
UFR Mathématiques et Informatique

LORIA


Master 2 Sciences Cognitives et Médias Numériques

---

# Graphes médians, FCA et phylogénie

---


*Étudiant :*
Baptiste Mounier

*Encadrants :*
Miguel Couceiro
Alain Gely
Amedeo Napoli

*Référent universitaire :*
Christine Bourjot


*Jury :*
Christine Bourjot
Miguel Couceiro
Manuel Rebuschi
Azim Roussanaly

*Date :*
09/04/18 - 08/08/18

# Résumé

L'étude de filiation des espèces, la phylogénie, est un domaine de recherche utilisant des arbres parcimonieux. Par le passé, H.J. Bandelt [1][2] a mis en avant l'avantage qu'il y a à encoder l'ensemble des informations dans un unique graphe médian regroupant l'ensemble de ces arbres phylogénétiques. Puis ce fut au tour d'U. Priss [3] de présenter l'avantage de cette vision qui permettrait de donner à la phylogénie accès aux méthodes de l'analyse de concepts formels. De plus elle donne un point de départ de méthode permettant afin de passer d'un treillis construit à partir de la matrice de données source des arbres, en un graphe médian. Ce travail a été poursuivi et explicité en un algorithme pour les cas simples par l'équipe ORPAILLEUR[4][5]. Mon travail va porter sur le prolongement de ces travaux pour leur mise en place dans un programme et leur amélioration pour la prise en compte de cas un peu plus complexes.

# Remerciements

Avant de débuter ce rapport, il me semble judicieux de remercier tous ceux et celles qui m'ont permis de réaliser ce stage dans ces conditions.

Je commencerai par Miguel Couceiro, Alain Gely et Amedeo Napoli, mes tuteurs qui m'ont accepté au sein du projet et qui m'ont accompagné tout au long de la réalisation de ma contribution et de l'écriture de ce rapport.

Je remercierai également Mme. Bourjot, ma marraine qui a effectué le suivi et l'encadrement de mon stage.

De même que l'équipe Orpailleur dans son intégralité et l'ensemble des autres stagiaires qui m'ont accueilli dans les locaux dans une joie et une bonne humeur omniprésentes, éléments essentiels d'un travail efficace au quotidien.

Et par extension le LORIA dans son ensemble pour ces mêmes raisons.

Je n'oublierai pas l'UFR Mathématiques et Informatique et ses enseignants qui m'ont apporté, tout au long de cette année d'études, les capacités, les connaissances et le recul nécessaires pour réaliser ce genre de projet.

# Table des matières

# 1 Contexte

## 1.1 Laboratoire

Le Loria, Laboratoire lorrain de Recherche en Informatique et ses Applications est une Unité Mixte de Recherche (UMR 7503), commune à plusieurs établissements : le CNRS, l'Université de Lorraine et Inria.

Le laboratoire a pour mission la recherche fondamentale et appliquée en sciences informatiques et ce, depuis sa création, en 1997.

Il est membre de la Fédération Charles Hermite qui regroupe les trois principaux laboratoires de recherche en mathématiques et STIC (Science et Technologies de l'Information et de la Communication) de Lorraine. Le laboratoire fait partie du pôle scientifique AM2I (Automatique, Mathématiques, Informatique et leurs interactions) de l'Université de Lorraine.

Les travaux scientifiques sont menés au sein de 28 équipes structurées en 5 départements, dont 15 sont communes avec Inria, représentant un total de plus de 400 personnes. Le Loria est un des plus grands laboratoires de la région lorraine.

## 1.2 Équipe ORPAILLEUR

Le stage s'est déroulé au sein de l'équipe ORPAILLEUR et plus précisément en collaboration directe avec Miguel Couceiro, Alain Gély et Amedeo Napoli qui la dirige. Elle fait partie du 4$^{\text{ème}}$ département « Traitement automatique des langues et des connaissances ».

Le domaine de recherche de l'équipe est l'exploration de connaissances dans les bases de données (knowledge discovery in databases). Cela consiste à traiter des données afin d'en ressortir des unités de connaissances utiles et réutilisables. Le processus se fait en trois mécanismes principaux : la préparation des données, leur exploitation et l'interprétation des unités extraites en unité de connaissance.

Son nom vient de la comparaison entre son activité et la recherche d'or. En effet en associant la présence d'or en tant qu'unités de connaissances et les données du terrain en tant que base de données, on obtient l'activité de l'équipe sur les données.

# 2 Motivations

Il est important avant de commencer cette section, de préciser que le détail des définitions et propriétés sera détaillé par la suite.

## 2.1 Phylogénie

La phylogénie est un domaine de la biologie ayant pour but l'étude des relations de parenté entre les êtres vivants. Nous nous concentrerons sur la partie du domaine qui traite des relations entre les espèces et leurs évolutions. Pour représenter les informations, on utilise des arbres montrant les différentes espèces avec leurs caractérisques communes comme en figure 2.1 sur laquelle chaque feuille, ici représentée en noir, correspond à une espèce actuelle. Sa lecture est très simple, on part du nœud le plus haut qui correspond à l'intégralité des espèces et on descend progressivement chaque branche qui vient sélectionner une partie des espèces suivant certaines caractéristiques notées sur les nœuds traversés. On part également du principe que chaque nœud correspond à un ancêtre commun à toutes les espèces des feuilles sur lesquelles ce nœud débouche.

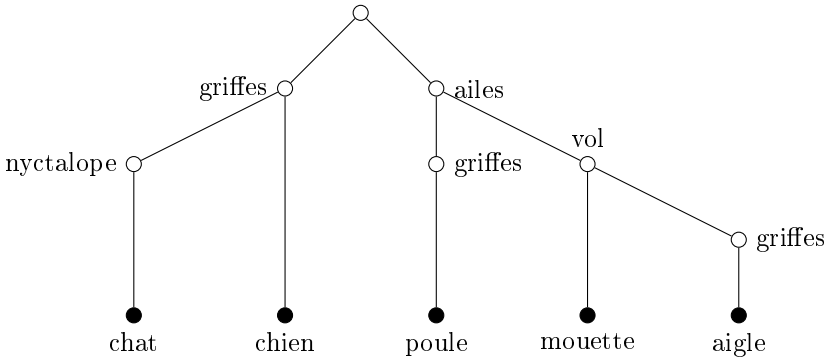


FIGURE 2.1 – Arbre phylogénétique

Ces arbres sont des représentations des données de la matrice et peuvent varier suivant sur quoi on cherche à mettre l'accent. Par exemple la figure 2.1 et la figure 2.2 représentent les mêmes espèces à partir des mêmes données mais avec un accent différent porté avant tout dans le cas de la figure 2.2 sur une catégorisation de la présence de griffes ou non. Ces données sont stockées dans des matrices binaires mettant en lien chaque espèce avec ses caractéristiques comme le montre la figure 2.3. Le problème est alors de choisir le bon arbre, pour diriger ce choix on utilise la notion de parcimonie qui consiste à minimiser le nombre de ramifications nécessaires pour atteindre les espèces visées. Les arbres ainsi obtenus sont dit parcimonieux.

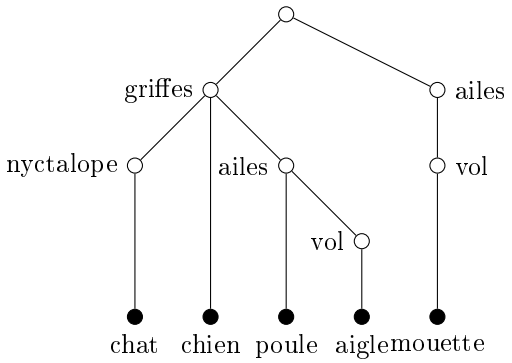


FIGURE 2.2 – Arbre phylogénétique, autre accent

| | griffes | ailes | nyctalope | vol |
|---|---|---|---|---|
| chat | x | | x | |
| chien | x | | | |
| aigle | x | x | | x |
| mouette | | x | | x |
| poule | x | x | | |

FIGURE 2.3 – Matrice espèce/caractéristique

## 2.2 Outil : graphes médians

Le vivant et la multitude de caractéristiques de chaque espèce ne permet pas d'avoir un unique arbre parcimonieux pour représenter tous les accents possibles avec un jeu de données. Pour compenser cela la première solution est de donner tous les arbres possibles, solution n'est pas viable qui pour des raisons évidentes de place et de pertinence. La seconde, proposée par Hans-Jürgen Bandelt dans [1], consiste à encoder l'ensemble de ces arbres dans un graphe médian. Ce sont des graphes dans lesquels pour chaque triplet il n'y qu'un unique nœud communs aux chemins les plus courts les reliant. Cette méthode permet ainsi d'avoir l'intégralité de ces arbres phylogénétiques parcimonieux en une unique représentation au lieu de devoir faire un arbre par information qu'on souhaite mettre en avant. Suivant ce principe nous obtenons la figure 2.4 dans laquelle toutes les espèces, toujours sur les nœuds noirs, disposent des caractéristiques des nœuds qui lui sont au dessus et de celui où elles se trouvent. Dans ce jeu de données l'espèce « chat » dispose des caractéristiques « nyctalope » et « griffes ». Ce graphe enraciné contient toutes les informations que les arbres précédents des figures 2.1 et 2.2 ainsi que les autres arbres qui peuvent être formés à partir de ce jeu de données, peuvent offrir.

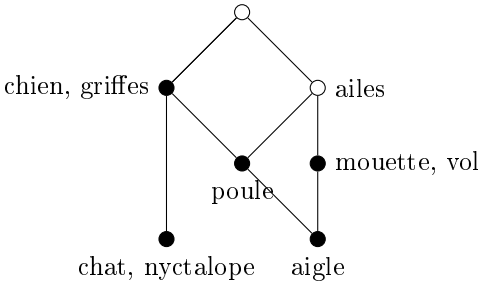


FIGURE 2.4 – Graphe médian

## 2.3 Approche : Analyse de Concepts Formels

Nous avons précédemment parlé de représenter grace à des arbres, des données sous la forme de matrices binares espèce/caractéristique. Cette façon de noter les données dans des matrices est centrale dans l'étude de la découverte de connaissances. Un domaine en découlant, l'analyse de concepts formels [1] se base sur ces matrices afin de créer des groupes d'individus en fontion de leur caractéristiques communes sous la forme de concept. Ils peuvent être ensuite ordonnés dans une représentation graphique, le treillis de concepts, très utilisée pour leurs classifications. Il se lit de bas en haut, chaque espèce dispose de toutes les caractéristiques des nœuds qui sont situés au dessus de son propre nœud.

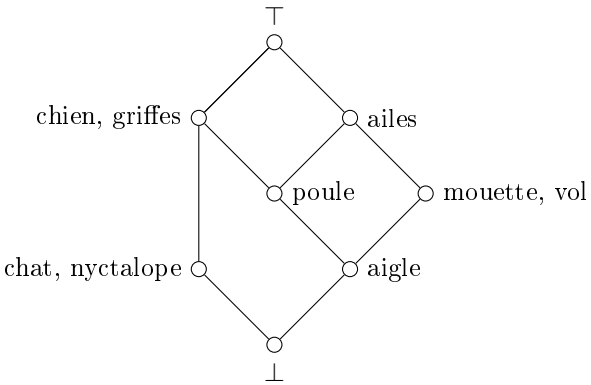


FIGURE 2.5 – Treillis construit à partir de la matrice de données en figure 2.3

Nous pouvons voir que le treillis de la figure 2.5 n'est ni un arbre, ni un graphe médian. En revanche des treillis disposant de certaines caractéristiques disposent de liens avec ces deux autres représentations. Uta Priss s'appuit sur ces ressemblances pour proposer un pont afin d'offrir à la phylogénie les nombreux outils et la

1. FCA - Formal Concept Analysis : Analyse de Concepts Formels

communauté relativement importante de la FCA de même qu'une méthode afin d'obtenir un graphe médian à partir d'un treillis de concepts dont elle n'a pas donné le détails.

L'équipe, notamment au travers de Yacine Namir, a effectué une première exploration [4] visant à implémenter une solution. Par la même occasion, des problèmes sur la méthode sont apparus et seront détaillés dans la suite de ce document. Mais avant cela, nous allons poursuivre par un chapitre de définitions sur les différents éléments qui sont nécessaires à la compréhension de la problèmatique. Puis nous reviendront sur la situation initiale et nous développerons la contribution de ce stage à la résolution de cette problèmatique d'obtenir un graphe médian à partir d'une matrice binaire espèce/caractéristique pour une utilisation en phylogénie.

# 3 Notions nécessaires

## 3.1 Analyse de Concepts Formels

L'Analyse de Concepts Formels utilise des matrices binaires mettant en relation des objets avec des attributs, une matrice porte le nom de contexte et est définie par $C(O, A, I)$ où $O$ est l'ensemble des objets, $A$ l'ensemble des attributs et $I$ l'ensemble des relations entre les objets et les attributs.

**Définition 1 (Contexte)** *Soit un contexte $C(O, A, I)$ dans lequel $O$ est l'ensemble des objets, $A$ l'ensemble des attributs et $I$ l'ensemble des relations entre les objets et les attributs.*

Nous définissons une opération, « prime » pour les objets et attributs, notée « ' ». Elle permet de passer des objets aux attributs et inversement. On lui donne un objet (resp. attribut) ou un ensemble d'objets (resp. attributs) et elle retourne l'ensemble des attributs (resp. objets) qui sont en correspondance, c'est-à-dire les attributs possédés par l'object (resp. les objects qui possèdent l'attribut).

**Définition 2 (Connexion de Gallois)** *Pour $X \subseteq O$ et $Y \subseteq A$, on définie :*
— $X' = \{y \in A : (x, y) \in I, \forall x \in X\}$
— $Y' = \{x \in O : (x, y) \in I, \forall y \in Y\}$

Dans la figure 3.1, nous avons $O = \{chat, chien, aigle, mouette, poule\}$, $A = \{griffes, ailes, nyclatope, vol\}$ et $I = \{(chat, griffes), (chat, nyctalope), (chien, griffes), (aigle, griffes), ...\}$. Nous avons entre autres pour les objets $chat' = \{griffes, nyctalope\}$ et $\{chat, aigle\}' = \{griffes\}$ et inversement pour les attributs $griffes' = \{chat, chien, aigle, poule\}$ et $\{griffes, ailes\}' = \{aigle, poule\}$.

| | *griffes* | *ailes* | *nyctalope* | *vol* |
|---|---|---|---|---|
| *chat* | x | | x | |
| *chien* | x | | | |
| *aigle* | x | x | | x |
| *mouette* | | x | | x |
| *poule* | x | x | | |

FIGURE 3.1 – Contexte

Nous pouvont effectuer plusieurs opérations sur les contextes sans réelles pertes d'informations sur la structure. La première est la clarification, cela consiste à ne garder que les lignes et les colonnes qui ne sont pas en doublon.

**Définition 3 (Contexte clarifié)** *Un contexte $C(O, A, I)$ est clarifié si et seulement si :*
— $\forall x1, x2 \in O$ *si* $x1' = x2'$ *alors* $x1 = x2$
— $\forall y1, y2 \in A$ *si* $y1' = y2'$ *alors* $y1 = y2$

À partir du contexte non clarifié de la figure 3.2 nous obtenons le contexte clarifié de la figure 3.3 avec la suppression de la ligne 5 qui est le doublon de la ligne 2 et la colonne $e$ qui est le doublon de la $b$. Nous pouvons le voir à travers les opérations de prime, deux objets (resp. attributs) sont en doublon losqu'ils obtiennent le même résultat. Nous obtenons $2' = \{a\}$ et $5' = \{a\}$. Il est important de bien comprendre que malgré la suppression de lignes ou de colonnes dans cette opération, nous ne perdons aucune données sur la structure.

| | $a$ | $b$ | $c$ | $d$ | $e$ |
|---|---|---|---|---|---|
| 1 | x | | | x | |
| 2 | x | | | | |
| 3 | x | x | x | | x |
| 4 | | x | x | | x |
| 5 | x | | | | |
| 6 | x | | x | | |

FIGURE 3.2 – Contexte non clarifié

| | $a$ | $b$ | $c$ | $d$ |
|---|---|---|---|---|
| 1 | x | | | x |
| 2 | x | | | |
| 3 | x | x | x | |
| 4 | | x | x | |
| 6 | x | | x | |

FIGURE 3.3 – Contexte clarifié

La seconde opération est la plus utilisée et va plus loin dans la réduction de la taille du contexte sans perte de données, les contextes ainsi obtenus sont dit « contexte réduit » [1]. Le principe est toujours de supprimer les lignes et les colonnes dont on peut se passer, celles qui sont recalculables à partir de celles qu'on garde. On ne garde que les lignes et les colonnes qui ne sont pas l'intersection d'une ou plusieurs autres.

**Définition 4 (Contexte réduit)** *Un contexte $C(O, A, I)$ clarifié est réduit si et seulement si :*
— $\forall x \in O, \forall X \subseteq O$, *si* $x' = X'$ *alors* $x \in X$
— $\forall y \in A, \forall Y \subseteq A$, *si* $y' = Y'$ *alors* $y \in Y$

Dans la figure 3.4 la ligne $2' = \{a\}$ est l'intersection des lignes $1' = \{a, d\}$ et $3' = \{a, b, c\}$ ou $1' = \{a, d\}$ et $6' = \{a, c\}$, on peut donc la supprimer. En revanche, nous n'avons aucune colonne dans ce cas ici.

| | $a$ | $b$ | $c$ | $d$ |
|---|---|---|---|---|
| 1 | x | | | x |
| 2 | x | | | |
| 3 | x | x | x | |
| 4 | | x | x | |
| 6 | x | | x | |

FIGURE 3.4 – Contexte non réduit

| | $a$ | $b$ | $c$ | $d$ |
|---|---|---|---|---|
| 1 | x | | | x |
| 3 | x | x | x | |
| 4 | | x | x | |
| 6 | x | | x | |

FIGURE 3.5 – Contexte réduit

Nous pouvons maintenant définir la notion de concept. Cela correspond aux rectangles maximaux qui se trouvent dans le contexte, regroupant tous les objets qui font partis du concept et tous les attributs qu'ils ont en commun. Un concept est l'ensemble des objets et attributs tel que tous les objets du concepts partagent les attributs du concepts et qu'il n'existe pas d'autre objet partageant les attributs du concept et pas d'autre attributs qui sont partagés par tous les objets du concept.

**Définition 5 (Concept)** *Soit un concept $c$, $X \subseteq O$ et $Y \subseteq A$ :*
— $c = \{X, Y\}$
— $\forall x \in O$, $x \in X \Leftrightarrow x' \subseteq Y$
— $\forall y \in A$, $y \in Y \Leftrightarrow y' \subseteq X$

Sur le contexte de la figure 3.5 nous avons entre autre le concept contenant les objets 3 et 4 et les attributs $b$ et $c$ dont les correspondances forment un rectangle plein.

Maintenant que nous avons défini la notion de concept, nous pouvons définir les treillis de concepts, également appelés treillis de Galois. Ce treillis se base sur une relation d'ordre entre les concepts exprimée sur un diagramme de Hasse. C'est un diagramme orientés du bas vers le haut où chaque élément, ici il s'agit du concept, est un nœud et chaque relation d'ordre est une arête.

**Définition 6 (Treillis de concepts)** *Soit deux concepts $c_1 = (O_1, A_1)$ et $c_2 = (O_2, A_2)$, on a $c_1 \leq c_2$ si et seulement si $O_1 \subseteq O_2$ ou $A_1 \supseteq A_2$.*

1. « standard context » pour la documentation anglaise

En plus des propriétés précédentes, un contexte peut contenir ce qu'on appelle des relations fléches qui ne viennent pas ajouter de l'information mais permettre une extraction plus rapide à porté humaine. Il en existe de trois types $\uparrow$, $\downarrow$, et $\updownarrow$.

**Définition 7 (Relations flèches)** *Soit un contexte $C(O, A, I)$, $o \in O$, $a \in A$ :*
— $o \uparrow a$ *ssi* $(o, a) \notin I$ *et si* $\exists x \in A$, $a' \subset x'$, $(o, x) \in I$ : $\forall y \in A$, $a' \subset y' \Rightarrow y = x$
— $o \downarrow a$ *ssi* $(o, a) \notin I$ *et si* $\exists x \in O$, $o' \subset x'$, $(x, a) \in I$ : $\forall y \in O$, $o' \subset y' \Rightarrow y = x$
— $o \updownarrow a$ *ssi* $o \uparrow a$ *et* $o \downarrow a$.

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | x | x | x | x | $\updownarrow$ |
| 2 | x | $\updownarrow$ | x | x | x |
| 3 | x | x | x | $\updownarrow$ | |
| 4 | x | $\uparrow$ | $\updownarrow$ | x | x |
| 5 | x | x | $\updownarrow$ | $\uparrow$ | |
| 6 | $\updownarrow$ | x | | | |

FIGURE 3.6 – Contexte avec relations fléches

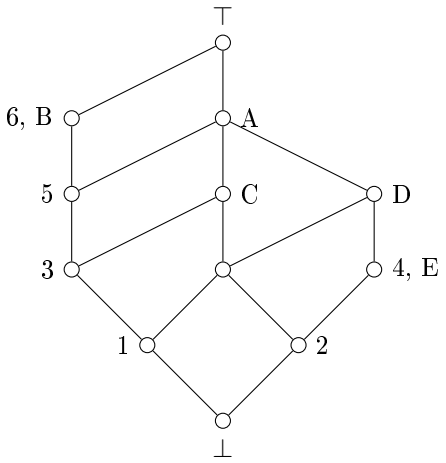


FIGURE 3.7 – Treillis associé

## 3.2 Ensemble ordonné

Un ensemble ordonné est un ensemble $P$ d'éléments avec une relation d'ordre $\leq$, qu'on note $(P, \leq)$. Pour le représenter, on utilise des diagrammes de Hasse. Pour un ensemble $X \subseteq P$ on note $\uparrow X$ le filtre (resp. $\downarrow X$ l'idéal) de $X$. Pour un élément $x \in P$, on note $\uparrow x$ le filtre principal (resp. $\downarrow x$ l'idéal principal) de $x$ tel que pour $\uparrow x = y$ (resp. $\downarrow x = y$) on a $x \leq y$ (resp. $y \leq x$).

**Définition 8 (Ensemble ordonné)** *Soit un ensemble ordonné $(P, \leq)$, $X \subseteq P$ et $Y \subseteq P$ :*
— $\forall x, y \in P$, *on a* $x \leq y$ *ou* $y \leq x$
— $\uparrow X = Y \Rightarrow x \leq y \ \forall x \in X, \forall y \in Y$
— $\downarrow X = Y \Rightarrow y \leq x \ \forall x \in X, \forall y \in Y$

Dans la figure 3.8 nous avons $\uparrow a = a, b, c$ et $\downarrow b = a, b$.

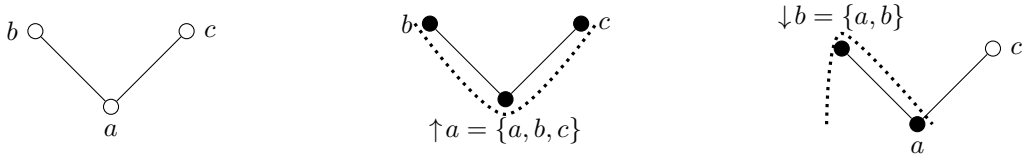


FIGURE 3.8 – Diagrammes de Hasse avec filtres et idéaux

## 3.3 Treillis

Un treillis est un ensemble ordonné $(T, \leq)$ sur lequel on définit pour chaque tuple d'éléments un supremum ou borne supérieure noté $\vee$ et un infimum ou borne inférieure noté $\wedge$, on le note $(T, \leq, \vee, \wedge)$. Pour trois éléments $x, y, z \in T$ tel que $z \leq x$ et $z \leq y$ on peut dire que $x \vee y = z$. Le second est celui d'infimum ou borne inférieure noté $\wedge$. Et de la même façon pour trois éléments $x, y, z \in P$ tel que $x \leq z$ et $y \leq z$ on peut dire que $x \wedge y = z$. Lorsqu'un élément est la borne supérieure (resp. inférieure) de tous les autres éléments on dit qu'il ferme le treillis par les supremums (resp. infimums), on note cet élément $\top$ (resp.$\bot$).

**Définition 9 (Treillis)** *Soit un treillis $(T, \leq, \vee, \wedge)$ :*

— $\forall x, y \in T,\ \exists z \in T : x \vee y = z$
— $\forall x, y \in T,\ \exists z \in T : x \wedge y = z$

Nous définissons également la notion de sous treillis correspondant à un sous ensemble des éléments d'un treillis, et qui vérifient les mêmes conditions définies précédemment.

**Définition 10 (Sous treillis)** *Soit un treillis* $(T, \leq, \vee, \wedge)$, $(T_S, \leq_S, \vee_S, \wedge_S)$ *est son sous-treillis si et seulement si :*
— $T_S \subseteq T$
— $\forall x, y \in T_S,\ \exists z \in T_S : x \vee_S y = z$
— $\forall x, y \in T_S,\ \exists z \in T_S : x \wedge_S y = z$

Sur notre exemple en figure 3.9 nous avons $a = b \vee c$, $d = b \wedge c$, $\top = d$ et $\bot = a$.


FIGURE 3.9 – Treillis avec supremum et infimum

Un élément qui est pas le supremum (resp. infimum) d'autres éléments est un $\vee$-irréductible (resp. $\wedge$-irréductible) et on note l'ensemble de ces élément $J(T)$ (resp. $M(T)$). Bien que le $\top$ et le $\bot$ soient des irréductibles, il est commun de les considérer comme particuliers, par exemple les contextes réduits ne font apparaitre que les irréductibles dans les objets et les attributs à l'exeption du $\bot$ et du $\top$..

**Définition 11 ($\vee$-irreductible et $\wedge$-irreductible)** *Soit un treillis* $(T, \leq, \vee, \wedge)$, $x, y, z \in T$ *:*
— $x$ *est* $\vee$*-irreductible ssi* $x \vee y = z \Rightarrow x = z$
— $x$ *est* $\wedge$*-irreductible ssi* $x \wedge y = z \Rightarrow x = z$

Jusqu'à présent nous utilisions des labels sur la totalité des nœuds, par la suite nous en mettront uniquement sur ceux importants pour la situation illustrée, dont les irréductibles. Dans cet objectif, nous utiliserons des lettres (resp. chiffres) pour les $\vee$-irréductibles (resp. $\wedge$-irréductible), de même que les labels $\top$ et $\bot$. Sur l'exemple de la figure 3.10 les irréductibles sont représentés en noir, d'abord les $\vee$-irréductibles puis les $\wedge$-irréductibles.


FIGURE 3.10 – Diagrammes de Hasse avec irréductibles

Il existe une catégorie de treillis appelée treillis distributif dont la particularité est d'avoir la distributivité des opérations $\vee$ et $\wedge$. Pour tout $x, y, z \in T$ on a $x \vee (y \wedge z) = (x \wedge y) \vee (x \wedge z)$ et $x \wedge (y \vee z) = (x \vee y) \wedge (x \vee z)$. Cette propriété transparait à travers trois conditions équivalentes qui permettent d'attester ou non de la distributivité du treillis permettant ainsi d'utiliser la plus adaptée dans une situation donnée. L'une des façons les plus rapide pour montrer visuellement la non distributivité et de mettre en évidence une apparition du treillis $N_5$ ou $M_3$ qui sont respectivement en figure 3.11 et 3.12.

**Définition 12 (Treillis distributif)** *Un treillis* $(T, \leq, \vee, \wedge)$ *est distributif si et seulement si au moins l'une des trois conditions équivalentes suivantes est vérifiée :*
— $(x \wedge y) \vee (x \wedge z) \vee (y \wedge z) = (x \vee y) \wedge (x \vee z) \wedge (y \vee z)$
— *Le treillis ne possède ni* $N_5$ *ni* $M_3$ *en guise de sous treillis*

— *Le contexte réduit contient une seule relation flèche double par ligne et par colonne*

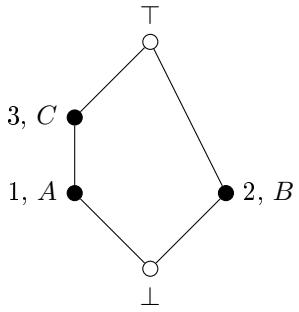


FIGURE 3.11 – $N_5$

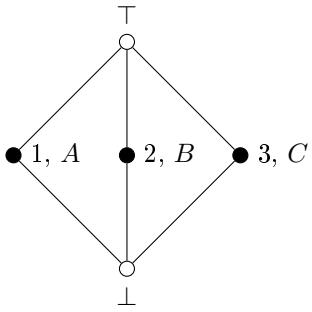


FIGURE 3.12 – $M_3$

## 3.4 Graphe médian

Un graphe est un diagramme composé de nœuds et d'arêtes. Pour chaque triplets de nœuds, on définie l'ensemble des nœuds formant les plus courts chemins entre les éléments du triplet. Si pour tous les triplets du graphe cet ensemble ne contient qu'un unique nœud, noté nœud médian, alors on dit que ce graphe est médian. Considérons deux graphes construit à partir de $N_5$ en figure 3.13 et $M_3$ en figure 3.14.

Nous pouvons voir que sur le premier qu'il y a pas de nœud médian pour le triplet $\{1, 2, 3\}$. Le nœud 1 est présent sur une partie des chemins les plus courts, $(1 \to 3)$ et $(1 \to \bot \to 2)$ tandis que le nœud 3 est présent sur les chemins les plus courts $(1 \to 3)$ et $(3 \to \top \to 2)$. Tandis que sur le second, nous avons deux nœuds médians pour le triplet 1, 2, 3. Le nœud $\top$ est présent sur tous les chemins les plus courts $(1 \to \top \to 2)$, $(1 \to \top \to 3)$ et $(2 \to \top \to 3)$. Mais le nœud $\bot$ est lui aussi présent sur tous les chemins les plus courts $(1 \to \bot \to 2)$, $(1 \to \bot \to 3)$ et $(2 \to \bot \to 3)$.

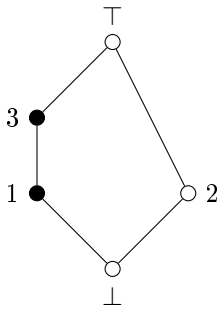


FIGURE 3.13 – Graphe construit à partir de $N_5$

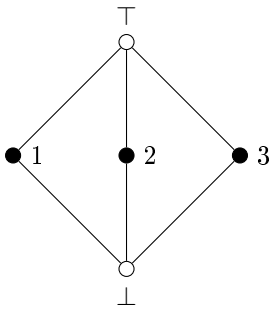


FIGURE 3.14 – Graphe construit à partir de $M_3$

# 4 Existant

Le lien mis en évidence par Uta Priss entre les treillis de concepts et les graphes médians se situe dans le fait qu'un treillis distributif peut être considéré comme un graphe médian. Le tout dans un objectif d'une solution minimale et puisque le graphe médian ne necessite pas de garder le $\perp$, elle propose de ne rendre distributif que les treillis obtenus à partir des filtres des atomes. Pour ce faire il faut agir sur deux aspects. Le premier est la décomposition de ces sous treillis et leur recomposition en un unique. Le second est la transformation en elle-même du sous treillis lambda en treillis distributif. Nous pouvons l'illustrer à travers la figure 4.1 est notre point de départ avec en noir le sous treillis $N_5$ bloquant la distributivité, la figure 4.2 qui se trouve être le treillis après la remise en commun et pour finir la figure 4.3 qui est simplement le graphe médian qui ressort après avoir fini le reste de la méthode et d'avoir supprimer le $\perp$.

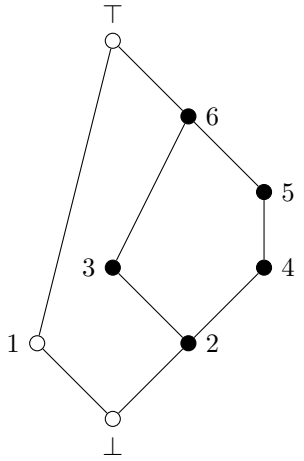


FIGURE 4.1 – Treillis de base

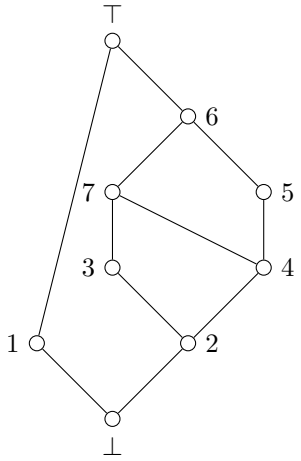


FIGURE 4.2 – Avec treillis des atomes distributifs

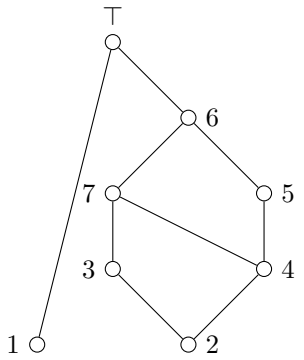


FIGURE 4.3 – Graphe médian résultat

Pour effectuer l'extraction du contexte d'un atome, il suffit d'extraire du contexte global tous les attributs qui sont en correspondance avec l'atome et de prendre tous les objets qui ont une correspondance avec l'un de ces attributs. En prenant en exemple le contexte de la figure 4.4 dans lequel vous voulons extraire le contexte de l'atome 1 nous obtenons le contexte de la figure 4.5.

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 1 | x | x | x | | | |
| 2 | x | | x | | | |
| 3 | | x | | | | |
| 4 | | | x | | | |
| 5 | | | | x | x | x |
| 6 | | | | x | | x |
| 7 | | | | | x | |
| 8 | | | | | | x |

FIGURE 4.4 – Contexte global

| C1 | A | B | C |
|---|---|---|---|
| 2 | x | | x |
| 3 | | x | |
| 4 | | | x |

FIGURE 4.5 – Contexte de l'atome 1 extrait

Ensuite nous utilisons la méthode de transformation d'un treillis quelconque vers un treillis distributif, elle est présentée par Uta Priss et explicitée par l'équipe à travers l'algorithme en figure 4.6. Nous allons l'illustrer avec en exemple le cas $N_5$.

**Algorithm 1:** Construction de contexte de treillis distributif

**Data:** Context réduit $C(J, M, \leq_C)$
**Result:** Contexte réduit $C_d(J, M_d, I_d)$ de $(\mathcal{O}(J), \subseteq, \cap, \cup)$
**begin**
  $M_d \leftarrow \emptyset$
  $I_d \leftarrow \emptyset$
  **foreach** $j \in J$ **do**
    $\uparrow j \leftarrow \emptyset$
    **foreach** $i \in J$ **do**
      **if** $j' \subseteq i'$ **then**
        $\uparrow j \leftarrow \uparrow j \cup i$
    $M_d \leftarrow M_d \cup m_j$
    $X \leftarrow J \setminus \uparrow j$
    **foreach** $x \in X$ **do**
      $I_d \leftarrow I_d \cup (x, m_j)$

FIGURE 4.6 – Construction de contexte de treillis distributif

| | A | B | C |
|---|---|---|---|
| 1 | x | | x |
| 2 | | x | |
| 3 | | | x |

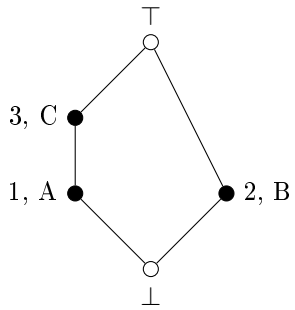


FIGURE 4.7 – Cas de $N_5$

Nous utilisons le contexte réduit. Nous allons créer un nouveau contexte à partir des $\wedge$-irreductibles. Le principe est de tous les parcourir et de créer un attribut pour chacun qui sera en correspondance avec tous les $\wedge$-irreductibles qui ne sont pas dans le filtre de l'attribut en cours.

| | m1 |
|---|---|
| 1 | |
| 2 | x |
| 3 | |

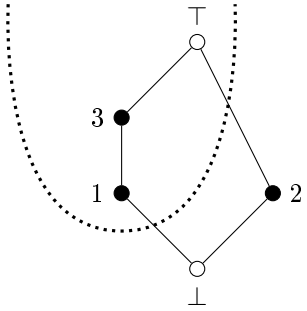


FIGURE 4.8 – Étape 1

| | m1 | m2 |
|---|---|---|
| 1 | | x |
| 2 | x | |
| 3 | | x |

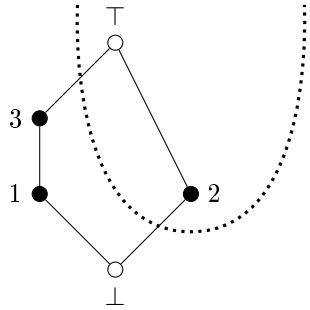


FIGURE 4.9 – Étape 2

| | m1 | m2 | m3 |
|---|---|---|---|
| 1 | | x | x |
| 2 | x | | x |
| 3 | | x | |

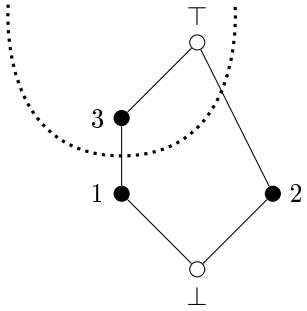


FIGURE 4.10 – Étape 3

Une fois ce nouveau contexte optenu, il suffit de tracer le treillis associé.

| | m1 | m2 | m3 |
|---|---|---|---|
| 1 | | x | x |
| 2 | x | | x |
| 3 | | x | |

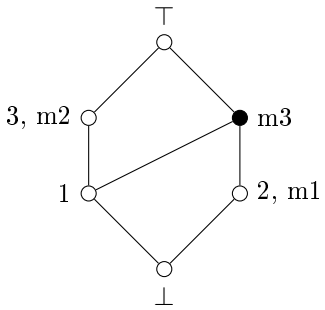


FIGURE 4.11 – $N_5$ après transformation

Nous obtenons ainsi un nouveau contexte pour un nouveau treillis distributif très proche du treillis de base. Il ne reste plus qu'à effectuer la remise en commun des différents contextes ainsi obtenu. À l'inverse de l'extraction, on assemble tout simplement les contextes entre eux sans oublier d'ajouter la correspondance entre l'atome source du contexte extrait avec tous les attributs contenus dans ce contexte comme le montre les figures 4.12 et 4.13. Il ne reste plus qu'à supprimer le nœud $\perp$ pour obtenir notre graphe médian.

| C1 | A | B | C |
|---|---|---|---|
| 2 | | x | x |
| 3 | | | x |
| 4/7 | x | x | |
| C4 | D | E | F |
| 5 | | x | x |
| 6 | | | x |
| 1/7 | x | x | |
| C7 | G | G | I |
| 8 | | x | x |
| 9 | | | x |
| 1/4 | x | x | |

FIGURE 4.12 – Contextes extraits pour les atomes 1, 4 et 7

| | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 1 | x | x | x | x | x | | x | x | |
| 2 | | x | x | | | | | | |
| 3 | | | x | | | | | | |
| 4 | x | x | | x | x | x | x | x | |
| 5 | | | | | x | x | | | |
| 6 | | | | | | x | | | |
| 7 | x | x | | x | x | | x | x | x |
| 8 | | | | | | | | x | x |
| 9 | | | | | | | | | x |

FIGURE 4.13 – Contexte réassemblé

Une fois la méthode mise en place, on constate des problèmes. Sur des cas moins triviaux il se pose la question de l'optimalité du treillis unique obtenu. Prenons en exemple la figure 4.14 avec en noir une partie des nœuds empêchant les treillis des atomes d'être distributif par la présence d'un sous treillis $N_5$. Si nous appliquons la méthode dessus nous obtenons la figure 4.15 avec également en noirs des nœuds posant le même problème.

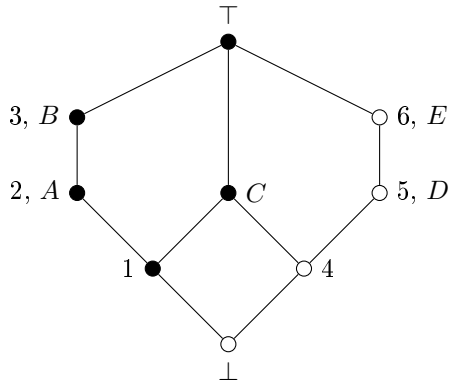


FIGURE 4.14 – Cas posant problème : situation de départ

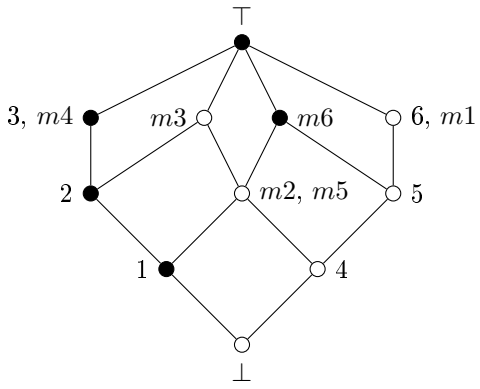


FIGURE 4.15 – Cas posant problème : situation après l'application de la méthode

Une première solution serait de refaire la méthode jusqu'à n'obtenir aucun sous treillis problèmatique. C'est une solution fonctionnelle mais qui vient perturber l'utilité de la méthode. Nous avons besoin d'une représentation globale la plus simple et proche du treillis d'origine possible. Cette méthode ne fait perdre aucune liaison et n'en ajoute uniquement pour rendre le treillis distributif, si nous la faisons en boucle, nous obtenons dans le pire des cas le treillis distributif le plus grand qui est en figure 4.16 avec en noirs les nœuds déjà présent dans le treillis d'origine. Or une solution existe en figure 4.17 mais que la méthode ne parvient pas à atteindre. Le stage a consisté en la modification de la méthode afin de trouver cette solution optimale et de rechercher d'autres potentiels cas problématiques pour les prendre en considération.

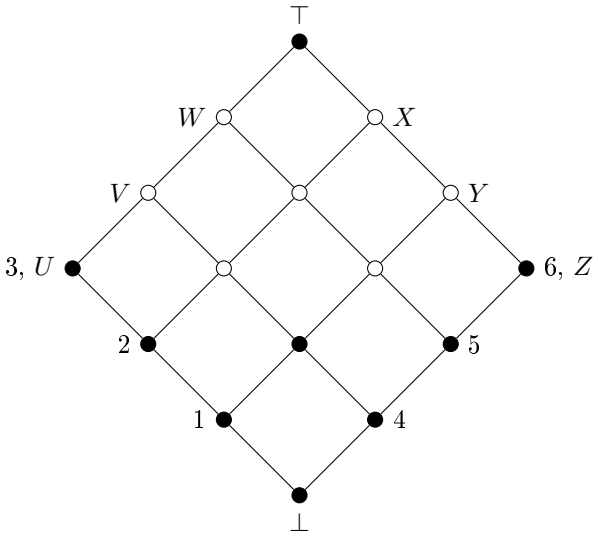


FIGURE 4.16 – Cas posant problème : situation maximale

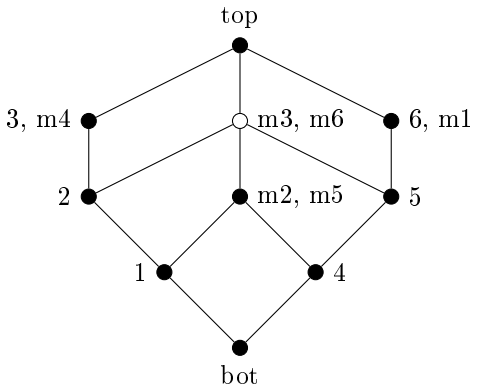


FIGURE 4.17 – Cas posant problème : situation souhaitée

# 5 Contribution

## 5.1 Implémentation

Le stage s'est déroulé en deux phases. La première consistait à implémenter la méthode détaillée dans la section précédente afin de la tester sur des cas de façon automatique. La seconde partie du stage avait pour but d'améliorer la méthode afin de prendre en compte le problème de fusion des nœuds. Avec en parallèle un renforcement de la compréhension du domaine à travers des articles sur l'existant et des discussions avec l'équipe.

La toute première question à se poser pour l'implémentation est celle du langage. Le problème n'ayant pas de contraintes particulières je suis partis sur un programme orienté objet en Python. C'est un langage assez répandu et très malléable, ce qui fait de lui un choix basique mais fiable. Choix également renforcé par mon expérience principalement tournée sur du langage objet permettant de m'adapter plus facilement. Une fois ce choix fait, le plus long a été d'adapter l'algorithme pour sa mise en fonction, cela m'a également permis de comprendre de mieux en mieux les mécanismes mis en jeu. À ce stade nous obtenons le workflow en figure 5.1. Le programme commence par effectuer l'importation des données d'entrée. Il extrait un contexte pour chaque treillis formé par les filtres des atomes et effectue l'algorithme vu précédemment en figure 4.6. Une fois fait, le programme regroupe les contextes en un unique et exporte le résultat sous forme graphique à l'aide de Graphviz[6] à travers sa bibliothèque[7]. De plus nous effectuons une exportation sous forme de matrice binaire pour ConExp[8], une application permettant de générer les treillis. Nous avons toujours dans cet état le problème sur les cas non triviaux développé plus tôt.

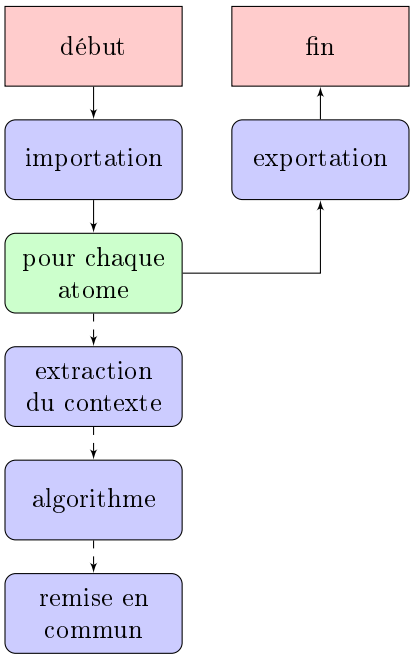


FIGURE 5.1 – Worflow de la première version du programme

Il est question à présent de la fusion des nœuds dans le but d'obtenir le treillis désiré, pour en ressortir des règles de fusions. Nous allons nous baser sur le cas mis en avant dans [5] avec en figure 5.2 le treillis de départ. La figure 5.3 est un rappel du résultat dans l'état actuel et la figure 5.4 est le résultat optimal. On remarque que les points noirs 10 et 12 du résultat obtenu peuvent ne faire qu'un pour obtenir le point 8 du résultat désiré. En regardant également le treillis distributif maximum en figure 5.5 on a l'intuition qu'il suffit de remplacer les points qu'on souhaite fusionner par leur supremum, cela à été notre point de départ.

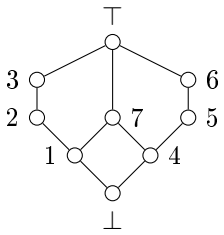


FIGURE 5.2 – Treillis de départ

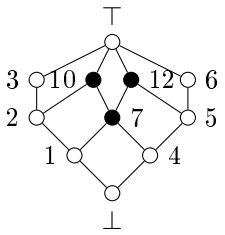


FIGURE 5.3 – Treillis obtenu

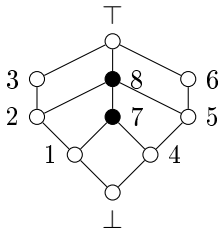


FIGURE 5.4 – Treillis désirée

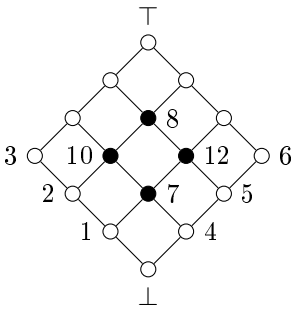


FIGURE 5.5 – Treillis distributif maximum

Nous avons ensuite travaillé sur cette piste à partir de toute une série de treillis qui se trouve en annexe A ayant chacun ses spécificités. À chaque résultat déviant du résultat souhaité sur l'un des cas, nous avons effectué un ajustement des conditions de fusion des nœuds tout en vérifiant le bon fonctionnement sur les cas déjà traités. Nous sommes arrivés à trois conditions.

**Définition 13 (Conditions de fusion)** *Soit $a$ et $b$ les deux nœuds à fusionner, le contexte actuel $C(J, M, I)$, le contexte de départ $C_o(J_o, M_o, I_o)$ et $X$ l'ensemble des atomes de $C$ :*

- *Les deux nœuds à fusionner ne doivent pas faire partis des nœuds du treillis de départ.*
  $a, b \in J \setminus J_o : a \neq b$
- *Les deux nœuds à fusionner doivent être en couverture d'un nœud présent dans au moins deux filtres d'atomes.*
  $Pour (x_1, x_2 \in X \text{ avec } x_1 \neq x_2), \exists c \in J : a' \subseteq c' \subseteq x'_1 \text{ et } b' \subseteq c' \subseteq x'_2$
- *Les deux nœuds à fusionner ne doivent pas avoir de nœuds en commun dans leurs idéaux une fois que les idéaux du nœud en commun (celui dont il est question dans la condition précédente) leur sont retirés.*
  $(\downarrow a \cap \downarrow b) \setminus \downarrow c = \emptyset$

Nous pouvons l'illustrer sur les figures précédentes. Les nœuds 10 et 12 de la figure 5.3 ne sont pas présent dans la figure 5.2, ils ne font donc pas partis du treillis de départ. Ces deux nœuds sont également en couverture du nœud 7 qui est présent dans les filtres de l'atome 1 et de l'atome 4. Et pour finir, nous avons $\downarrow 10 \cap \downarrow 12 = 7, 1, 4, \bot$ et $\downarrow 7 = 7, 1, 4, \bot$ donc $(\downarrow 10 \cap \downarrow 12) \setminus \downarrow 7 = \emptyset$.

Avec la mise en place de cette fusion des nœuds, nous obtenons un nouveau workflow détaillé en figure 5.6.

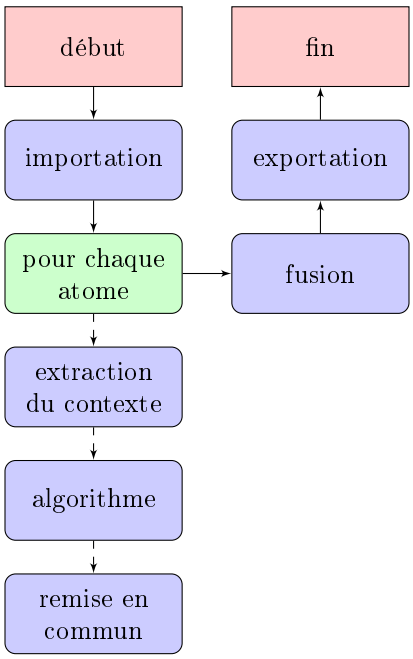


FIGURE 5.6 – Worflow de la seconde version du programme, avec la fusion des nœuds

## 5.2 Problèmes et difficultés

### 5.2.1 Système de boucle

Nous avons rencontré un problème qui porte sur des cas où un passage complet de la méthode avec post traitement de fusion ne suffit pas, ce cas est présenté sur la figure 5.7. La fusion recréant une situation avec des treillis d'atomes non distributif et demandant donc de refaire un passage dans tout le processus.

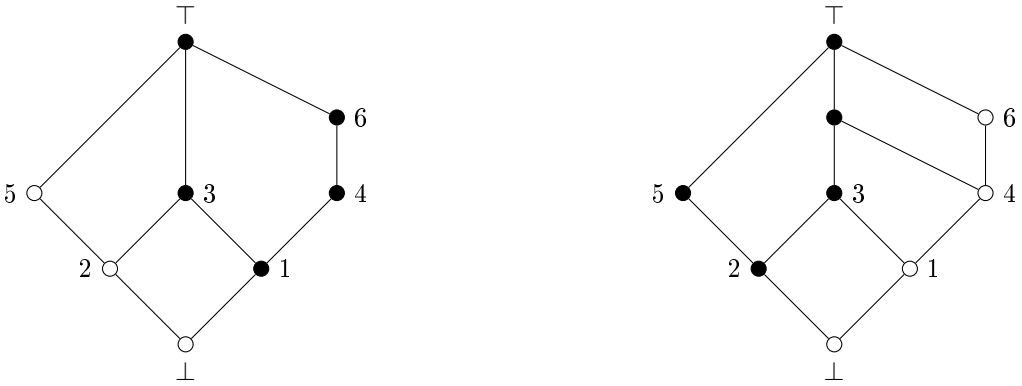


FIGURE 5.7 – Cas cla_v7 : première itération

À ce stade, plusieurs correctifs ont été envisagé. Le premier est de faire une boucle de traitement où on refait tout le processus tant que tous les treillis d'atome ne sont pas disctributif, comme le montre la figure 5.8 représentant la seconde itération, c'est la solution utilisée à l'heure actuelle. La figure 5.9 représente le nouveau workflow global et la figure 5.10 détaille le comportement du bloc « médian ? ».

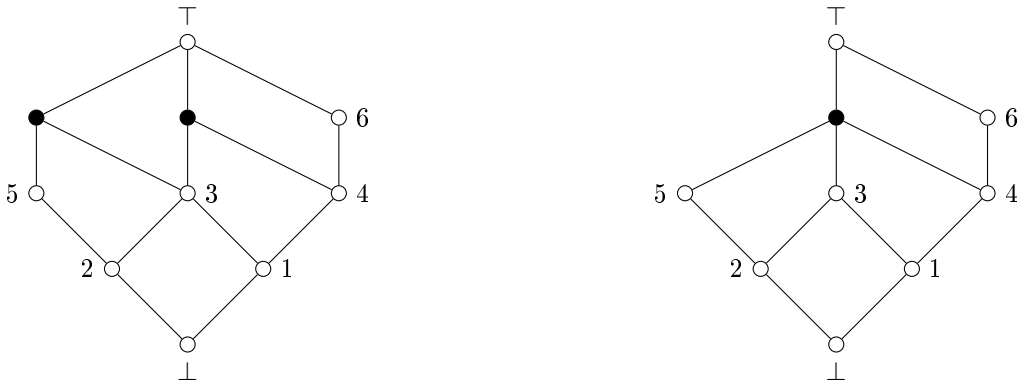


FIGURE 5.8 – Cas cla_v7 : seconde itération

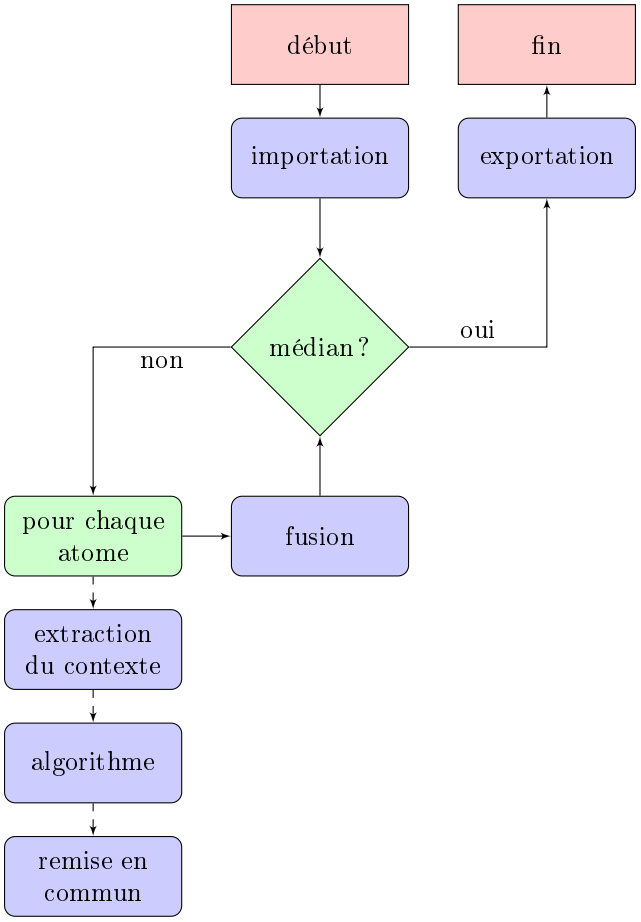


FIGURE 5.9 – Worflow de la version du programme incluant la boucle

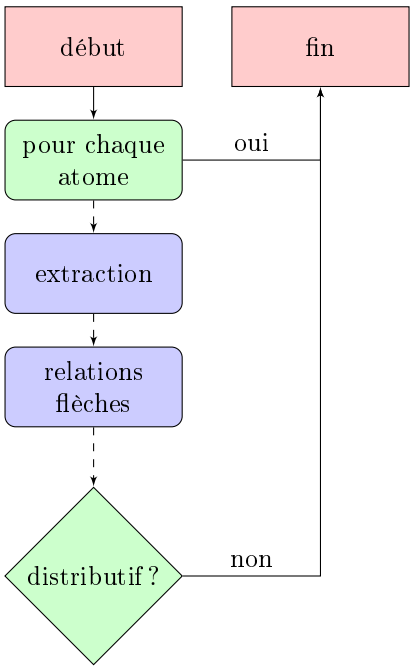


FIGURE 5.10 – Sous worflow correspondant au bloc « médian ? »

### 5.2.2 Apparition de chaines

Le second problème qui est apparu ne se pose que sur un seul cas actuellement. Ce n'est pas que la méthode ne fonctionne plus, mais elle donne en résultat un treillis bien plus gros que celui espéré. Il provoque à un moment donné la création d'une chaine suite à la fusion de nœuds qui va par la suite dégénérer en un sous treillis non désiré. Alors qu'avec une analyse du cas particulier, nous pouvons couper court à ce probème. Voici en figure 5.11 le treillis de départ et en figure 5.12 le treillis souhaité et obtenu manuellement.

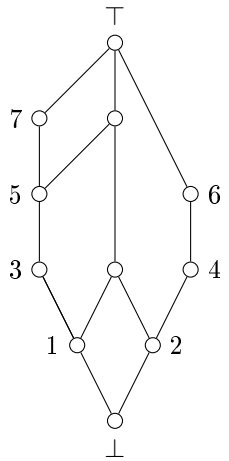


FIGURE 5.11 – Treillis du problème de chaine : départ

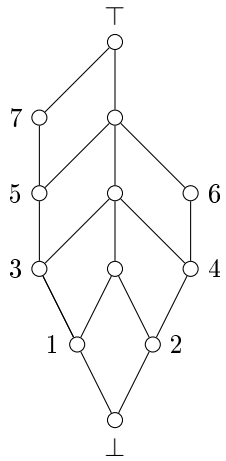


FIGURE 5.12 – Treillis du problème de chaine : souhaité

Les figures 5.13 à 5.18 représentent les treillis successifs au fil du passage dans le programme du cas problèmatique.

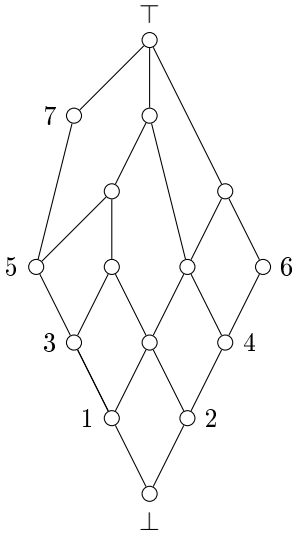


FIGURE 5.13 – Treillis du problème de chaine : 1ère distributivité

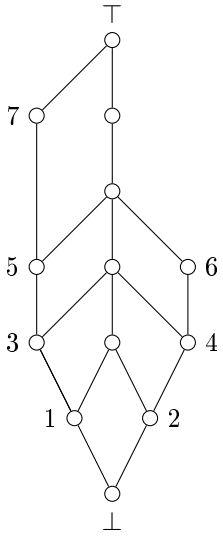


FIGURE 5.14 – Treillis du problème de chaine : 1ère distributivité avec fusion

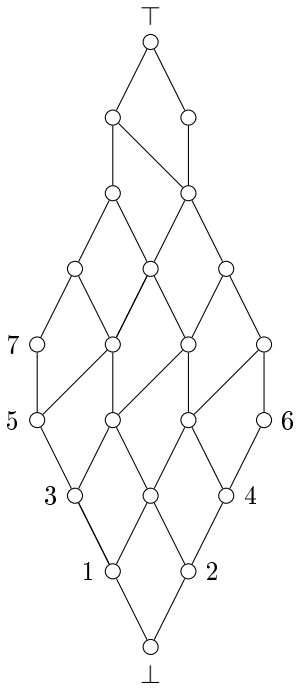


FIGURE 5.15 – Treillis du problème de chaine : 2nde distributivité

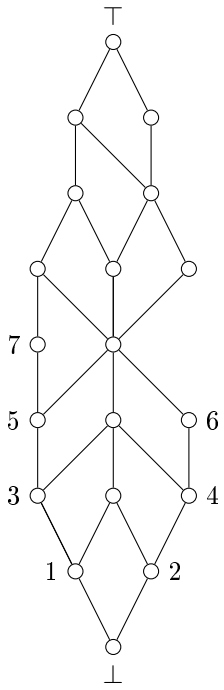


FIGURE 5.16 – Treillis du problème de chaine : 2nde distributivité avec fusion

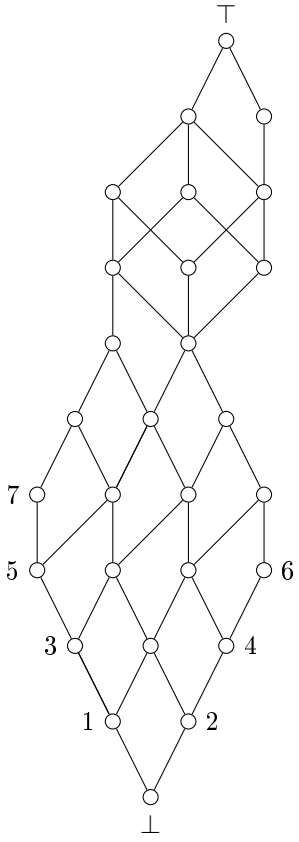


FIGURE 5.17 – Treillis du problème de chaine : 3ème distributivité

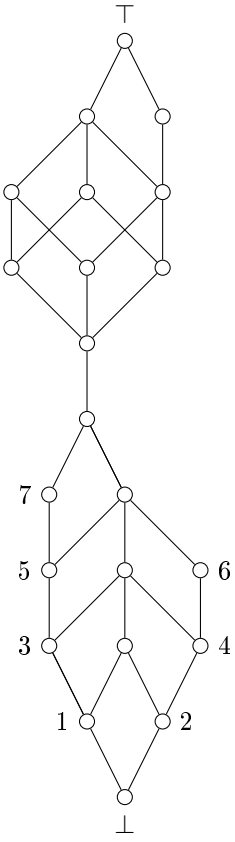


FIGURE 5.18 – Treillis du problème de chaine : 3ème distributivité avec fusion

Comme nous pouvons le voir, cela donne un treillis plutôt éloigné de ce qui est souhaité. L'idéal est de trouver réellement le moment exacte et la raison exacte du soucis. Faute d'une meilleure solution à ce moment, pour

contourner ce problème j'ai ajouté une condition visant à épurer les situations de chaines inutiles, en fusionnant vers le bas les nœuds n'ayant qu'un unique nœud en ouverture et n'étant la couverture que d'un unique nœud et sans être l'un des nœuds du treillis de base. Cette épuration est effectuée juste après la fusion.

## 5.3 Ouvertures

### 5.3.1 Système de boucle

Comme nous l'avons vu précédemment, nous devons boucler sur l'algorithme afin de prendre en considération les cas qui génèrent un nouvel $N_5$ ou $M_3$ lors de leur première résolution. Les méthodes présentées ici ne sont pas utilisées dans le programme mais sont des pistes intéressantes à garder à l'esprit en cas de problèmes futurs.

Pour la première, il s'agit de faire la fusion pas à pas. On execute la méthode différemment. On extrait les contextes des treillis des atomes puis on effectue pour chacun d'eux la transformation en contexte de treillis distributif et on l'assemble avec le contexte global et c'est à ce moment où on fait les fusions si besoin avant de passer au contexte du treillis de l'atome suivant. Cette façon de faire peut permettre de gagner du temps sur la phase de la fusion mais en ferra perdre lors de l'étape de transformation en treillis distributif. Je ne peux pas déterminer laquelle est la meilleure à l'heure actuelle, reste à voir si à l'avenir de nouveaux problèmes se posent et si cette nouvelle méthode peut apporter des réponses.

La seconde est de faire une boucle sur tout le cœur du processus et de faire les fusions de nœuds à la fin. Cette solution contrairement à la première empêche toute possibilité de cycle dû à la fusion mais donne la plupart du temps le treillis distributif maximal pour chaque atome et la fusion devient par conséquent très compliquée, ce qui donne des treillis bien plus grand que ceux visés.

### 5.3.2 Le cas des chaines

Malheureusement, cette partie n'est pas fiable et résulte principalement d'une intuition basée sur un objectif à atteindre et donnant lieu à un raccourci par rapport à une propriété ou à un traitement non mis à découvert. C'est sous doute la principale du programme à garder sous surveillance et à chercher à améliorer.

### 5.3.3 Optimalité du treillis

La base même du besoin de ces travaux qui est d'obtenir un graphe médian pour la phylogénie à partir d'une matrice binaire, demande d'avoir en sortie un treillis le plus petit et le plus proche possible du treillis du contexte d'origine. Il est donc évident que l'une des prochaines étapes les plus importantes consiste à quantifier et mesurer cette différence entre deux treillis. On est dans une situation d'ajout d'éléments, la solution la plus simple est donc de compter tout simplement les éléments ajouter sous forme de tuple comprenant un compteur pour les arcs et un compteurs pour les nœuds. Par rapport au treillis de la figure 5.19, le treillis de la figure 5.20 est à une distance de $(1, 2)$ pour avoir un nœud et deux arcs de plus. Cela reste un ébauche de méthode, il faut évidemment une reflexion bien plus approfondie et travaillée sur de nombreux cas pour commencer à arriver sur une méthode viable.

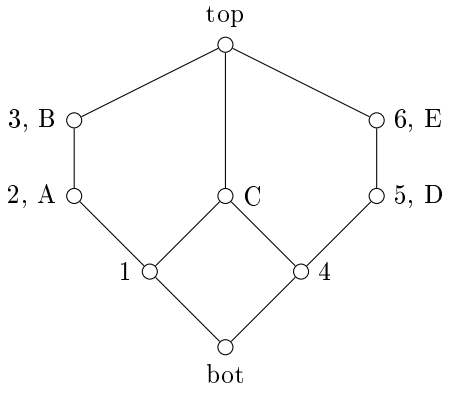


FIGURE 5.19 – Treillis avant traitement

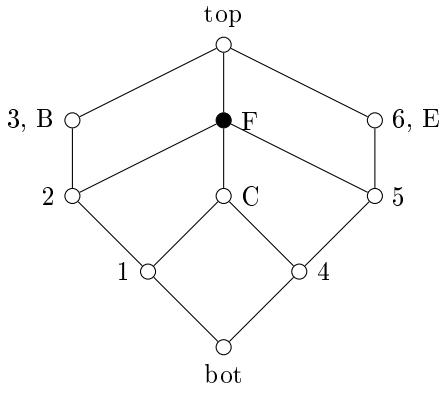


FIGURE 5.20 – Treillis après traitement

De plus, mes travaux dans ce rapport nécessite également un travail plus approfondi pour juger de son efficacité sur ce plan d'optimisation. De regarder sur des cas beaucoup plus complexe et plus proche de données réelles si nous obtenons des graphes médians utilisables à partir des treillis obtenus.

### 5.3.4 Optimisation du programme

Le dernier point important à traiter venant à la suite de ce travail est l'optimisation du programme en lui-même afin de lui faire gagner en vitesse de traitement et lui faire perdre en possibilité de bogues via une optimisation de son code. Il reste un logiciel développé dans un cadre de recherche et qui a subit des ajouts itératifs qui n'ont pas été systématiquement gardés.

# 6 Bilan

## 6.1 Contribution

Nous cherchions à produire des treillis utilisables sous forme de graphes médians afin d'être utilisable dans la phylogénie en encodant l'ensemble des arbres parcimonieux dans un graphe unique. Pour se faire nous nous sommes basés sur les travaux de H.J. Bandelt, U. Priss et de l'équipe, en recherchant à générer un treillis dont les sous-treillis construits à partir des filtres de ses atomes sont distributifs et ceci à partir d'un treillis quelconque.

Nous avons obtenu un programme permettant de le faire. Il commence par extraire le contexte initial d'un fichier pour ensuite regarder si tous les treillis générés à partir des filtres des atomes sont distributifs en calculant les relations flèches pour utiliser la propriété que le contexte réduit d'un treillis distributif possède uniquement une relation fléche double par ligne et par colonne et aucune autre relations fléches. Si ce n'est pas le cas, nous appliquons une méthode de transformation sur chacun d'eux avant de les remettre en commun dans un unique treillis. À partir de là, nous effectuons la fusion deux à deux de tous les couples de nœuds remplissant les conditions suivantes :

— Les deux nœuds à fusionner ne doivent pas faire partis des nœuds du treillis de départ.
— Les deux nœuds à fusionner doivent être en couverture d'un nœud présent dans au moins deux filtres d'atomes.
— Les deux nœuds à fusionner ne doivent pas avoir de nœuds en commun dans leurs idéaux une fois que les idéaux du nœud en commun (celui dont il est question dans la condition précédente) leur sont retirés.

Une fois les fusions effectuées nous recommencons à l'étape de vérification de la distributivité des treillis des atomes. Une fois que nous l'atteignons le programme s'arrête et génére le treillis obtenu sous forme pdf et un fichier texte du contexte réduit exploitable par ConExp. Vous pourrez trouver en Annexe A tous les cas qui ont été traité et testé pendant le développement. Tandis que l'Annexe B présente le formatage du fichier source et du fichier d'exportage pour ConExp.

## 6.2 Stage

Ce stage a été bénéfique sur le plan technique pour les raisons classiques d'approfondissement des différents éléments dont je me suis servis tel que le langage Python, Latex et des notions vues en cours qui ont été appliqué sur un cas concret. Mais également sur le plan personnel, le Loria reste un lieu des plus intéressant pour échanger sur un sujet afin de faire avancer la problèmatique avec des personnes ayant des approches différentes pour s'approcher au mieux d'une solution fiable.

Et surtout que la recherche est un domaine comprenant des personnes passionnées pas seulement par ce qu'elles font mais également par le simple fait d'avancer dans une parcelle d'inconnue pour faciliter un fragment de vie d'un inconnu. Elles en viennent à devoir faire des choix dont les conséquences sont encore inconnues et qu'elles doivent justifier au mieux à ce moment tout en devant les remettre en cause à chaque itération. Ce stage m'a permis de confirmer une nouvelle fois mon goût pour cette façon de faire, d'agir au mieux de nos connaissances à ce moment pour le simple besoin de vouloir faire les choses au mieux même si ce n'est pas pour notre propre profit direct.

# Bibliographie


[1] Hans-Jürgen Bandelt and Jarmila Hedlíková. Median algebras. *Discrete Mathematics*, page 30, 1980.

[2] Hans-Jürgen Bandelt, Peter Forster, and Röhl Arne. Median-joining networks for inferring intraspecific phylogenies. *Molecular Biology and Evolution*, page 12, 1998.

[3] Uta Priss. Representing median networks with concept lattices. *journal name*, page 11, 2013.

[4] Alain Gély, Miguel Couceiro, Yassine Namir, and Amedeo Napoli. Contribution à l'étude de la distributivité d'un treillis de concepts. *journal name*, page 12, 2018.

[5] Alain Gély, Miguel Couceiro, and Amedeo Napoli. Achieving distributivity in formal concept analysis. *journal name*, page 12, 2018.

[6] John Ellson, Emden Gansner, Yifan Hu, Erwin Janssen, and Stephen North. Documentation graphviz pour python. Disponible sur : `https://www.graphviz.org/`. Consulté de avril à août 2018.

[7] Sebastian Bank. Documentation graphviz pour python. Disponible sur : `https://pypi.org/project/graphviz/`. Consulté de avril à août 2018.

[8] Serhiy Dr. Serhiy Yevtushenko, Tim Kaiser, Julian Tane, and Sergei ObjedkovEllson. Conexp application. Disponible sur : `http://conexp.sourceforge.net/`. Consulté de avril à août 2018.

# Annexes

# A Exemples de cas

Cette annexe regroupes tous les cas utilisés pour les divers essais de cas particuliers avec l'état initial et l'état final pour chacun. Ils considèrent tous le programme final du stage.

## A.1 Dérivé de $N_5$

Tout d'abord, une série dérivé du cas cla.

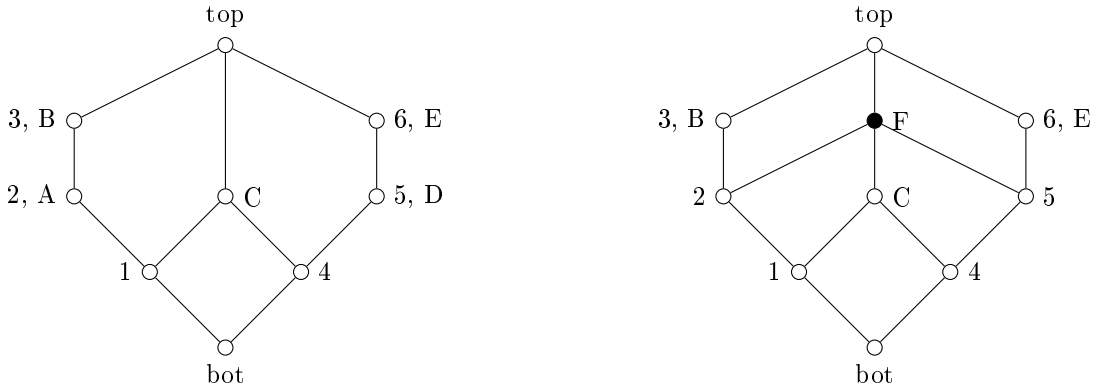


FIGURE A.1 – Cas cla_v1

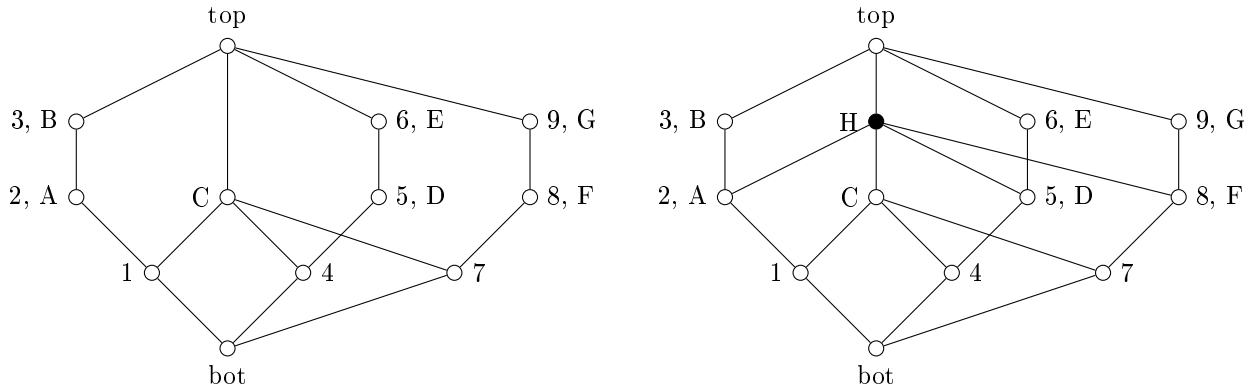


FIGURE A.2 – Cas cla_v2

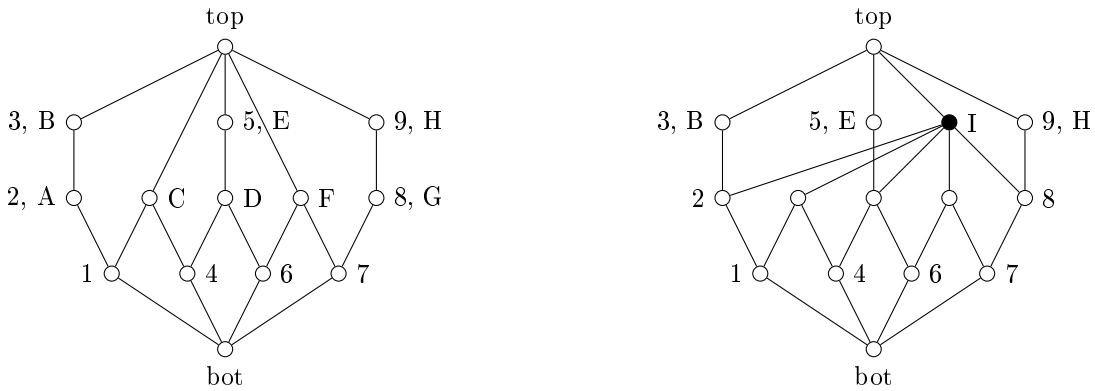


FIGURE A.3 – Cas cla_v3

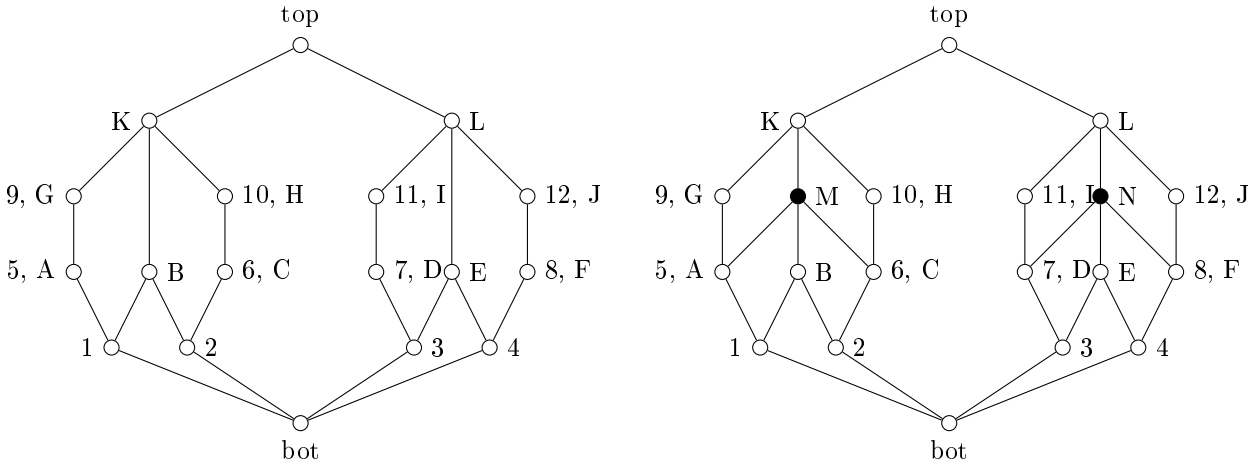


FIGURE A.4 – Cas cla_v4

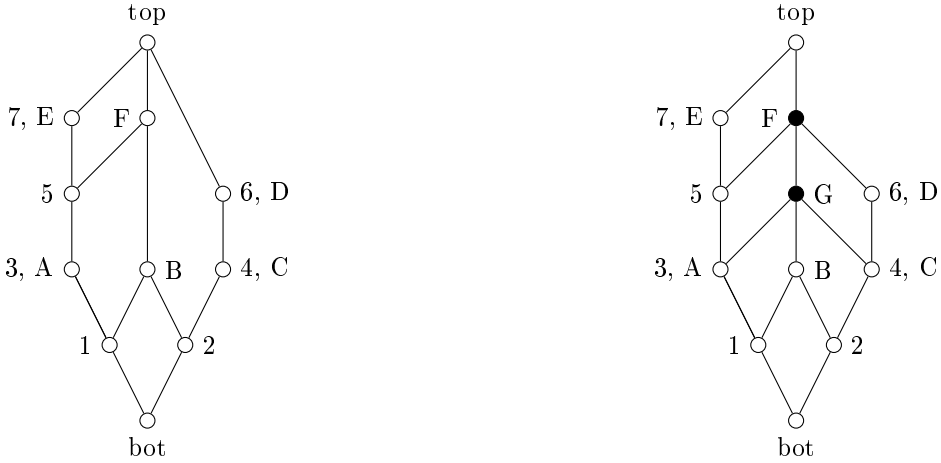


FIGURE A.5 – Cas cla_v6

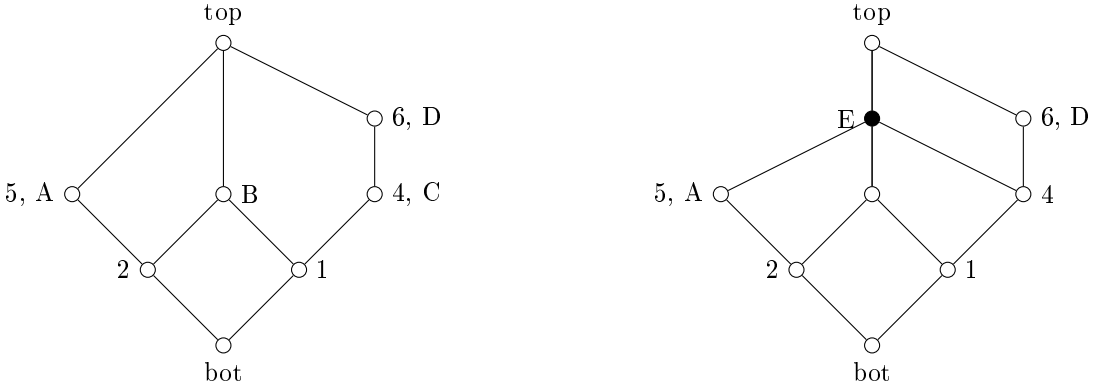


FIGURE A.6 – Cas cla_v7

## A.2 Dérivé de $M_3$

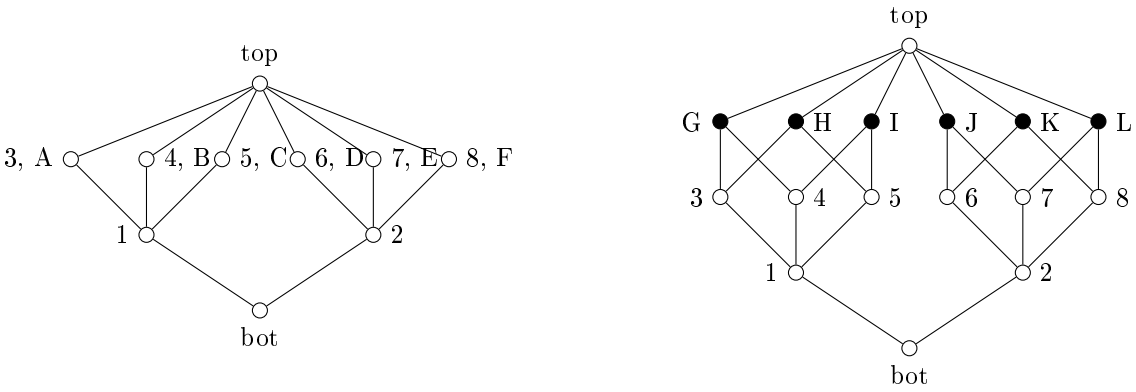


FIGURE A.7 – Cas dm3_v0

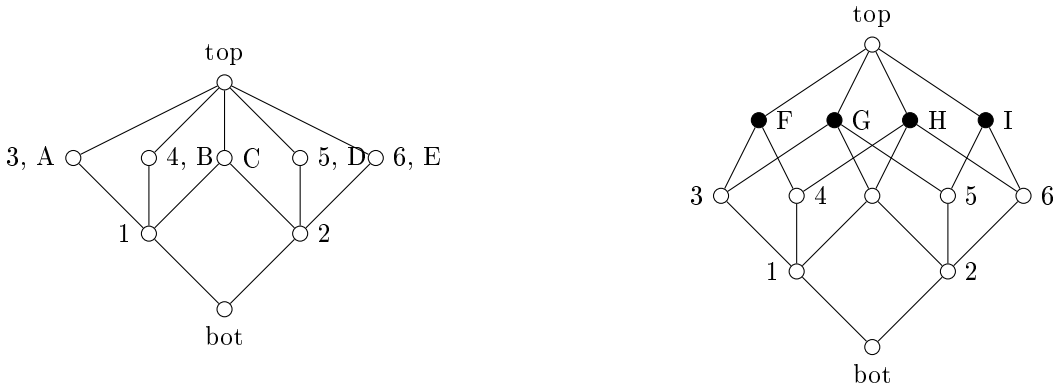


FIGURE A.8 – Cas dm3_v1

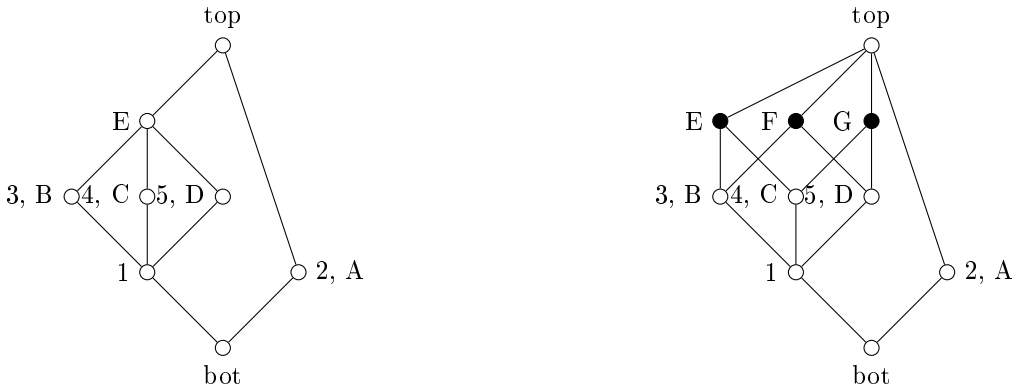


FIGURE A.9 – Cas dm3_v2

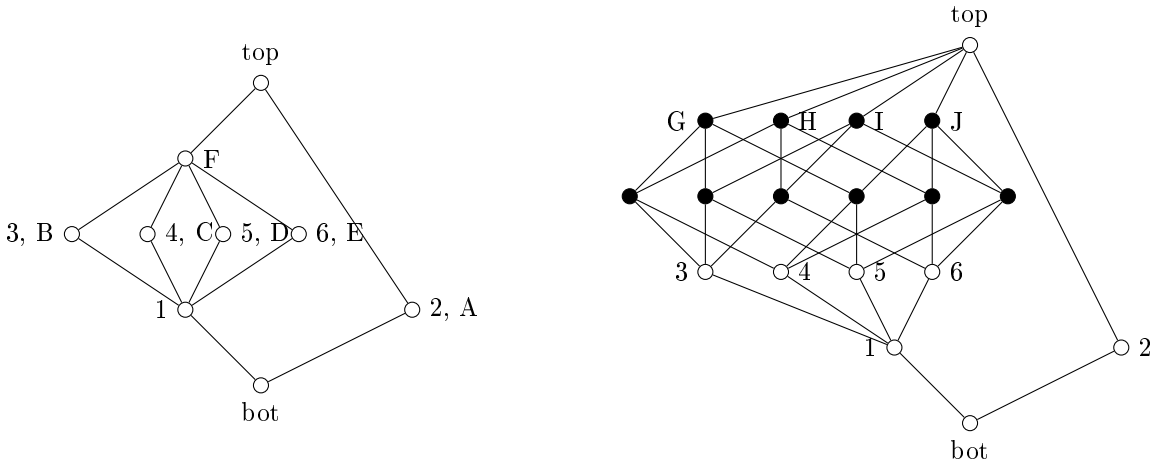


FIGURE A.10 – Cas dm3_v3

## A.3 Cas de H.J. Bandelt

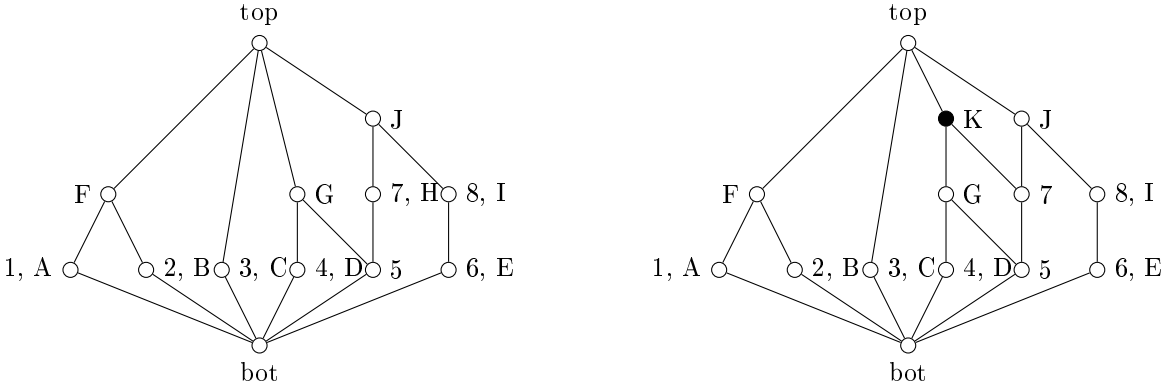


FIGURE A.11 – Cas Bandelt table 1 ([2])

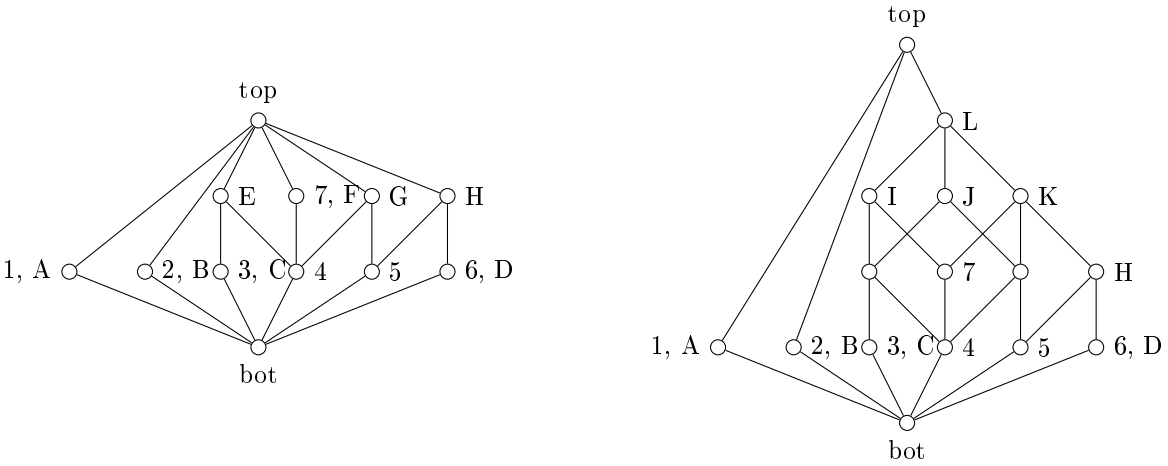


FIGURE A.12 – Cas Bandelt table 2 ([2])

## A.4 Cas de U. Priss

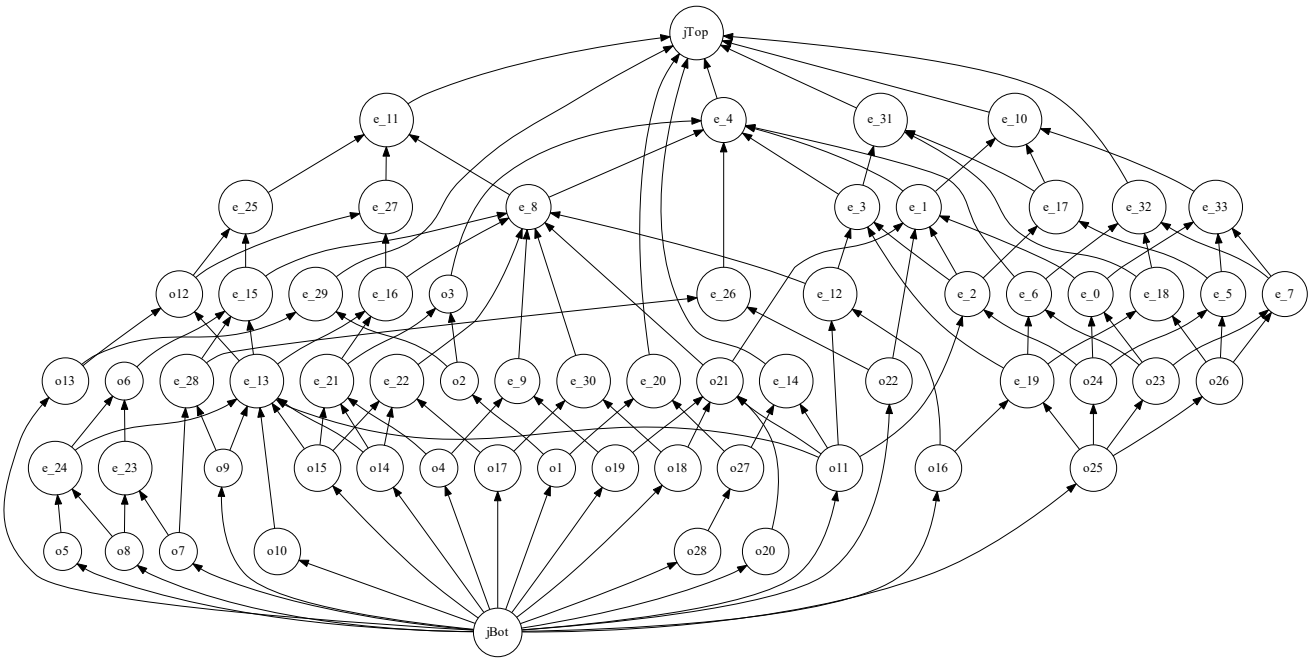


FIGURE A.13 – Cas Priss ([3])

# B Fichiers d'entré et de sortie

Cette annexe présente le fichier texte d'entré contenant le contexte de base et le fichier texte de sortie pour l'utilisation du résultat sous ConExp.

## Input

Le fichier d'entré peut prendre de multiples formes. Il demande néanmoins d'avoir en première ligne les attributs séparé par une tabulation. Chaque ligne suivant correspondra à un nouvel objet sans avoir à indiquer son labe et dont chaque correspondance ou absence de correspondance séparées par un tabulation. Une non correspondace se marque avec un « 0 » ou l'absence de caractère. La correspondance est signalé par tout autre caractère.

| a1 | a2 | a3 | a4 | a5 |
|---|---|---|---|---|
| x | x | | x | |
| | x | x | | x |
| x | | | x | |
| | | x | | x |
| | | | | x |

| a1 | a2 | a3 | a4 | a5 |
|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 |

FIGURE B.1 – Fichier source de cla_v1

## Output

Le fichier de sortie quant à lui respecte la mise en forme d'importation de ConExp. Les attributs sur la première ligne séparé par une tabulation suivis d'une ligne vide et enfin des correspondances avec un objet par ligne sans indiquer le label et une notation binaire « 0 » et « 1 » indiquant respectivement la non correspondance et la correspondance.

| a1 | a2 | a3 | a4 | a5 |
|---|---|---|---|---|
| 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 |

FIGURE B.2 – Exportation de cla_v1 pour ConExp