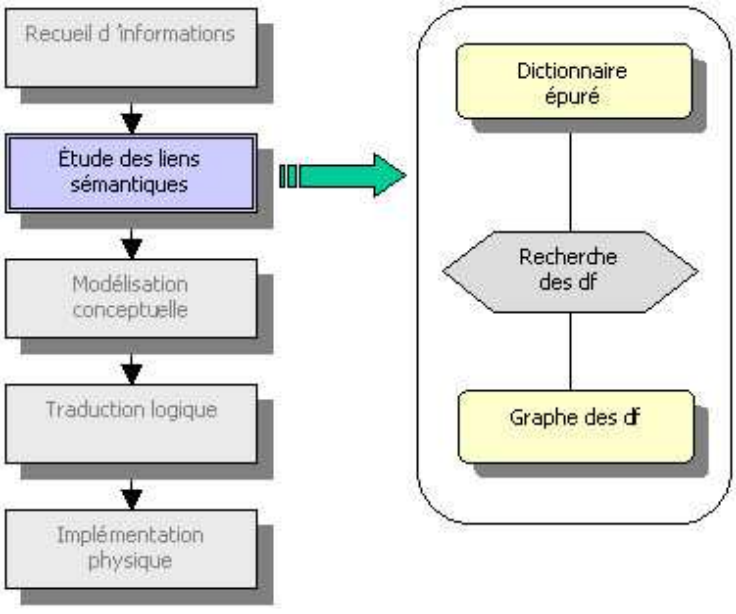
Cours 1

Sommaire

1 Introduction

Cette analyse des dépendances est la deuxième phase de l'analyse des données. Elle a pour but de préparer la phase suivante, à savoir le MCD en recherchant les liens entre les différentes données également appelés les dépendances. L'analyse de ces liens produira comme document un diagramme ou un graphe des dépendances. Cet outil est assez peu utilisé dans le monde professionnel, le placement des données étant réalisé in-situ au moment de la conception du MCD. Cette étape reste néanmoins importante en phase de démarrage dans l'analyse des données.



2 Les notions de dépendances fonctionnelles

Soient A et B les ensembles de valeurs prises par deux données. **Il y a dépendance fonctionnelle entre A et B lorsque, connaissant une valeur de A, quelque soit cette valeur, on détermine une et une seule valeur de B.** Dans ce qui suit, dépendance fonctionnelle sera abrégée par DF

On symbolise la dépendance fonctionnelle par A $\rightarrow$ B où

- A est appelé source de la DF (on dit aussi déterminant ou partie gauche)
- et B la cible (on dit aussi but, déterminé ou partie droite) de la DF.

Exemple :

NumClient → NomClient
NumCLient → AdresseClient
RefProduit → LibProduit

Cela signifie qu'à un numéro de client ne correspondent qu'un et un seul nom et une et une seule adresse. En revanche, il n'est pas impossible qu'à une adresse, résident plusieurs clients, d'où l'importance de bien définir le sens de la dépendance.

Par contre, la DF *RefProduit → NumClient* n'est pas avérée. En effet, un produit peut avoir été commandé par plusieurs clients. Il faut dans ce cas de plus invoquer la notion de commande.

2.1 Dépendance fonctionnelle forte et faible

La définition de la dépendance fonctionnelle peut être affinée :

- Définition stricte → **DF forte :**
  - la DF associe à chaque valeur de A une et une seule de B : il y a **unicité** au départ
  - la DF est vérifiée pour toutes les valeurs de A : il y a **totalité** au départ (toutes les valeurs de A ont une image dans l'ensemble d'arrivée B)

Exemple :
la dépendance fonctionnelle **NumCommande → NumClient** est une DF forte car il n'y a pas de commande sans client.

- Définition large → **DF faible** :
  - Il y a dépendance fonctionnelle entre A et B lorsque, connaissant une valeur de A, quelque soit cette valeur, on détermine au plus une valeur de B.
  - Cette définition supprime la contrainte de totalité au départ. On parle de **DF faible**. Certaines valeurs de A n'ont pas de valeurs de B

Exemple :
La dépendance fonctionnelle **NoInsee → NomJeuneFille** est une DF faible car certaines valeurs de *NoInsee* n'ont pas de correspondance dans l'ensemble d'arrivée ; c'est le cas pour les hommes pour lesquels la propriété *NomJeuneFille* n'a pas de sens.

2.2 Dépendance fonctionnelle à partie gauche composée

Une dépendance fonctionnelle peut comporter dans sa partie gauche plusieurs attributs. On parle dans ce cas de **dépendance fonctionnelle à partie gauche composée**. Pour connaître une valeur de l'ensemble d'arrivée C, il faut connaître un couple (ou plus) de valeurs provenant de A et de B.
Ce type de DF est noté : $(d_1, d_2) \rightarrow d_3$

Exemples :

(NoFacture, CodeProduit) → QtéFacturée, (NoElève, Matière, Date) → Note

Je ne peux connaître la note de Pierre en Français pour le devoir du 10 mai que si je connais ces trois éléments : Le code de l'élève , la matière concernée et la date. S'il manque l'un quelconque de ces éléments, je ne peux déterminer avec exactitude la note correspondante.

2.3 Dépendance fonctionnelle élémentaire

Une dépendance fonctionnelle est élémentaire s'il n'existe aucune donnée ou sous-ensemble de données de la partie gauche assurant une dépendance fonctionnelle vers le même but. Autrement dit, il ne doit pas y avoir de propriété superflue dans la source de la DF.
Par définition les dépendance fonctionnelle à deux rubriques (A→ B) sont toujours élémentaires.

Exemples :
RefProduit → LibProduit est élémentaire (deux rubriques)
(*NumFacture, RefProduit*) → *QtéFacturée* est élémentaire (ni la référence produit seule, ni le numéro de facture seul permettent de déterminer la quantité)
(*NumFacture, RefProduit*) → *LibProduit* n'est pas élémentaire puisque la référence du produit suffit à déterminer le libellé.

2.4 Dépendance fonctionnelle directe

Une dépendance fonctionnelle $d_1 \rightarrow d_2$ est directe s'il n'existe aucune donnée d_3 qui engendrerait une dépendance fonctionnelle **transitive** telle que $d_1 \rightarrow d_2 \rightarrow d_3$

Par exemple, soient les dépendances fonctionnelles :
NumFacture → NumReprésentant et ***NumReprésentant → NomReprésentant***
NumFacture → NomReprésentant n'est pas une dépendance fonctionnelle directe puisqu'elle est obtenue par transitivité. Il conviendra alors de ne considérer que la premiére DF.

2.5 Dépendances fonctionnelles symétriques

Certaines dépendances fonctionnelles sont symétriques, c'est à dire que la partie gauche détermine la partie droite et inversement.

Par exemple :
NoSérieVéhicule → NoImmatriculation et ***NoImmatriculation → NoSérieVéhicule***

Dans ce cas, il faut choisir de privilégier une des dépendances fonctionnelles, en fonction des règles de gestion.
S'il s'agit d'assurer le suivi du véhicule tout au long de sa vie, le no d'Immatriculation pouvant changer, on choisira la premiére DF (***NoSérieVéhicule → NoImmatriculation)***.

3 La recherche et la formalisation des dépendances fonctionnelles

La recherche passe par deux phases, à savoir quels sont les objets du domaine de gestion observé, quels éléments du dictionnaire des données sont rattachés à cet objet, puis ensuite l'analyse des DF entre ces éléments. Le tout sera ensuite formalisé dans un diagramme ou un graphe des DF.

3.1 La recherche des objets

Un objet est un élément du système d'information pourvu d'une existence propre, conforme aux règles de gestion de l'organisation. Le repérage des objets de gestion permet ensuite de faciliter la recherche des DF et la construction du diagramme des DF.

Exemple :
Les véhicules …. font l'objet d'un suivi dans des garages …
Ce texte fait apparaître 2 objets qui sont VEHICULE et GARAGE.

3.2 La recherche des dépendances fonctionnelles

Un objet est représenté par une donnée particulière : l'identifiant. Par définition, l'identifiant est en dépendance fonctionnelle avec toutes les autres propriétés de l'objet.
Repérer les objets du système d'information permet d'avancer très vite dans l'étude des dépendances fonctionnelles. La démarche consistera alors à partir du dictionnaire des données et du repérage des identifiants à ***rechercher les DF élémentaires et directes***

Exemple :

Les véhicules sont repérés par leur numéro d'immatriculation et caractérisé par une couleur…
Le dictionnaire des données comprendra comme information Immat et couleur. Immat étant repéré comme un identifiant, on en déduira la DF Immat → Couleur

3.3 La représentation des dépendances fonctionnelles

Cette représentation se fait à l'aide de deux outils qui sont la matrice ou le graphe des DF. Le graphe des DF permettant de mieux représenter les liens, et surtout les DF à partie gauche composée.

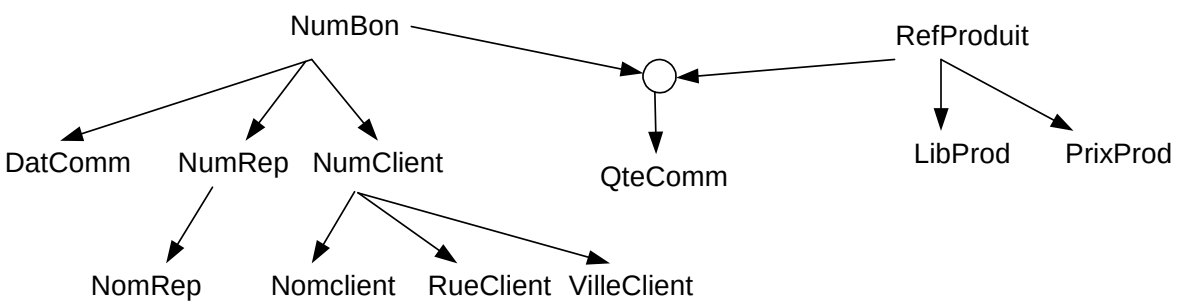
Exemple à partir d'une gestion de commande réduite (Matrice puis graphe des DF)

No	Propriété	Signification	1	2	3	4	5	6	7	8	9	10	11	12
1	NumBon	No Bon Commande	■	X	X				X					C
2	DatComm	DateCommande		■										
3	NumClient	No de Client			■	X	X	X						
4	NomClient					■								
5	RueClient						■							
6	VilleClient							■						
7	NumRep	No représetant							■	X				
8	NomRep	Nom représentant								■				
9	RefProduit	Référence Produit									■	X	X	C
10	LibProduit	Nom Produit										■		
11	PrixProd	PrixProduit											■	
12	QteComm	Qte Commandée												■

DF élémentaire et directe

DF partie gauche composée

NumBon → DatComm, NumClient, NumRep
NumClient → Nomclient, RueClient, Villeclient
NumRep → NomRep
RefProduit → LibProd, PrixProd
NumBon, RefProd → QteComm



Cours 2

Analyse de dépendances et correction de réseaux de preuve

Marc Bagnol[1] & Amina Doumane[2] & Alexis Saurin[3]

1 : IML, Université d'Aix-Marseille, bagnol@iml.univ-mrs.fr
2 : PPS, Université Paris Diderot & École centrale Paris, amina.doumane@student.ecp.fr
3 : PPS, CNRS, Université Paris Diderot & INRIA, alexis.saurin@pps.univ-paris-diderot.fr

Ce travail a bénéficié du soutien du projet ANR portant la référence ANR-10-BLAN-0213.

1. Introduction

Les fruits de Curry-Howard. Depuis la mise en évidence via la correspondance de Curry-Howard [21], des relations entre démonstrations mathématiques et programmes informatiques, théorie de la programmation et théorie de la démonstration s'enrichissent mutuellement. Parmi les nombreux allers-retours fructueux entre preuves et programmes, la logique linéaire [13] tient une place exemplaire. Girard a introduit le système F [11] alors qu'il travaillait sur l'arithmétique du second ordre. Ce système, redécouvert de manière indépendante par Reynolds [27], fournit un λ-calcul polymorphe (ou λ-calcul du second-ordre).

Une quinzaine d'années plus tard, Girard mit en évidence la sémantique cohérente du système F [12] qui élabore sur la sémantique stable de Berry [2]. Les espaces cohérents ont conduit à la décomposition linéaire à l'origine de LLe [13] : $A \Rightarrow B = !\,A \multimap B$. Cette décomposition linéaire, observée dans la sémantique du système F, se reflète syntaxiquement en un système de déduction bien structuré. S'est alors ouvert un champ original où se sont renouvelés les points de vue sur les preuves et les programmes.

Du côté logique, sont apparus les réseaux de preuve [13, 15, 7], les sémantiques interactives (géométrie de l'interaction, sémantique de jeux, ludique), les logiques polarisées [24], allégées [16] et plus récemment différentielles [10].

Du côté programmation, l'étude de la linéarité a eu des conséquences sur de nombreux problèmes : optimalité et partage [18, 22], interprétations calculatoires de la logique classique [14, 6, 24], systèmes de type fournissant des bornes de complexité [23], lien entre focalisation et langages logiques [1, 25], interprétation logique du filtrage [28, 4].

De la logique linéaire aux réseaux de preuve. La décomposition linéaire mentionnée plus haut fait donc apparaître des connecteurs, dits exponentiels (!, ?), qui contrôler l'usage des hypothèses (via les règles structurelles de contraction et affaiblissement). Une conséquence importante du contrôle des règles structurelles réside dans le fait que les diverses présentations des inférences pour la conjonction et la disjonction, équivalentes en logique classique, ne le sont plus dans le cadre linéaire. On voit alors apparaître deux disjonctions et deux conjonctions, deux constantes logiques pour le vrai et deux pour le faux, répartis en un groupe dit multiplicatif ($\parr$, $\otimes$, 1, $\bot$) et un groupe additif ($\oplus$, &, $\top$, 0). La linéarité permet également de récupérer une négation involutive ($A^{\bot\bot} = A$) et les lois de de Morgan permettent alors de restreindre la négation aux formules atomiques tout en considérant des séquents dont toutes les formules sont à droite. LL possède une bonne notion de fragments logiques : on

peut ainsi considérer le fragment restreint aux connecteurs multiplicatifs (avec ou sans les constantes) appelé MLL ou bien le fragment multiplicatif exponentiel, appelé MELL.

Les réseaux de preuve constituent l'une des innovations les plus originales de la logique linéaire. Fondés de manière essentielle sur la linéarité, les réseaux de preuve constituent une syntaxe graphique pour les preuves constituant des objets de preuve très canoniques [1] et l'élimination des coupures y est particulièrement élégante.

L'élégance et la simplicité des réseaux sont particulièrement frappant dans le fragment multiplicatif et sans unités de la logique linéaire (MLL) sur lequel l'article va se concentrer. Il s'agit d'ailleurs plus d'une simplification de présentation qu'une véritable restriction puisque nos résultats s'étendent tous à MELL de manière directe, capturant ainsi le lambda-calcul typé.

Réseaux de preuve et correction logique. Abandonnant une notation séquentielle au profit d'une représentation de la preuve comme un graphe, les réseaux de preuve s'éloignent de l'idée habituelle qu'on se fait d'une démonstration (typiquement un raisonnement qui se déroule progressivement, de manière ordonnée, de son point de départ jusqu'à sa conclusion, structuré par des lemmes intermédiaires).

L'une des conséquences de cette non-séquentialité est que s'ils contiennent des erreurs de raisonnement (ou paralogismes), celles-ci ne sont pas forcément faciles à détecter, contrairement à la plupart des systèmes de déduction (systèmes de Hilbert, déduction naturelle, calcul des séquents, ...) où la correction logique d'une preuve se vérfie de manière purement locale. On parlera de structure de preuve pour un objet qui ressemble à un réseau mais n'est pas forcément logiquement correct.

C'est le prix à payer pour avoir la syntaxe graphique et les bonnes propriétés des réseaux : on passe en effet d'objets inductifs (les arbres des preuves en calcul des séquents par exemple) à des objets de nature plus géométrique (les structures de preuve) dont les propriétés comme l'acyclicité ou la connexité ne sont plus locales mais sont globales.

Pourtant, les démonstrations sont avant tout (et même avant d'être des objets calculatoires !) des objets qui servent à se forger une conviction et à la transmettre. Reformulé en ces termes, le problème de la correction des structures de preuve se résume à ceci : si l'on communique à quelqu'un une structure de preuve R, il doit disposer des moyens pour être sûr que la structure est logiquement correct et qu'elle ne contient pas un raisonnement erroné [2]. Il faudra éventuellement fournir à cette personne, en même temps qu'on lui communique la structure de preuve, un certificat de correction de cette structure de preuve [3].

On souhaite donc disposer de conditions sur les réseaux, a priori de nature graphique et géométrique, assurant de la correction des réseaux, c'est-à-dire qui permette de discriminer les réseaux de preuve qui sont logiquement corrects de ceux qui contiennent des erreurs de raisonnement.

C'est toute la problématique de la correction des réseaux de preuve auquel cet article se veut une contribution : nous proposons un nouveau critère particulièrement simple qui permet de déterminer si une structure de preuve représente bien une preuve. Notre critère simplifie un critère récemment publié par Mogbil et Naurois.

Organisation de l'article. On commence par rappeler les bases de la logique linéaire et des réseaux de preuve. On rappelle le critère de correction de Danos et Régnier. On considère ensuite le critère

1. Contrairement aux preuves du calcul des séquents qui contiennent beaucoup d'information non pertinente sur l'ordonnancement des règles d'inférence.
2. Ou encore, vu sous l'angle calculatoire, qu'il s'agit d'un programme bien typé...
3. L'idée la plus simple est bien sûr de transmettre une preuve séquentialisée de la structure R, mais on voit vite les limites de cette idée : non seulement parce qu'elle nous fait nous reposer sur le calcul des séquents mais aussi parce que le réseau qu'on souhaite transmettre peut résulter d'une élimination des coupures et dans ce cas on ne dispose pas forcément d'une version séquentialisée de la preuve.

de contractilité qui nous servira dans la dernière partie de l'article. On en vient ensuite au cœur de l'article qui est la notion de graphe de dépendance. On commence par présenter le critère de Mogbil et Naurois puis on discute du rôle de l'interrupteur considéré dans le critère. On élimine la dépendance de notre critère aux interrupteurs en deux étapes. On montre tout d'abord qu'on peut définir une structure de graphe de dépendance sur la structure de preuve elle-même et non plus sur ses graphes de corrections ce qui nous donne une première variante de Mogbil et Naurois. On achève de se libérer des interrupteurs en proposant le critère DepGraph. Finalement, on analyse les relations entre les graphes de dépendance introduits dans cet article et les graphes de dépendance de Mogbil et Naurois.

2. Logique linéaire et réseaux de preuve

MLL. Nous nous restreindrons dans la suite de l'article au fragment multiplicatif de la logique linéaire sans constantes, noté MLL. Les formules de MLL sont construites à partir de la grammaire suivante :

$$\boxed{A, B := X \mid X^{\perp} \mid A \otimes B \mid A \parr B \quad (X \in \mathcal{V})}$$

MLL se présente d'habitude en calcul des séquents : un séquent de MLL est une liste finie non orientée de formules de MLL, noté $\vdash \Gamma$ et une preuve est un arbre dont les nœuds sont étiquetés par $(ax), (cut), (\otimes), (\parr)$ et dont les arêtes sont étiquetées par des séquents, suivant les règles ci-dessous :

$$\boxed{\frac{}{\vdash A, A^{\perp}}\ (ax) \qquad \frac{\vdash \Gamma, A \qquad \vdash A^{\perp}, \Delta}{\vdash \Gamma, \Delta}\ (cut) \qquad \frac{\vdash \Gamma, A \qquad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B}\ (\otimes) \qquad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}\ (\parr)}$$

Remarque. *Si on peut voir MLL comme un fragment de LL, on peut également le voir comme un restriction du calcul des séquents de LK où on interdit les règles structurelles de contraction et d'élimination. Le $\otimes$ doit alors être lu comme la conjonction habituelle $\wedge$ et le $\parr$ comme la disjonction $\vee$. Ce parallèle pourra être utile au lecteur qui n'est pas familier avec la logique linéaire.*

Structures et réseaux de preuve. Les structures de preuve sont une nouvelle syntaxe pour la logique linéaire. Comme nous ne nous restreignons qu'au fragment MLL, nous allons présenter uniquement les structures de preuve pour ce fragment. Il existe dans la littérature des notions de structures de preuve qui correspondent aux autres fragments de la logique linéaire.

Définition 2.1. ***Structure de preuve.*** *On appelle structure de preuve un graphe orienté fini dont les sommets (appelés aussi nœuds) sont étiquetés soit par des connecteurs de la logique linéaire $(\parr, \otimes)$ soit par **ax** (pour axiome) ou **cut** (pour coupure) ou **c** (pour conclusion) et les arêtes sont étiquetées par des formules de la logique linéaire. Les sommets et les arêtes vérifient de plus les propositions suivantes :*

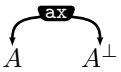
- *Les nœuds étiquetés par $\otimes$ (resp. $\parr$) ont deux prémisses et une conclusion. Si l'étiquette de la première prémisse est A et celle de la deuxième prémisse est B alors l'étiquette de la conclusion est $A \otimes B$ (resp. $A \parr B$) ;*
- *Les nœuds étiquetés par **ax** n'ont aucune prémisse et ont deux conclusions. Si l'étiquette de la première conclusion est A alors celle de la deuxième conclusion est $A^{\perp}$;*
- *Les nœuds étiquetés par **cut** ont deux prémisses et n'ont aucune conclusion. Si l'étiquette de la première prémisse est A alors celle de la deuxième prémisse est $A^{\perp}$;*
- *Les nœuds étiquetés par **c** ont une prémisse et n'ont aucune conclusion*[4].

4. Pour alléger la représentation graphique des réseaux, on laissera ces liens implicites dans les figures.

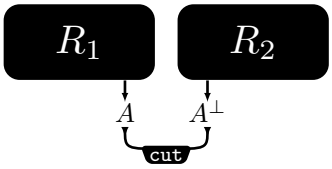
À chaque preuve dans MLL on va associer une structure de preuve. On appellera *réseaux de preuve* le sous-ensemble des structures de preuve qui proviennent de dérivations MLL.

Définition 2.2. ***Réseaux de preuve.*** *On définit par induction sur la dernière règle utilisée la structure de preuve correspondant à une preuve de MLL.*

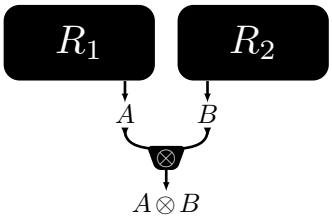
- *règle* ***ax*** *: le réseau de preuve correspondant à $\vdash A, A^\perp$ est le graphe contenant un nœud ax dont les arêtes sortantes-étiquetées par A et $A^\perp$ sont reliées ces nœuds conclusion :*



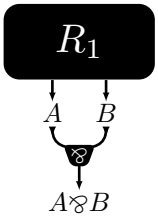
- *règle* ***cut*** *: si R_1 et R_2 sont les réseaux de preuve associés aux deux prémisses de la règle, le réseau de preuve associé à la preuve complète est obtenu en ajoutant un nœud* ***cut*** *entre les arêtes correspondant aux occurrences de formules impliquées dans la règle :*



- *règle $\otimes$: si R_1 et R_2 sont les réseaux de preuve associés aux deux prémisses de la règle, le réseau de preuve associé à la preuve complète est obtenu en ajoutant un nœud $\otimes$ entre les arêtes correspondant aux occurrences de formules impliquées dans la règle, et en reliant l'arête sortante à un nœud conclusion :*



- *règle $\parr$: si R_1 est le réseau de preuve associé à la prémisse de la règle, le réseau de preuve associé à la preuve complète est obtenu en ajoutant un nœud $\parr$ entre les arêtes correspondant aux occurrences de formules impliquées dans la règle, et en reliant l'arête sortante à un nœud conclusion :*



Le graphe de la figure 1 est bien une structure de preuve, pourtant il ne peut être associé à une preuve dans MLL. Une structure de preuve ne correspond donc pas nécessairement à une preuve en calcul des séquents et une telle structure est dite non séquentialisable. Pour distinguer les structures de preuve séquentialisables — les réseaux de preuve — de celles qui ne le sont pas, il existe de nombreux résultats décrivant des méthodes pour faire cette distinction, sous le nom de *critères de correction*.

Remarque. *Dans la suite de l'article, nous ne considérerons plus que des réseaux sans coupures. La coupure se comporte en effet exactement comme le $\otimes$ du point de vue de la correction et n'introduit donc ni difficulté ni intérêt particuliers.*

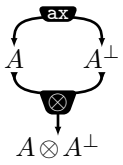


FIGURE 1 – UNE STRUCTURE DE PREUVE QUI N'EST PAS UN RÉSEAU.

3. Critères de correction

Plusieurs critères de correction pour les structures de preuve ont été introduits pour distinguer, parmi les structures de preuve, celle qui sont des réseaux de preuve de celles qui ne sont pas correctes. On compte notamment le critère original des longs voyages (LV) [13], le critère de Danos-Regnier (DR) [7], le critère des contre-preuves (CP) [3, 17], le critère de contractilité [5], le critère de parsing [19, 20] et, plus récemment apparu, le critère de Mogbil-Naurois [8].

Dans cette section, nous allons présenter les critères Danos-Regnier et contractilité qui seront utilisés ultérieurement. Le critère de Mogbil et Naurois sera introduit dans la section suivante. Le reste de l'article consistera en l'introduction d'un nouveau critère de correction, DepGraph, basée sur le critère de Mogbil et Naurois.

3.1. Critère de Danos-Regnier

Définition 3.1. ***Interrupteur.*** *On appelle interrupteur (ou switching) d'une structure de preuve le choix pour chacun de ses nœuds $\parr$ de l'une des deux prémisses. De chaque nœud $\parr$, on dit qu'il est switché à droite (resp. à gauche), si on a choisi la prémisse droite (resp. gauche).*

Définition 3.2. ***Graphe de correction.*** *On appelle graphe de correction $\mathcal{S}(R)$ d'une structure de preuve R et d'un interrupteur $\mathcal{S}$ le graphe dont :*

1. *Les sommets sont ceux de la structure de preuve.*
2. *Les arêtes sont les mêmes que celles de la structure de preuve où on a supprimé l'arête gauche (resp. droite) d'un nœud $\parr$ s'il a été switché à droite (resp. à gauche).*
3. *Les étiquettes sont les mêmes que dans la structure de preuve.*

Définition 3.3. ***Critère de Danos-Regnier(DR).*** *On dit qu'une structure de preuve vérifie le critère de Danos-Regnier si tous les graphes de correction associés à des interrupteur sont connexes et acycliques. On dit aussi qu'elle est DR-correcte.*

Théorème 3.4. *Un structure de preuve est un réseau de preuve si et seulement si elle est DR-correcte.*

Le critère de Danos-Regnier est un critère qui ne fait pas référence directement à la séquentialisabilité d'une structure de preuve en une preuve du calcul des séquents. La simplicité de son énoncé et son élégance en ont fait l'un des critères les plus fameux. En revanche, il n'est pas efficacepour tester la correction d'une structure du preuve puisqu'il faut vérifier la connexité et l'acyclicité des 2^n graphes de correction, où n est le nombre de $\parr$ de la structure, ce qui est exponentiellement chronophage. On ne peut pas espérer améliorer aisément cette borne en se restreignant à un sous-ensemble d'interrupteurs bien choisis : on peut en effet montrer qu'il existe des structures de preuves ayant un nombre arbitrairement grand de $\parr$, dont tous les graphes de correction sont connexes et acycliques sauf un. À moins de disposer d'une information très spécifique sur la topologie du réseau (par exemple si le graphe est planaire, on peut se restreindre à ne tester que deux interrupteurs pour décider de la correction d'une structure de preuve [26]).

Obtenir des algorithmes efficaces pour tester la correction d'une structure de preuve est donc un enjeu important pour rendre utilisable cette représentation des preuves et, logiquement, de nombreuses recherches ont été consacrées à produire des critères dont les complexités étaient de plus en plus faibles. Ainsi, par exmeple, le critère de contractilité [5] a-t-il une complexité quadratique tandis que le critère de parsing [20] peut être testé en temps linéaire. Dans la section suivante, nous présentons le critère de contractilité qui nous servira dans la suite de notre développement.

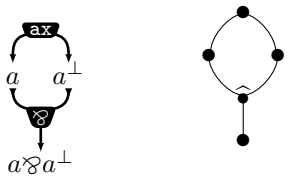
3.2. Critère de contractilité

Le critère de Danos-Regnier reposait déjà sur des propriétés topologiques des structures de preuve, plus exactement de leur graphes de correction. Le critère que l'on introduit maintenant exprime quant à lui directement une propriété topologique de la structure de preuve (ou plus exactement du graphe apparié qui lui est associé, qui contient juste l'information nécessaire pour distinguer les arêtes prémisses d'un $\parr$ des autres arêtes de la structure) en terme de contractilité.

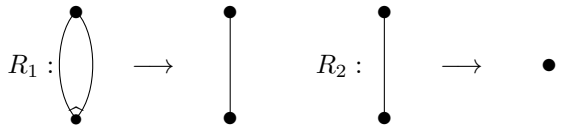
Définition 3.5. ***Graphe apparié.*** *Un graphe apparié est la donnée d'un graphe $G = (V, E)$ et d'un ensemble $P(G)$ de paires (non orientées) d'éléments de E qui ont au moins une extrémité commune. L'extrémité commune de deux arêtes appariées est appelée un nœud apparié.*

Définition 3.6. $C(R)$**.** *Soit R une structure de preuve. Le graphe apparié qui lui est associé, qu'on note $C(R)$ est le graphe R muni de l'ensemble des paires d'arêtes qui sont prémisses d'un nœud $\parr$.*

Exemple. *On donne ci-dessous l'unique réseau de preuve $R_{a\parr a^\perp}$ pour le séquent $\vdash a \parr a^\perp$ et le graphe apparié $C(R_{a\parr a^\perp})$ qui lui est associé (les arêtes appariées sont distinguées par un $\frown$) :*



Définition 3.7. ***Règles de contraction.*** *On définit les règles de réécriture sur les graphes appariés suivantes (on notera que dans la règle R_2, on demande que les deux nœuds sont distincts) :*



Définition 3.8. ***Contractilité.*** *On dit qu'une structure de preuve R est contractile si $C(R) \to^* \bullet$.*

La contractilité caractérise les réseaux de preuve et fournit donc un critère de correction :

Théorème 3.9. *Une structure de preuve est un réseau de preuve si et seulement si elle est contractile.*

4. Le critère du graphe de dépendance

On introduit maintenant le critère de Mogbil et Naurois qui repose :

1. sur l'existence d'un interrupteur dont le graphe de correction est connexe et acyclique et
2. sur le fait que le *graphe de dépendance* construit à partir de cet interrupteur soit un graphe acyclique et possédant une source.

On cherche ensuite à s'abstraire complètement de l'interrupteur, ce qui se fait en deux étapes. Dans la section 4.2, on commence par définir une nouvelle notion de graphe de dépendance qui ne dépend pas du choix d'un interrupteur, on obtient ainsi une condition nécessaire pour qu'une structure de preuve soit séquentialisable. Finalement, on se passera complètement de l'interrupteur dans la section 4.3 où l'on analysera l'utilité véritable de la première partie du critère dont nous montrerons qu'elle peut être remplacée par une condition nécessaire beaucoup plus simple qui ne dépend pas des interrupteurs. La conjonction de ces deux conditions nécessaires fournit en fait une condition suffisante comme nous le montrons dans la section 4.4. On conclut la section en comparant en 4.5 les graphes de dépendance à la Mogbil-Naurois et les graphes de dépendance que nous venons d'introduire.

4.1. Le critère de Mogbil et Naurois

Le critère de Mogbil et Naurois a été introduit pour montrer que la correction des structures de preuve MLL était NL-complète [5].

Définition 4.1. ***Chemin élémentaire.*** *Un chemin dans un graphe non orienté sera dit élémentaire s'il ne passe pas deux fois par la même arête.*

Définition 4.2. ***Graphe de dépendance d'un graphe de correction.*** *Le graphe de dépendance d'un graphe de correction provenant d'un interrupteur $\mathcal{S}$ d'une structure de preuve R, noté $D(\mathcal{S}, R)$ est un graphe orienté (S, A) défini comme suit :*

- *L'ensemble des nœuds S est constitué de l'ensemble des conclusions des nœuds $\parr$ de R et d'un nœud s appelé source.*
- *Soit x un nœud $\parr$ dans R, x_d et x_g ses prémisses droite et gauche respectivement dans R.*
  - *Il y a une arête $(s \to x)$ dans A s'il existe un chemin élémentaire $x_g, \ldots, x_d$ dans $\mathcal{S}$ qui ne passe par aucun nœud $\parr$.*
  - *Soit y un autre nœud $\parr$ de R. Il y a une arête $(y \to x)$ s'il existe un chemin élémentaire $x_g, \ldots, x_d$ dans $\mathcal{S}$ qui contient y.*

Définition 4.3. ***Graphe*** *SDAG.* *Un graphe G est dit SDAG si :*

- *il est acyclique ;*
- *il contient un nœud s, nommé nœud source, tel que, pour chaque nœud n de G différent de s, il existe un chemin de s vers n.*

Définition 4.4. ***Critère de Mogbil-Naurois.*** *Une structure de preuve vérifie le critère de Mogbil-Naurois (MN) si, et seulement si, il existe un interrupteur $\mathcal{S}$ tel que :*

- *$D(\mathcal{S}, R)$ est SDAG ;*
- *$\mathcal{S}$ est connexe et acyclique.*

Theorem 4.5. *Une structure de preuve est séquentialisable si, et seulement si, elle vérifie (MN).*

Les figures 2 et 3 illustrent comment ce critère discrimine les structures correctes et incorrectes. On remarque sur la figure 3 que le graphe de dépendance dépend de l'interrupteur, on reviendra sur ce point dans la section 4.5.

5. NL désigne la classe des problèmes qui peuvent être décidés en espace logarithmique par une machine de turing non déterministe.

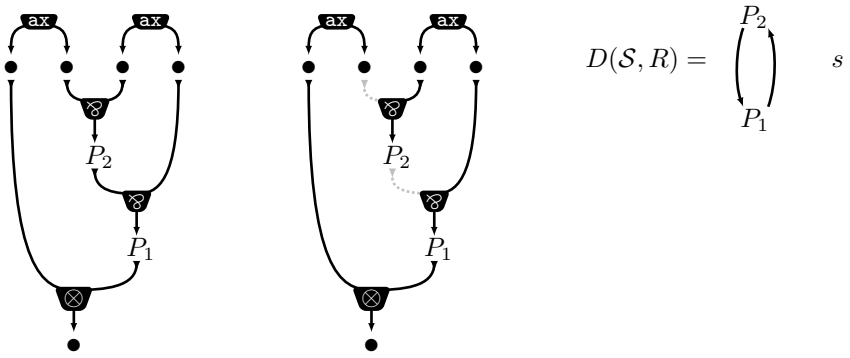


FIGURE 2 – UNE STRUCTURE INCORRECTE R, UN INTERRUPTEUR $\mathcal{S}$ DE R ET LE GRAPHE DE DÉPENDANCE (CYCLIQUE ET SANS SOURCE) ASSOCIÉ.

Par rapport à Danos-Regnier, le critère de Mogbil et Naurois possède une caractéristique troublante : il ne repose que sur l'analyse d'un interrupteur et du graphe de correction induit par cet interrupteur, d'autant que le choix de l'interrupteur est lui-même arbitraire.

Il semble donc naturel de se demander la véritable raison d'être de cet interrupteur : À quoi sert-il ? En a-t-on vraiment besoin ? Nous répondons à ces deux questions dans la suite en énonçant un critère de correction à base de graphe de dépendance qui ne dépende par d'un interrupteur.

4.2. Un graphe de dépendance qui ne dépend pas des interrupteurs

On définit une notion de graphe de dépendance directement à partir d'une structure de preuve et non plus à partir de ses graphes de correction.

Définition 4.6. ***s-chemin dans une structure de preuve.*** *Dans une structure de preuve, un chemin est appelé s-chemin s'il ne passe pas successivement par les deux prémisses d'un nœud $\parr$ et s'il est élémentaire.*

Définition 4.7. ***Graphe de dépendance d'une structure de preuve.*** *Le graphe de dépendance d'une structure de preuve R, noté $D(R)$ est un graphe orienté (S, A) défini comme suit :*

- *L'ensemble des nœuds S est constitué de l'ensemble des conclusions des nœuds $\parr$ de R et d'un nœud s appelé source.*
- *Soit x un nœud $\parr$ dans R, x_d et x_g ses prémisses droite et gauche respectivement dans R.*
  - *Il y a une arrête $(s \to x)$ dans A s'il existe un s-chemin $x_g, \ldots, x_d$ dans R qui ne passe par aucun nœud $\parr$.*
  - *Soit y un autre nœud $\parr$ de R. Il y a une arête $(y \to x)$ s'il existe un s-chemin $x_g, \ldots, x_d$ dans R qui contient y.*

Définition 4.8. ***Critère de Mogbil-Naurois modifié.*** *Une structure de preuve vérifie le critère de Mogbil-Naurois modifié (MN') si, et seulement si :*

- *son graphe de dépendance $D(R)$ est SDAG de source s ;*
- *un de ses graphes de correction est connexe et acyclique.*

Le condition (MN') fournit une condition nécéssaire pour qu'une structure soit séquentialisable :

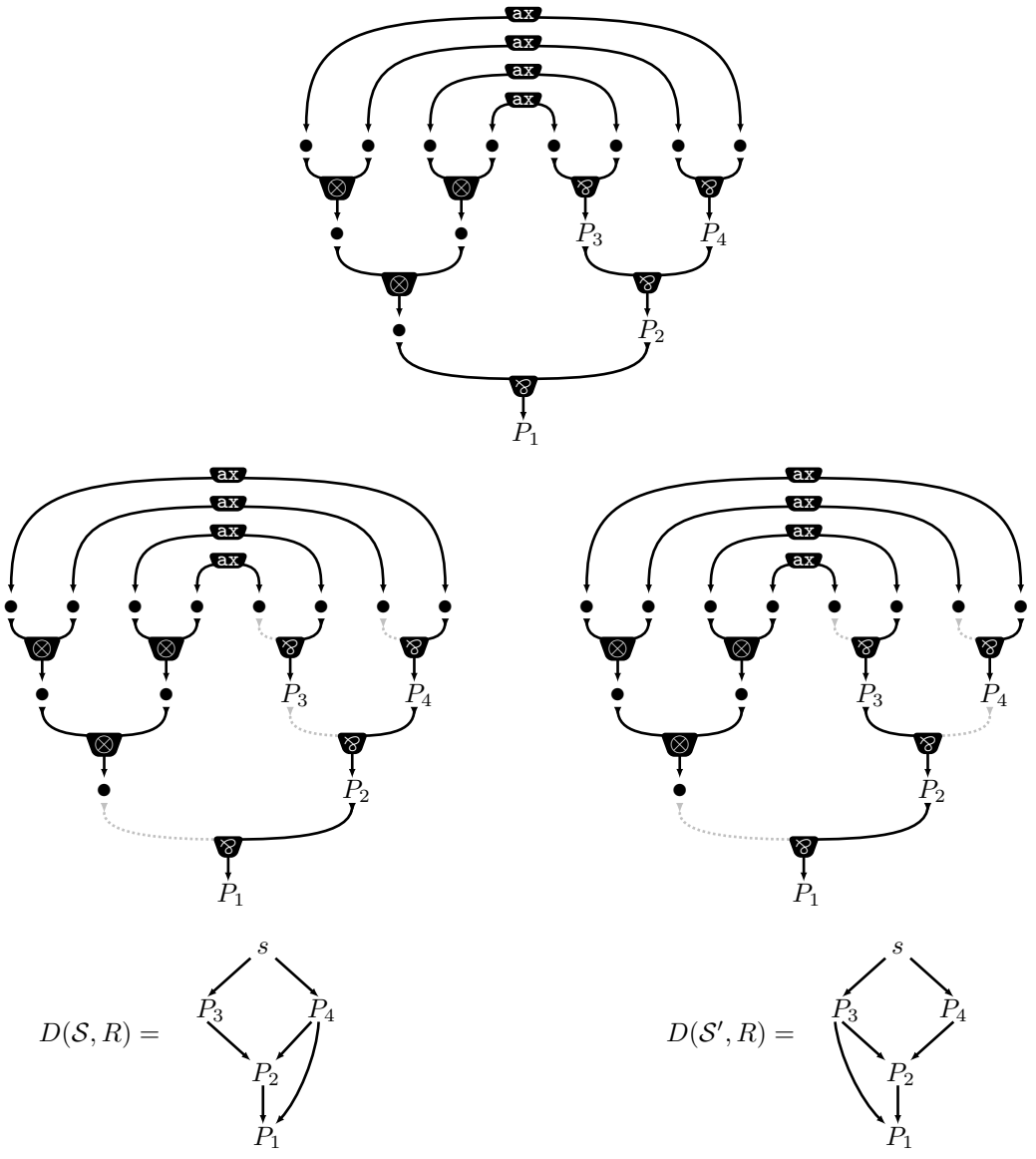


FIGURE 3 – UN RÉSEAU R, DEUX INTERRUPTEURS $\mathcal{S}$ ET $\mathcal{S}'$ DE R ET LES GRAPHE DE DÉPENDANCE ASSOCIÉS.

Theorem 4.9. *Une structure de preuve séquentialisable satisfait (MN').*

Pour montrer ce résultat, on s'appuie sur une relation entre les $\parr$ induite par l'ordre dans lequel les inférences sont utilisées dans une des séquentialisations de R qui nous donnera l'acyclicité du graphe de dépendance. Une fois l'acyclicité obtenue, l'existence d'une source découle du fait qu'au plus un nœud ne possède pas de parent.

Définition 4.10. ***Ordre d'introduction.*** *Soit π une preuve MLL. On définit un ordre sur les formules introduites par une règle $\parr$ dans π, qu'on notera $<_\pi$, comme suit :*

$$F <_\pi G \text{ si } F \text{ a été introduite au dessus de } G \text{ dans l'arbre de } \pi.$$

On notera le graphe de cette relation $O(\pi)$.

Théorème 4.11. *Soit π une preuve MLL et R le réseau de preuve correspondant. On a :*

$$D(R) \subseteq O(\pi)$$

Démonstration. Voir en annexe. □

Corollaire 4.12. *Soit π une preuve MLL. $O(\pi)$ est acyclique.*

Démonstration. Immédiat. □

Lemme 4.13. *Soit G un graphe dirigé acyclique. Si chaque nœud de G a un parent, sauf un seul nœud, noté s, alors G est $SDAG$ de source s.*

On peut maintenant passer à la preuve du théorème 4.9 :

Démonstration. Soit R un réseau de preuve.

- Par Danos-Régnier, il est évident que tout graphe de correction est connexe et acyclique.
- Comme R est un réseau de preuve, il existe une preuve π dont R est le réseau. Par le théorème 4.11, on a que $D(R) \subseteq O(\pi)$. Mais $O(\pi)$ est acyclique par le corollaire 4.12. On conclut alors que $D(R)$ est acyclique. D'autre part, comme R est un réseau de preuve, par le critère de Danos-Regnier, tous ses interrupteurs sont connexes. Soit G un graphe de correction de R. Comme il est connexe, les deux prémisses de chaque nœud $\parr$ sont reliées par un chemin dans G, qui est un s-chemin dans R. Il s'en suit que dans $D(R)$, chaque nœud - à part le nœud s- a un parent. Le lemme 4.13 nous assure donc que $D(R)$ est $SDAG$.

□

Remarque. *Le critère MN' n'est pas seulement une condition nécessaire : il s'agit bien d'un critère de correction[9]. Nous n'allons cependant pas plus loin dans l'étude de cette variante de Mogbil et Naurois car nous sommes ici intéressé par un critère qui s'affranchisse véritablement des interrupteurs.*

4.3. (In)utilité du témoin d'acyclicité et de connexité

On se pose ensuite la question du témoin d'acyclicité et de connexité fournit par l'interrupteur et on arrive au fait que si un interrupteur vérifie cela, alors pour tous les interrupteurs, on a une composante connexe de plus qu'il n'y a de cycles et on en extrait la version affaiblie avec l'invariant sur $\#ax - \#tenseurs$. Cela nous donne le dernier critère. On remarque que connexité et la caractérisation axiome/tenseurs sont évidemment nécessaire et donc il ne nous reste plus qu'à montrer que Dep-Graph est suffisant pour assurer de la séquentialisation.

Théorème 4.14. *Soit G un graphe et soient respectivement n, a, c et b ses nombres de nœuds, arêtes, composantes connexes et cycles. On a l'égalité suivante : $n - a = c - b$.*

Définition 4.15. *On notera χ_G la valeur caractéristique du graphe fournit par 4.14.*

Proposition 4.16. *Pour tout graphe de corrections G d'une structure correcte, on a $\chi_G = 1$.*

Démonstration. Immédiat par connexité et acyclicité. □

Proposition 4.17. *Tout graphe de correction G d'une structure R de preuve vérifie : $\chi_G = \#ax - \#\otimes$. On notera χ_R ce nombre, la caractéristique de la structure.*

Démonstration. On fait un simple dénombrement dont le résultat découle immédiatement :

- $n = \#ax + \#\otimes + \#\parr + \#concl$;
- Pour dénombrer les arêtes, on remarque que chaque arête est arête entrante d'exactement un nœud du graphe de correction, ce qui nous donne : $a = 0 \times \#ax + 2 \times \#\otimes + \#\parr + \#concl$.

□

La conjonction des deux propositions précédentes nous indique donc que pour qu'une structure de preuve soit séquentialisable il faut que la différence du nombre de liens axiome et du nombre de liens tenseur soit égale à un ($\#ax - \#\otimes = 1$).

Nous avons maintenant les outils pour énoncer notre nouveau critère, *DepGraph* :

Définition 4.18. ***Critère de graphe de dépendance.*** *Une structure de preuve G vérifie le critère de graphe de dépendance (D-correcte) si :*

- *$D(G)$ est SDAG ;*
- *G est connexe ;*
- *La caractéristique (définie dans la proposition 4.17) de G vaut 1.*

4.4. DepGraph est un critère de correction

Théorème 4.19. *Si une structure de preuve est D-correcte alors elle est un réseau de preuve.*

Pour montrer qu'une structure de preuve D-correcte est un réseau de preuve, nous allons montrer qu'elle vérifie le critère de contractilité (ie. son graphe apparié est contractile). Pour ce faire, nous avons besoin les lemmes suivants :

Lemme 4.20. *La connexité est présérvée par les étapes de contractilité.*

Lemme 4.21. *La quantité $n - a_{nap} - a_{ap}$ (où n est le nombre de nœuds du graphe, a_{nap} est son nombre d'arêtes non-appariées et a_{ap} est son nombre de paires d'arêtes appariés) est conservée par les étapes de contractilité.*

On définit un s-chemin dans une un graphe apparié G comme étant un chemin élémentaire qui ne contient pas deux arêtes appariés entre elles. Si p est un s-chemin, on le note aussi $p\{a_1, \ldots, a_n\}$ pour indiquer les arêtes appariées par lesquelles il passe.

On définit les prémisses d'un nœud apparié n via les arêtes a, a' comme étant les nœuds reliés à n par ces arêtes appariées.

La notion de graphe de dépendance est aussi transportée pour les graphes appariés : les nœuds du graphe de dépendance $D(G)$ d'un graphe apparié G sont les nœuds appariés de G et un nœud supplémentaire s. De plus, si x est un nœud apparié,

- Il y a une arrête $(s \to x)$ dans $D(G)$ s'il existe un s-chemin $p\emptyset$ dans G entre les deux nœuds prémisses de x.
- Soit y un autre nœud apparié. Il y a une arête $(y \to x)$ s'il existe un s-chemin dans G qui contient l'une des arêtes appariées de y.

Lemme 4.22. *Soit G un graphe apparié et G' tel que $G \to^\star G'$. Soient $\{b_1, \dots, b_m\}$ les arêtes appariées qui ont été contractées. Soient x et y deux nœuds dans G. Si dans G, x et y sont reliées par un s-chemin $p\{a_1, \dots, a_n\}$, alors :*

- *soit les nœuds x et y ont été fusionnés*
- *sinon il existe un s-chemin $q\{a_1, \dots, a_n\} \setminus \{b_1, \dots, b_m\}$ qui relie x et y dans G'.*

Réciproquement, tout s-chemin de G' provient de la contraction d'un s-chemin de G.

Démonstration. On montre le résultat pour $\to$ et on l'étend ensuite à $\to^\star$. Le raisonnement s'effectue en considérant les différentes configurations possibles pour la contraction des arêtes de G et ne pose pas de problème. □

Retour à la preuve du théorème :

Démonstration. Soit R une structure de preuve et $G = C(R)$ son graphe apparié. Nous allons construire une suite de contractions $c_1, \dots, c_n$ telles que, si on pose : $G := G_0 \to^\star_{c_1} G_1 \cdots \to^\star_{c_n} G_n$, on a :

- Le graphe de dépendance de G_{i+1} est le graphe de dépendance de G_i, où un nœud n directement relié à la source a été enlevé et où les nœuds dont le seul parent est n sont reliés à la source par une arête. On montre facilement que le caractère SDAG du graphe de dépendance est invariant par cette transformation.
- Le graphe de dépendance G_n contient l'unique nœuds s, G_n ne contient donc aucun nœud apparié.

Supposons qu'on a construit le i^{me} graphe G_i. Soit $D(G_i)$ son graphe de dépendance. Soit n un nœud directement lié à la source. Il existe un chemin $p\emptyset$ qui relie les deux nœuds prémisses de n. En contractant toutes les arêtes de $p\emptyset$ (par l'opération r_2), les deux nœuds prémisses de n se retrouvent fusionnés, on peut alors appliquer r_1 sur les arêtes appariées de n. On note cette séquence de contractions c_{i+1}, et on note G_{i+1} le graphe obtenu à partir de G_i en appliquant c_{i+1} . Analysons l'effet de cette suite de contractions sur les s-chemins de G_i : comme les seules arêtes appariées qui ont été contractées sont a_1 et a_2 qui sont les arêtes appariés de n, par le lemme 4.22, pour tout nœud apparié x, si $p\{E\}$ est un s-chemin qui relie les deux nœuds prémisses de x, alors il existe un s-chemin $q\{E\} \setminus \{a_1, a_2\}$.

Il ya deux cas possibles :

- Si x avait n comme unique parent dans $D(G_i)$, cela signifie que tous les s-chemins qui relient les prémisses de x dans G_i (on sait qu'il en existe au moins un) sont de la forme $p\{a_1\}$ ou $p\{a_2\}$. Ces chemins deviennent des s-chemins $q\emptyset$ dans G_{i+1}. x est alors relié à la source dans $D(G_{i+1})$.
- Si x avaient plusieurs parents $\{n_1, \dots, n_k\}$ dans $D(G_i)$, alors ses parents deviennent $\{n_1, \dots, n_k\} \setminus \{n\}$ dans $D(G_{i+1})$.

$D(G_{i+1})$ contient les mêmes nœuds que $D(G_i)$ sauf n et ses arêtes sont les mêmes sauf pour les nœuds qui avaient n comme seul parent et qui sont maintenant reliées à la source.

Comme le graphe de dépendance qu'on obtient à chaque étape est encore SDAG, il existe toujours un nœud directement relié à la source, et de ce fait on peut renouveler cette procédure jusqu'à épuisement des nœuds appariés. On obtient un graphe G_n qui ne contient pas de nœuds appariés. Le graphe G' obtenu en appliquant toutes les contractions r_2 possible est bien défini.
G' est connexe (la contractilité conserve la connexité par le lemme 4.20) et il ne peut donc contenir qu'un nœud (sinon on pourrait appliquer r_2).
Par le lemme 4.21, la quantité $n - a_{nap} - a_{ap}$ est préservée par les étapes de contraction. Comme elle vaut 1 pour le graphe D-correct G, elle vaut aussi 1 pour le graphe G'. G' ne contient pas de nœuds appariés, on a donc $a_{nap} = O$, et il contient qu'un unique nœud, donc $a_{ap} = 0$. G' est donc le graphe réduit a un seul nœud : G est bien contractile. □

4.5. Comparaison entre les deux notions de graphe de dépendance

L'exemple de la figure 3 nous a montré que les graphes de dépendance à la Mogbil-Naurois dépendent des choix d'interrupteur. On va ici montrer que, pour les réseaux de preuve, on a presque cette invariance. plus précisément, les clôtures transitives des graphes des différents interrupteurs (c'est-à-dire l'ordre associé au graphe de dépendance) sont toutes égales et elles sont égales à la clôture transitive du graphe de dépendance que nous avons introduit.

Notations: *si $\mathcal{S}$ est un interrupteur d'une structure de preuve R et a un lien $\parr$ de cette structure, on note $\mathcal{S}_a$ l'interrupteur $\mathcal{S}$ dans lequel on a inversé le positionnement de a.*

Étant donné un graphe D, on notera D^* sa clôture transitive.

Lemme 4.23. *Soient z et a deux liens $\parr$ d'une structure DR-correcte R et un interrupteur $\mathcal{S}$.*

- *si $(z \to a) \in D(\mathcal{S}, R)$, alors $(z \to a) \in D(\mathcal{S}_a, R)$*
- *si $(a \to z) \in D(\mathcal{S}, R)$, alors $(a \to z) \in D(\mathcal{S}_a, R)$*

Démonstration. Le premier point découle du fait que le chemin élémentaire entre les prémisses de a n'est pas affecté par l'inversion de a.

Pour le second point : si ce n'était pas le cas, on aurait un chemin élémentaire entre les prémisses X_g et X_d de x qui ne passe pas par a dans $\mathcal{S}_a(R)$ qui serait également présent dans $\mathcal{S}(R)$, mais par hypothèse on a dans $\mathcal{S}(R)$ un chemin élémentaire entre X_g et X_d qui passe par a, ce qui donnerait deux chemins élémentaires différents entre X_g et X_d dans $\mathcal{S}(R)$, en contradiction avec l'acyclicité. □

Théorème 4.24. *Soit R une structure de preuve DR-correcte. Pour tous $\mathcal{S}, \mathcal{S}'$ interrupteurs de cette structure, on a*

$$D(\mathcal{S}, R)^* = D(\mathcal{S}', R)^*$$

Démonstration. On montre en fait que pour tout interrupteur $\mathcal{S}$ et pour tout lien $\parr$ a de la structure, on a $D(\mathcal{S}, R) \subseteq D(\mathcal{S}_a, R)^*$. Par symétrie et comme $\mathcal{S}' = \mathcal{S}_{a_1 \dots a_n}$ pour une certaine suite $a_1 \dots a_n$ de liens $\parr$, cela est suffisant.

Soient donc R une structure de preuve DR-correcte, un interrupteur $\mathcal{S}$ de R, x et a deux liens $\parr$ de R. Soit également y tel que $(y \to x) \in D(\mathcal{S}, R)$. On note X_g et X_d les deux prémisses de x.

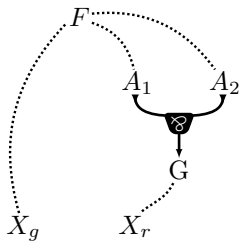
Si y est s, la source de $D(\mathcal{S}, R)$, alors $(a \to x) \notin D(\mathcal{S}, R)$ par définition. Le chemin élémentaire entre X_g et X_d n'est pas affecté par l'inversion de a et ne passe toujours par aucun lien $\parr$, donc $(s \to x) \in D(\mathcal{S}_a, R)$.

On suppose donc que $y \neq s$, c'est à dire que y est un lien $\parr$ du réseau.

Si $(a \to x) \notin D(\mathcal{S}, R)$, le chemin élémentaire entre X_g et X_d n'est pas affecté par l'inversion de a et passe toujours par y, donc $(y \to x) \in D(\mathcal{S}_a, R)$.

Si $(a \to x) \in D(\mathcal{S}, R)$ et $(y \to a) \in D(\mathcal{S}, R)$, le lemme 4.23 implque que $(y \to a) \in D(\mathcal{S}_a, R)$ et $(a \to x) \in D(\mathcal{S}_a, R)$, et donc $(y \to x) \in D(\mathcal{S}_a, R)^*$.

Finalement, si $(a \to x) \in D(\mathcal{S}, R)$ et $(y \to a) \notin D(\mathcal{S}, R)$, on note A_1 et A_2 les deux prémisses de a, telles que le chemin élémentaire entre X_g et X_d passe par A_1 dans $\mathcal{S}(R)$ et par A_2 dans $\mathcal{S}_a(R)$. On note également F et G les points de branchements de ces deux chemins (l'un des deux, disons G sans perte de généralité, étant nécessairement la conclusion de a), comme décrit dans le schéma suivant :



où les lignes pointillées représentent des chemins élémentaires présents à la fois dans $\mathcal{S}(R)$ et $\mathcal{S}_a(R)$.

Comme par hypothèse $(y \to a) \notin D(\mathcal{S}, R)$, y ne peut pas être sur un chemin élémentaire reliant F à A_1 ou F à A_2 dans $\mathcal{S}(R)$. De plus $(y \to x) \in D(\mathcal{S}, R)$, donc y doit être soit sur le chemin élémentaire reliant X_g à F soit sur celui reliant G à X_d dans $\mathcal{S}(R)$.

Ces deux chemins étant toujours présents dans $\mathcal{S}_a(R)$, on a bien $(y \to x) \in D(\mathcal{S}_a, R)$. □

Il ne nous reste plus qu'à montrer la dernière partie du résultat :

Théorème 4.25. *Soit R un réseau et soit $\mathcal{S}$ un interrupteur pour R. On a $D(R)^\star = D(R, \mathcal{S})^\star$.*

Démonstration. En effet, étant donné un interrupteur $\mathcal{S}_0$, on a grace au théorème 4.24

$$D(R, \mathcal{S}_0) \subseteq D(R) = \bigcup_{\mathcal{S} \text{ interrupteur de } R} D(R, \mathcal{S}) \subseteq \bigcup_{\mathcal{S} \text{ interrupteur de } R} D(R, \mathcal{S})^\star = D(R, \mathcal{S}_0)^\star$$

On en déduit $D(R, \mathcal{S}_0)^\star \subseteq D(R)^\star = D(R, \mathcal{S})^{\star\star} = D(R, \mathcal{S}_0)^\star$ ce qui donne l'égalité souhaitée. □

5. Conclusion

Comparaison au critère original Le critère DepGraph défini dans la section 4.2 reprend la notion de graphe de dépendance introduite dans [8] et la rend indépendante de la notion d'interrupteur. Le choix de le formuler dans un cadre très restreint (MLL) est fait pour favoriser l'exposition, mais ces résultats s'étendent à MELL de la manière habituelle.

La preuve du théorème de séquentialisation met en évidence que la condition (SDAG) sur le graphe de dépendance est d'une nature différente des deux autres conditions de la définition 4.18. En passant par la contractilité, on a pu voir que la première condition (SDAG) assure l'absence de deadlocks causés des arêtes appariées, tandis que les deux autres (connexité et caractéristique égale à 1) forcent la forme normale à être réduite à un point.

En particulier, l'ordre induit par le graphe de dépendance d'une structure correcte donne une information sur les liens de causalité entre connecteurs ⅋. Le théorème 4.11 montre que toute séquentialisation d'un réseau correct devra *a minima* respecter cet ordre.

Perspectives Une des directions futures de notre travail serait d'explorer plus en détails les rapports entre graphe de dépendance et causalité dans les preuves. Il serait par exemple intéressant de prouver une "réciproque" du théorème 4.11 : si toutes les séquentialisations d'un réseau voient un lien ⅋ introduit avant un autre, cela devrait être visible sur le graphe de dépendance.

Les théorèmes 4.24 et 4.25 sont encourageants : ils montrent que derrière le graphe de dépendance se cache un invariant des séquentialisations d'un réseau, l'ordre induit par le graphe de dépendance.

Références

[1] Jean-Marc Andreoli : *Proposition pour une synthèse des paradigmes de la programmation logique et de la programmation par objets.* Thèse de doctorat, Université Paris VI, juin 1990.

[2] Gérard Berry : Stable models of typed lambda-calculi. pages 72–89, 1978.

[3] Pierre-Louis Curien : Introduction to linear logic and ludics, part ii, 2006.

[4] Pierre-Louis Curien et Guillaume Munch-Maccagnoni : The duality of computation under focus. *In IFIP TCS*, volume 323, pages 165–181. Springer, 2010.

[5] Vincent Danos : *Une application de la logique linéaire à l'étude des processus de normalisation (principalement du λ-calcul).* Ph.D. Thesis, Université Denis Diderot, Paris 7, 1990.

[6] Vincent Danos, Jean-Baptiste Joinet et Harold Schellinx : A new deconstructive logic : Linear logic. 62(3):755–807, 1997.

[7] Vincent Danos et Laurent Regnier : The structure of multiplicatives. 28:181–203, 1989.

[8] Paulin Jacobé de Naurois et Virgile Mogbil : Correctness of linear logic proof structures is nl-complete. *Theor. Comput. Sci.*, 412(20):1941–1957, 2011.

[9] Amina Doumane : *Géométrie de l'Interation VI.* Mémoire de master 2, Université Paris Diderot, septembre 2013.

[10] Thomas Ehrhard et Laurent Regnier : Differential interaction nets. *Theor. Comput. Sci.*, 364(2):166–195, 2006.

[11] Jean-Yves Girard : *Interprétation fonctionnelle et élimination des coupures de l'arithmétique d'ordre supérieur.* Thèse de doctorat d'etat, Université Paris VII, 1972.

[12] Jean-Yves Girard : The system F of variable types, fifteen years later. *Theoretical Computer Science*, 45:159–192, 1986.

[13] Jean-Yves Girard : Linear logic. *Theoretical Computer Science*, 50(1):1 – 101, 1987.

[14] Jean-Yves Girard : A new constructive logic : classical logic. 1(3):255–296, 1991.

[15] Jean-Yves Girard : Proof-nets : the parallel syntax for proof-theory. *In* Aldo Ursini et Paolo Agliano, éditeurs : *Logic and Algebra*, volume 180 de *Lecture Notes In Pure and Applied Mathematics*, pages 97–124, New York, 1996. Marcel Dekker.

[16] Jean-Yves Girard : Light linear logic. 143(2):175–204, juin 1998.

[17] Jean-Yves Girard : *Le Point Aveugle : Cours de logique. Tome 1, Vers la perfection ; Tome 2, Vers l'imperfection.* Hermann, 2006.

[18] Georges Gonthier, Martin Abadi et Jean-Jacques Lévy : The geometry of optimal lambda reduction. *In Proceedings of the 19th Annual ACM Symposium on Principles of Programming Languages*, pages 15–26, 1992.

[19] Stefano Guerrini : Correctness of multiplicative proof nets is linear. *In LICS*, pages 454–463. IEEE Computer Society, 1999.

[20] Stefano Guerrini : A linear algorithm for mll proof net correctness and sequentialization. *Theor. Comput. Sci.*, 412(20):1958–1978, 2011.

[21] William A. Howard : The formulae-as-type notion of construction, 1969. *In* J. P. Seldin et R. Hindley, éditeurs : *To H. B. Curry : Essays in Combinatory Logic, Lambda Calculus, and Formalism*, pages 479–490. Academic Press, New York, 1980.

[22] Yves Lafont : Interaction nets. pages 95–108, San Francisco, California, 1990. ACM Press.

[23] Yves Lafont : Soft linear logic and polynomial time. *TCS*, 318(1–2):163–180, juin 2004.

[24] Olivier Laurent : *Étude de la polarisation en logique.* Thèse de doctorat, mars 2002.

[25] Dale Miller : Overview of linear logic programming. *In* Thomas Ehrhard, Jean-Yves Girard, Paul Ruet et Phil Scott, éditeurs : *Linear Logic in Computer Science*, volume 316 de *London Mathematical Society Lecture Note*, pages 119–150. Cambridge University Press, 2004.

[26] Misao Nagayama et Mitsuhiro Okada : A new correctness criterion for the proof nets of non-commutative multiplicative linear logics. *J. Symb. Log.*, 66(4):1524–1542, 2001.

[27] John C. Reynolds : Towards a theory of type structure. *In Symposium on Programming*, volume 19 de *Lecture Notes in Computer Science*, pages 408–423. Springer, 1974.

[28] Noam Zeilberger : Focusing and higher-order abstract syntax. *In POPL*, pages 359–369. ACM, 2008.

A. Preuve du théorème 4.11

Théorème A.1. *Soit π une preuve MLL et R le réseau de preuve correspondant. On a :*

$$D(R) \subseteq O(\pi)$$

Démonstration. Par induction sur π.

- Supposons que la preuve π de $\vdash \Gamma, A \parr B$ a été obtenue a partir de la preuve ν de $\vdash \Gamma, A, B$. On notera R_π et R_ν les réseaux de preuves respectifs de π et ν.
  Comme $A \parr B$ est en dessous de toutes les formules introduites par une règle $\parr$ dans π, et comme l'ordre sur toutes les autres formules est le même dans π et dans ν, le graphe $O(\pi)$ est constitué du graphe $O(\pi)$, du nœud $A \parr B$ et d'arêtes partant des nœuds de $O(\pi)$ et arrivant sur $A \parr B$.

  D'autre part, $D(R_\pi)$ est constitué du graphe $D(R_\nu)$, d'un nœud $A \parr B$ et de certaines arêtes partant des nœuds de $D(R_\nu)$ et arrivant sur $A \parr B$. En effet, on ne peut pas créer de nouvelles arêtes entre les nœuds de $D(R_\nu)$, car cela signifierait qu'il existe un chemin élémentaire entre les nœuds $\parr$ correspondants dans π qui n'existait pas dans ν. Un tel chemin passerait forcément par les prémisses gauche et droite consécutivement du nœud $A \parr B$, ce qui est absurde. De plus, il ne peut exister une arête de $A \parr B$ vers l'un des nœuds de ν car ici encore le chemin élémentaire correspondant passerait consécutivement par les deux prémisses du nœuds $A \parr B$ ce qui est absurde. Or, par hypothèse d'induction, on a $D(R_\nu) \subseteq O(\nu)$. Par ce qui précède, on conclut qu'on a aussi $D(R_\pi) \subseteq O(\pi)$.
- Supposons que la preuve π de $\vdash \Gamma, A \otimes B$ a été obtenue a partir de la preuve π_1 de $\vdash \Gamma, A$ et de la preuve π_2 de $\vdash \Delta, B$. On notera R_π, R_{π_1} et R_{π_2} les réseaux de preuves respectifs de π, π_1 et π_2.
  Il est facile de voir que $O(\pi)$ est l'union de $O(\pi_1)$ et $O(\pi_2)$.

  D'autre part, $D(R_\pi)$ est l'union de $D(R_{\pi_1})$ et $D(R_{\pi_2})$. En effet, s'il y a un chemin élémentaire entre deux nœuds $\parr$ appartenant respectivement à R_{π_1} et R_{π_2}, celui-ci contiendrait forcément un cycle puisqu'il doit passer deux fois par le nœud $A \otimes B$, ce qui est absurde.

  Or, par hypothèse d'induction, on a $D(R_{\pi_1}) \subseteq O(\pi_1)$ et $D(R_{\pi_2}) \subseteq O(\pi_2)$. Par ce qui précède, on conclut qu'on a aussi $D(R_\pi) \subseteq O(\pi)$.

□

B. Preuve du théorème 4.14

Théorème B.1. *Soit G un graphe et soient respectivement n, a, c et b ses nombres de nœuds, arêtes, composantes connexes et cycles.*

On a l'égalité suivante :

$$n - a = c - b.$$

On notera χ_G cette valeur caractéristique du graphe.

Démonstration. Le résultat se démontre aisément par induction sur le nombre d'arêtes du graphe. Si le graphe ne contient aucune arête, on n'a aucun cycle et autant de composantes connexes que de nœuds : $a = b$ et $n = c$ et l'égalité est vérifiée. Si l'égalité est vraie pour les graphes comportant k arêtes et soit G un graphe à $k + 1$ arêtes, obtenue en ajoutant une arête a_0 à un graphe G' (qui satisfait l'égalité par induction). L'ajout de cette arête a relié deux composantes connexes distinctes ou bien a créé un cycle suplémentaire : dans tous les cas, les deux membres de l'égalité ont diminué de 1 (c'est-à-dire que $\chi_G = \chi G' - 1$) d'où le résultat. □