset f'$*

$$f \prec f' \Leftrightarrow \forall (b_1, b_2) \in \gamma(f') \setminus \gamma(f), B_1 \setminus f = B_2 \setminus f$$

ce qui donne le corollaire :

$$f \prec f' \Leftrightarrow \forall b \in \gamma(f') \setminus \gamma(f), f' = f \cup b$$

L'algorithme calculera ainsi les éléments candidats à la couverture de chaque élément par un parcours de l'arbre lexicographique. Ensuite, il décrétera qu'ils sont éléments de la couverture si ils les rencontre $|\gamma(f')\setminus\gamma(f)|$ fois. C'est la fonction de la variable $COUNT(f)$ pour $f \in F$.

Précisément, voici la démarche :

Algorithm 2 Covering-Graph F

Require: arbre lexicographique des $(f, \gamma(f))$
Ensure: Liste d'adjacence du diagramme de Hasse du treillis
  $COUNT(f) \Leftarrow 0$ pour tout $f \in F$
  for all $f \in F$ **do**
    for all $b \in B \setminus \gamma(f)$ **do**
      $f' \Leftarrow f \cup b$
      $COUNT(f') + +$
      if $|\gamma(f')| = COUNT(f') + |\gamma(f)|$ **then**
        $Succ(f) \Leftarrow Succ(f) \cup \{f'\}$
      end if
    end for
    réinitialise COUNT
  end for

Ce second algorithme est réalisé aussi en $O((|X| + |B|) \cdot |B| \cdot |F|)$. Le coût total de l'algorithme est par conséquent en $O((|X| + |B|) \cdot |B| \cdot |F|)$.

Cet algorithme de calcul de clôture peut avoir plusieurs autres application comme :

- Calculer en $O(|X|^2 \cdot F)$ le complété de Dedekind-MacNeille d'un ordre $P = (X, \leq)$ qui est le plus petit treillis complet contenant P comme sous-ordre.
  Il est défini comme l'ensemble des couples (A, B) de sous ensemble de X tel que $A^u = B$ et $B^l = A$ muni de l'ordre $(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2 \Leftrightarrow B_2 \subseteq B_1$
- Calculer le treillis des idéaux d'un ordre avec la même complexité
- Obtenir l'antichaîne maximum pour un ordre partiel donné par sa relation de couverture

Conclusion

Dans ce rapport, nous avons essayé de montrer que les relations d'ordres partiels sur des ensembles finis possédaient des propriétés intéressantes. Nous avons exhibé des situations où il est commode d'utiliser le formalisme des ordres aussi bien en informatique qu'en sociologie. Des ordres particuliers ainsi définis on peut extraire des caractéristiques importantes sur les objets qu'ils représentent.

Nous nous sommes surtout concentrés à définir des outils et des représentations des relation d'ordres sur des ensembles finis qui permette de les étudier et de les visualiser bien plus simplement.

Enfin par le parallèle que nos définitions ont posé entre les ordres et leur représentabilité sous forme de graphe, nous avons, comme souvent en mathématique, ouvert le champs des résultats sur ceux-ci en utilisant les propriétés de ceux-là.

La factorisation d'informations à partir de très grands graphes, et l'apport que peut fournir dans cette optique la vision ordre et treillis, n'en est sûrement qu'à ses balbutiements car elle est au coeur de la problématique d'étude des systèmes complexes.

Références

[DP02] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, April 2002.

[NR99] Lhouari Nourine and Olivier Raynaud. A fast algorithm for building lattices. *Inf. Process. Lett.*, 71(5-6) :199–204, 1999.