Sur les systèmes de formes normales pour représenter efficacement des fonctions multivaluées

Pierre Mercuriali

▶ **To cite this version:**

Pierre Mercuriali. Sur les systèmes de formes normales pour représenter efficacement des fonctions multivaluées. Informatique [cs]. Université de Lorraine, 2020. Français. NNT : 2020LORR0241 . tel-03202757

HAL Id: tel-03202757

https://hal.univ-lorraine.fr/tel-03202757v1

Submitted on 20 Apr 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

AVERTISSEMENT

Ce document est le fruit d'un long travail approuvé par le jury de soutenance et mis à disposition de l'ensemble de la communauté universitaire élargie.

Il est soumis à la propriété intellectuelle de l'auteur. Ceci implique une obligation de citation et de référencement lors de l'utilisation de ce document.

D'autre part, toute contrefaçon, plagiat, reproduction illicite encourt une poursuite pénale.

Contact : ddoc-theses-contact@univ-lorraine.fr

LIENS

Code de la Propriété Intellectuelle. articles L 122. 4
Code de la Propriété Intellectuelle. articles L 335.2- L 335.10
http://www.cfcopies.com/V2/leg/leg_droi.php
http://www.culture.gouv.fr/culture/infos-pratiques/droits/protection.htm

UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

Sur les systèmes de formes normales pour représenter efficacement des fonctions multivaluées

THÈSE

présentée et soutenue publiquement le 15 décembre 2020

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention informatique)

par

Pierre Mercuriali

Composition du jury

Rapporteurs :	Mirna Džamonja Damiano Mazza
Examinateurs :	Jean-Luc Marichal Michael Rusinowitch
Directeurs :	Miguel Couceiro Romain Péchoux

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thesul.

Remerciements

Mes directeurs de thèse, Miguel et Romain ; les membres des équipes Orpailleur et Mocqua ainsi que tous les chercheurs dont j'ai croisé la route et avec lesquels j'ai eu la chance de travailler et d'apprendre ; mes parents et mes frères ; mes amis ; mes cobureaux ; Mimi ; Carmen, Hercule, Léonardo.

Table des matières

1

Introduction

> Πάντες ἄνθρωποι τοῦ εἰδέναι ὀρέγονται φύσει.
> L'homme a naturellement la passion de connaître.
>
> *Métaphysique*, Aristote ([Ari79]).

Le partage d'information de manière efficace est une problématique commune : nous cherchons à communiquer de manière de plus en plus rapide, de manière de plus en plus succincte, tout en conservant, le plus exactement possible, la teneur du propos. Une autre problématique consiste à résumer une certaine information, avec une perte possible de contenu, perte gouvernée par des règles différentes selon le but à atteindre et la nature de l'information à résumer. C'est le cas, par exemple, du vote par majorité : l'opinion d'un certain nombre de personnes, qui choisissent entre un certain nombre de candidats, est résumée en un seul de ces candidats. Nous étudions ici un cas particulier de la rencontre entre ces deux buts en apparence opposés : nous représentons et manipulons des formules logiques en conservant *exactement* leur contenu sémantique, mais les construisons à partir d'une fonction d'agrégation, la médiane, qui, dans le cas Booléen, correspond au vote par majorité de trois personnes, ou plus si l'on considère la médiane généralisée (toujours à un nombre impair d'entrées).

Langage et représentations

Dans le but d'éclairer le sujet, nous pourrions proposer de partir d'un raisonnement par analogie. Par exemple, dans la figure 1.1, nous avons représenté de manière abstraite un contenu sémantique α. Très généralement, α peut être un recueil de recettes de cuisine, une classe de fonctions, un livre de géométrie, etc. Ce contenu α est représenté dans deux langues différentes, A et B. Le livre écrit dans la langue A est plus mince que le livre écrit dans la langue B. Dans une certaine mesure, nous pouvons dire que, *en ce qui concerne la représentation du contenu sémantique α, la langue A est plus efficace que la langue B.*

Nous pouvons observer cet effet dans la langue naturelle. Les titres de la table 1.1 sont tirés du même article [Ish20], traduit en français, portugais, anglais, russe, polonais et japonais, ainsi qu'en une langue artificielle fictive, ad-hoc, dont le seul symbole "$\star$" correspond exactement au sens voulu.

Ici, le japonais, avec seulement 11 caractères, permet d'exprimer le titre de la manière la plus concise, selon la mesure qui consiste à compter le nombre de caractères du titre. La langue

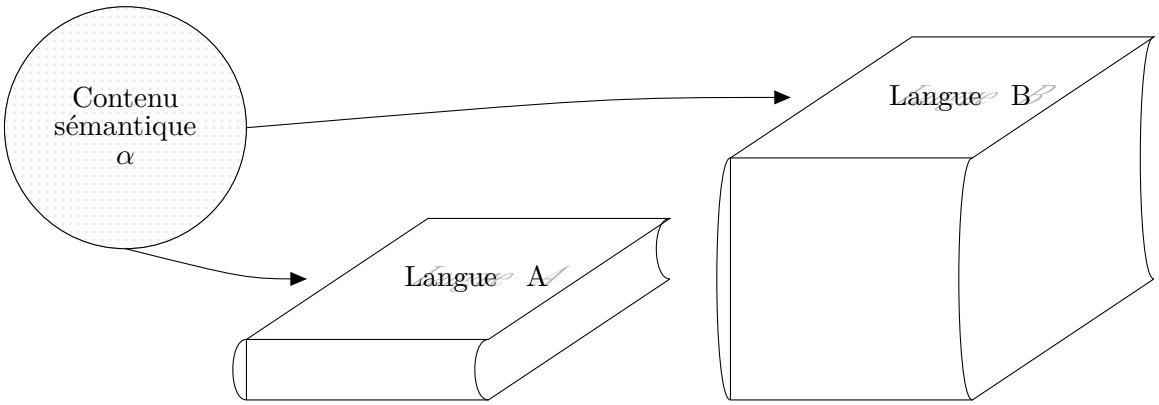


FIGURE 1.1 – Différentes représentations pour un même contenu sémantique.

Titre	Taille	Langue
La taille des textes dans les traductions	41	français
O tamanho dos textos nas traduções	34	portugais
Dimensiunea textului in traduceri	33	polonais
Размер текста в переводе	24	russe
Text size in translation	24	anglais
訳文における文字サイズ	11	japonais
$\star$	1	conlangue fictive

TABLE 1.1 – Comparaisons d'une phrase traduite dans différentes langues.

construite fictive, créée dans le but précis d'exprimer le plus succinctement possible le titre, ne nécessite qu'un seul symbole.

En quantifiant de manière universelle tous les contenus sémantiques possibles, c'est-à-dire en les considérant tous, nous pouvons comparer les langues elles-mêmes, en mesurant les représentations minimales de ces contenus, c'est-à-dire que nous nous posons la question suivante.

> Étant donnés deux langages A et B, est-ce que, pour tout contenu sémantique, la représentation du contenu dans le langage A est plus efficace, pour une certaine définition de l'efficacité, que la représentation du contenu dans le langage B ?

Nous étudions cette question dans le cas particulier des fonctions Booléennes et des fonctions multi-valuées à valeurs dans un treillis, chapitres 5 et 6. Malgré la présentation très générale que nous donnons dans cette introduction, il convient de définir précisément les notions utilisées, ce que nous faisons par la suite, afin d'éviter autant que faire se peut les paradoxes liés à la description et la représentation d'objets mathématiques. Par exemple, le paradoxe de Berry, attribué à un bibliothécaire londonien par Russell ([Rus06, Rus08]), s'énonce ainsi en français :

> « Le plus petit entier naturel non descriptible par une expression de quinze mots ou moins ».

Ce paraxode est similaire au paradoxe d'Épiménide de Cnossos, poète crétois que Paul de Tarse mentionne dans l'Épître à Tite :

> « L'un d'entre eux, leur propre prophète, a dit :
> "Crétois, perpétuels menteurs,

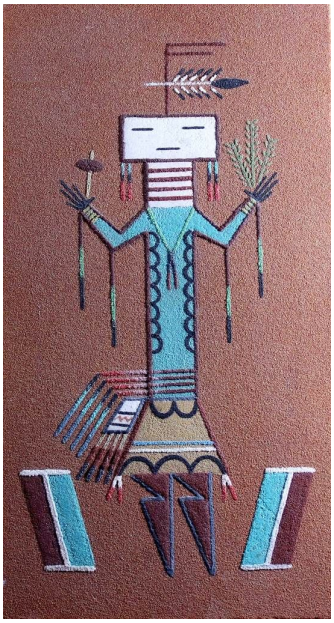


FIGURE 1.2 – *Iikááh* peint par Mitchell Silas.

bêtes méchantes, panses fainéantes." » ([LB04]).

La version la plus simple de ce paradoxe dit « du menteur » met en scène un homme qui dit « Je mens ! » : s'il dit la vérité, alors il ment (comme il le dit), mais s'il ment, alors il ne ment pas (comme il le dit) et dit donc la vérité. Le paradoxe provient là de l'auto-référence, la confusion entre le discours, c'est-à-dire le sens porté par la phrase, et le discours sur le discours, qui exprime quelque chose sur la phrase elle-même (le fait qu'elle soit un mensonge).

Dans la langue naturelle, le phénomène d'existence d'un sens qui admet un représentation efficace dans une certaine langue peut être en partie le résultat de l'utilisation historique et culturelle qui en est faite. Si l'on veut en conserver le sens exact, certains termes ne peuvent être traduits, d'une langue à une autre, que par une paraphrase. C'est le cas de termes qui correspondent à une pratique culturelle ou artistique bien précise. Par exemple, les amérindiens Navajo du Sud-Ouest des États-Unis possèdent une pratique rituelle sacrée qui consiste à dessiner, sur le sol, des images codifiées, à l'aide de sables colorés naturellement ou à l'aide de poudres de charbon, de gypse blanc ou bleu, de grès, etc ; voir, par exemple, la figure 1.2. Cette description relativement longue en français est rendue directement par le terme Navajo *iikááh*, beaucoup plus succinct ([HGC79, Wym83]).

Bien sûr, cette analogie n'est pas complète ; la reconnaissance d'un terme dépend entre autres du contexte de communication et de la culture des communicants. C'est le cas, par exemple, du jargon scientifique, ou technique, ou artistique, etc. qui fait sens dans un contexte particulier et entre des communicants qui connaissent le jargon, et qui est hermétique pour toute personne extérieure.

> Peut-on classifier tous les langages ainsi ?

Nous pouvons nous trouver dans différentes situations qui rendent la comparaison difficile : par exemple, le langage A peut être plus efficace que dans le langage B pour un certain contenu, mais, pour un autre contenu, c'est alors le langage B qui sera plus efficace. Même si les langues

x	y	$\min(x, y)$
0	0	0
0	1	0
0	2	0
1	0	0
1	1	1
1	2	1
2	0	0
2	1	1
2	2	2

TABLE 1.2 – Table de vérité du minimum binaire sur $\{0, 1, 2\}$.

d'un ensemble de langues ne peuvent pas nécessairement toutes être comparées, nous pouvons tout de même poser la question suivante.

> Existe-t-il alors des langues qui sont plus efficaces que d'autres ?

La notion d'efficacité est également sujette à discussion. Dans notre comparaison de langues, nous avons choisi la taille comme mesure d'efficacité : une représentation est plus efficace qu'une autre si elle est plus petite. Bien d'autres notions d'efficacité peuvent être considérées :

— la rapidité de lecture,
— la possibilité de parallélisation pour un circuit ou un programme,
— le support de transmission, etc.

Dans cette thèse, nous privilégions la mesure qui consiste à compter le nombre de symboles « importants » de nos représentations, c'est-à-dire, dans le cas Booléen par exemple, les connecteurs logiques, comme nous compterions le nombre de composants électroniques dans un circuit, au lieu de compter, par exemple, le nombre de vis qui servent à maintenir le circuit ou la quantité de fil de soudure à utiliser pour connecter les composants.

Opérations

Dans cette thèse, nous nous intéressons plus particulièrement à la représentation d'opérations ; étant donné un ensemble A contenant au moins deux éléments, une *opération* f est une fonction

$$f\colon \underbrace{A \times A \times \cdots \times A}_{n \text{ fois}} \to A,$$

pour un certain n, qui est l'arité de l'opération f. Par la suite, nous considérerons des ensembles A de cardinalité finie. Le plus souvent, nous examinons les représentations de toutes les opérations sur A.

Une manière de représenter une opération est de dresser sa table de valeurs, ou *table de vérité*. Nous en donnons un exemple dans la table 1.2. Dresser une table de vérité nécessite de calculer les valeurs que prend l'opération sur tous les tuples de A^n, c'est-à-dire, en notant $|A|$ la cardinalité de A, d'écrire une table de taille $|A|^n$, ce qui est généralement coûteux en temps et en espace.

Nom	Connecteurs utilisés	Exemple
Forme normale disjonctive (DNF)	$\wedge, \vee$	$(x_1 \wedge x_2) \vee (\overline{x_1} \wedge x_3 \wedge 1)$
Forme normale conjonctive (CNF)	$\wedge, \vee$	$1 \wedge (x_1 \vee x_2)$
Forme normale polynomiale (PNF)	$\wedge, \oplus$	$1 \oplus (x_1 \wedge \overline{x_2})$
Forme normale polynomiale duale (PNF^{d})	$\vee, \oplus$	$x_1 \oplus (x_1 \vee x_2) \oplus x_2$
Forme normale médiane (MNF)	$\mathtt{m}$	$\mathtt{m}(\mathtt{m}(x_1, 0, 1), x_2, \overline{x_3})$
⋮	⋮	⋮

TABLE 1.3 – Formes normales.

Formes normales

Avec la formalisation de la logique propositionnelle ([Boo47, CL61]) apparaît une claire distinction entre la *syntaxe* et la *sémantique* : ici, cette dernière correspond à la table de vérité. La syntaxe consiste en des formules qui, une fois interprétées, correspondent à l'opération sémantique correspondante. Ici, la table 1.2 est résumée en la formule, beaucoup plus compacte,

$$\min(x, y),$$

grâce au symbole min.

À un même contenu sémantique correspond potentiellement une infinité de représentations. L'expression $\min(x, y)$ pour le minimum est équivalente à, c'est-à-dire a la même interprétation sémantique que la représentation de taille arbitraire

$$\min(\ldots \min(\min(x, y), y) \ldots, y).$$

> Pour un langage et une fonction donnés, c'est-à-dire un ensemble de formules construites à partir d'un ensemble de symboles, existe-t-il des formules de tailles minimales qui représentent cette fonction ?

C'est le cas pour les formes normales utilisées habituellement pour la représentation des fonctions Booléennes telles que la forme normale disjonctive ou la forme normale conjonctive. Dans le chapitre 5, nous discutons de la définition d'une forme normale appropriée pour les formules à base du connecteur médian, liée en particulier à un ordre structurel qui nous permet d'accéder à cette notion de « formule de taille minimale », dans la section 5.2.

Calcul à base de médiane

Dans le cadre des fonctions Booléennes, c'est-à-dire lorsque $A = \{0, 1\}$, les *formes normales* sont des formules codifiées qui permettent de représenter toute fonction. La *forme normale disjonctive* est une disjonction de conjonctions de littéraux (variables x_i ou variables niées $\overline{x_i}$) et de constantes (0 ou 1) ; son dual, la *forme normale conjonctive*, est une conjonction de disjonctions de littéraux et de constantes ; la *forme normale polynomiale*, ou *somme de produits*, ou *forme normale de Reed-Muller* ([Ree53, Mul54]), ou *forme normale de Zhegalkine* ([Gé27]), est une somme modulo 2 (`XOR` logique) de conjonctions de littéraux et de constantes, ou de disjonctions dans le cas dual de cette forme. La table 1.3 récapitule ces formes normales.

Ces quatre formes normales, dites canoniques, sont utilisées dans le domaine du choix social, de l'électronique, de l'intelligence artificielle, etc. Dans l'électronique en particulier, l'utilisation des connecteurs logiques simples tels que le `OU` et le `ET` correspond à la facilité avec laquelle on peut les simuler par des circuits simples. Dans [CFL06], après avoir étudié de manière exhaustive la composition de clones Booléens, les auteurs ont introduit et étudié la *forme normale médiane*, qui repose sur l'utilisation d'un seul connecteur ternaire `m`, défini par l'équivalence

$$\mathtt{m}(x_1, x_2, x_3) \equiv (x_1 \wedge x_2) \vee (x_2 \wedge x_3) \vee (x_3 \wedge x_1).$$

Celui-ci peut-être vu comme donnant le résultat d'un vote majoritaire entre trois personnes : si deux au moins votent pour le même candidat, alors celui-ci sera élu. Malgré l'utilisation d'un seul connecteur, les formes normales obtenues sont strictement plus efficaces, en un sens que nous définirons précisément (définition 112, page 95), que les DNF, CNF, PNF et PNF$^{\mathrm{d}}$ qui, elles, sont deux-à-deux incomparables ([CFL06]). Intuitivement, cela signifie qu'il n'existe pas de fonction qui a une représentation minimale exponentielle en MNF et des représentations minimales polynomiales dans les autres formes, mais qu'il existe, entre les formes DNF, CNF, PNF et PNF$^{\mathrm{d}}$, des fonctions qui ont des représentations exponentielles dans l'une et polynomiale dans l'autre, et vice-versa.

> Existe-t-il d'autres formes normales qui vérifient cette propriété d'efficacité de la MNF ?

Dans le chapitre 6, nous montrons qu'il en existe effectivement d'autres (une infinité !), que nous caractérisons. Par exemple, toutes les médianes $2n+1$-aires génèrent des formes normales qui sont équivalentes, en termes de complexité, à la MNF (proposition 133).

La réécriture de formules dans le cadre des formes normales usuelles a été beaucoup étudiée tant de manière théorique que pratique, notamment dans le but de simplifier des formules ou de passer d'une forme à une autre. Nous discutons de la réécriture en général dans la section 3.4.2. Par exemple, passer d'une formule en DNF à une formule en CNF peut nécessiter l'utilisation des règles dites de De Morgan ([DM47])

$$\overline{x_1 \wedge x_2} \equiv \overline{x_1} \vee \overline{x_2},$$
$$\overline{x_1 \vee x_2} \equiv \overline{x_1} \wedge \overline{x_2},$$

et des règles de distributivité

$$x_1 \wedge (x_2 \vee x_3) \equiv (x_1 \wedge x_2) \vee (x_1 \wedge x_3),$$
$$x_1 \vee (x_2 \wedge x_3) \equiv (x_1 \vee x_2) \wedge (x_1 \vee x_3).$$

Pour la MNF, il existe également des règles similaires, telles que la règle de majorité

$$\mathtt{m}(x_1, x_1, x_2) \equiv x_1$$

ou la règle de distributivité

$$\mathtt{m}(\mathtt{m}(x_1, x_2, x_4), \mathtt{m}(x_1, x_2, x_5), x_3) \equiv \mathtt{m}(\mathtt{m}(x_1, x_2, x_3), x_4, x_5).$$

> Ces règles sont-elles suffisantes pour réécrire n'importe quelle formule médiane en n'importe quelle autre ? Si oui, est-il difficile de simplifier une formule médiane ?

Dans le chapitre 5, nous montrons que c'est le cas pour un certain système d'équations (système 62, résultat connu depuis Birkhoff ([Bir40]).

La médiane est une fonction d'*agrégation* (voir, par exemple, [GMMP11]) : elle permet de combiner différentes valeurs numériques en une seule, comme on le fait en calculant une moyenne arithmétique, géométrique, etc. Le but est de pouvoir résumer les valeurs numériques, ou d'arriver à un consensus entre différents avis. En aide à la décision, en statistiques, en économie, en finance, etc., ces fonctions peuvent être utilisées pour prendre en compte des préférences individuelles ([Mar09a]). Les valeurs à agréger sont alors des *degrés de satisfaction* ou de *préférence*, c'est-à-dire à quel point une alternative est satisfaisante, comme par exemple une appréciation qualitative de la nourriture d'un restaurant ("terrible $<$ médiocre $<$ bon $<$ délicieux"), ou à quel point une option est préférée à une autre (préférer une salade à un pain de viande). Marichal ([Mar09b]) démontre que les *intégrales de Sugeno* discrètes ([Sug74, Mar00]) sur des treillis distributifs peuvent être caractérisées par un système d'équations basé sur la médiane. Ces intégrales sont étudiées dans le domaine de l'intégration et de l'évaluation multi-critères, lorsque qu'il n'existe pas d'ordre total sur les préférences. C'est le cas lorsque, par exemple, nous rajoutons une appréciation "étrange" sur la qualité de la nourriture d'un restaurant, sans savoir plus que "terrible $<$ étrange" et que "étrange $<$ délicieux". Une importante direction de recherche consiste en l'apprentissage de ces intégrales ([BC17]), ce qui peut être très coûteux en termes de calcul, à partir de données réelles.

Plan de thèse et contributions

— Nous commençons dans le chapitre 3 par présenter le bagage théorique dont nous avons besoin pour examiner ces questions. Les ensembles ordonnés et les treillis nous permettent de comparer les objets que nous manipulons, tels que les polynômes latticiels et les fonctions Booléennes. La théorie des clones Booléens, classes de fonctions qui contiennent les projections et qui sont fermées par composition, oriente nos recherches, entre autres, de candidats au titre de connecteur le plus efficace. En particulier, grâce à la description complète des clones ([Pos41]) et de leurs compositions ([CFL06]), nous pouvons étudier toutes les formes normales possibles, qui correspondent à des compositions de clones, et, ainsi, exhiber des connecteurs intéressants, dont les interprétations sont des générateurs de clones. Par exemple, la médiane correspond au générateur d'arité minimale du clone SM des fonctions auto-duales et monotones (voir définition 101).
Certains modèles de calcul, tels que les machines de Turing, les termes et les circuits, et, enfin, quelques notions de complexité, nous permettent de déterminer la difficultés de résolution de certaines de ces questions. En particulier, les machines de Turing nous permettent de formaliser la notion de coût en temps ou en espace d'un algorithme ; nous illustrons par ailleurs une machine de Turing qui nous permet de résoudre un de nos problèmes de simplification de formules (figure 5.3). Les termes et les systèmes de réécriture nous permettent de formaliser la notion de formules et de représentation de fonctions, que nous manipulons tout au long de cette thèse.

— Dans le chapitre 4, nous présentons plus précisément la médiane, en particulier du point de vue historique, géométrique, comme un point dont la somme des distances à une famille forme une médiane pour cette famille, et statistique, comme une caractéristique d'une série.

Nous mentionnons également l'étude algébrique, combinatoire de la médiane sur les ensembles ordonnés, les arbres, etc. , née au vingtième siècle.
Nous mentionnons finalement, dans la section 4.4, certains travaux récents sur l'utilisation de la médiane dans les circuits et les bénéfices qu'elle apporte notamment en terme de minimisation.

— Dans le chapitre 5, après avoir défini les termes médians de manière précise, ainsi que les polynômes latticiels, objets que nous allons représenter par des termes médians, nous rappelons une spécification équationnelle des formules médianes qui contient, en particulier, les deux règles de majorité et de distributivité donnée ci-haut (système 62). Cette spécification a l'avantage d'être à la fois correcte et complète (théorème 64), ce qui a pour conséquence que toute réécriture de formule conserve l'équivalence logique, et que toute formule peut être réécrite en toute autre qui lui est équivalente. Ce dernier point est particulièrement intéressant : en effet, cela signifie que cette spécification seule nous permet de simplifier une formule autant qu'il est possible.
Nous abordons ensuite la question de la définition d'une forme normale médiane de manière précise dans la section 5.2. Nous proposons une description structurelle des formules (définition 65) qui donne lieu à un bon ordre sur les formules et nous permet ainsi de proposer une définition des formes normales médianes (définition 69). Cette représentation consiste à compter le nombre de médianes qui apparaissent à chaque profondeur d'une formule.
Puis, nous étudions la complexité du problème qui consiste à simplifier des formules médianes, à la fois dans le cadre des polynômes latticiels sur des treillis distributifs que dans le cadre de la représentation de fonctions Booléennes. Ainsi, le problème Monotone Smallmed (définition 91), sur les polynômes latticiels, consiste à savoir, étant donné une formule médiane et une représentation structurelle, s'il existe une autre formule médiane équivalente à celle en entrée et de représentation structurelle strictement plus petite. Nous démontrons que ce problème est modérément intractable, c'est-à-dire qu'il est au moins dans NP ; en particulier nous majorons sa difficulté par Σ_2^{P} (théorème 73). Nous étudions également le problème similaire Monotone Smallmed$_S$, qui consiste à fixer la représentation structurelle auparavant ; plus simple, ce problème est dans la classe coNP (théorème 76). Dans le but de rendre ce problème plus facile à résoudre, c'est-à-dire avec des réécritures de tailles polynomiales (lemme 78), nous étudions le système 77) dans lequel les règles sont orientées, ce qui se fait au prix de la complétude (proposition 81). Nous discutons finalement la possibilité de rajouter des règles à un système de réécriture pour en conserver la complétude.
Nous étendons naturellement nous résultats au cas non-monotone, c'est-à-dire dans le cadre de la représentation des fonctions Booléennes, en ajoutant la négation à nos formules médianes et en modifiant le système en conséquence (système 85). De manière étonnante, permettre la négation dans les formules médianes permet d'obtenir des représentations de fonctions monotones strictement plus petites que sans la négation (proposition 89).

— Dans le chapitre 6, nous étudions plus précisément la représentation de fonctions Booléennes, cette fois en autorisant des connecteurs autres que la médiane, et en étendant la notion de forme normale donnée dans [CFL06]. La théorie des clones Booléens nous permet d'orienter et de guider cette étude, notamment en produisant des connecteurs et des formes normales explicites grâce à la classification de Post ([Pos41], figure 3.6).
Nous commençons par proposer une définition des systèmes de formes normales, comme un ensemble de termes stratifiés qui permettent de représenter toutes les fonctions Booléennes (définition 100). Nous en démontrons quelques propriétés, notamment sur la na-

ture des connecteurs qui les composent, et en tirons une classification des connecteurs Booléens en deux classes : les connecteurs quasi-Sheffer, qui à eux seuls, avec les constantes et des litéraux, permettent de représenter toutes les fonctions Booléennes, tels que la médiane ou la fonction Sheffer (`NAND` logique) et les autres, tels que la conjonction ou la disjonction (proposition 105). Nous démontrons également que le nombre de connecteurs nécessaires dans un NFS est inférieur à 2 (proposition 108), et déterminons tous les NFSs, similaires à la forme normale médiane, qui ne nécessitent qu'un seul connecteur (ou un connecteur et la négation).

Par la suite, nous nous concentrons sur l'efficacité des formes normales (définition 112) pour représenter des fonctions. Nous formalisons la manière de transformer des termes d'un NFS à un autre, dans le but de montrer, par exemple, qu'il est possible de transformer un terme de manière efficace, c'est-à-dire sans explosion exponentielle de sa taille, comme il est parfois le cas lorsque l'on passe d'une DNF à une CNF. Nous démontrons grâce à cette formalisation que rajouter des connecteurs n'améliore pas de manière significative l'efficacité des représentations, que l'arité des connecteurs n'importe pas non plus, et que la forme normale médiane, pour un certain sous-ensemble de formes normales, reste la manière la plus efficace de représenter les fonctions correspondantes (théorèmes 135 et 137).

— Dans le dernier chapitre 7, nous discutons d'ouvertures de recherche future, notamment dans le domaine des circuits, ce qui nécessite des mesures de complexité différentes, et sur le plan des formes normales en logique multi-valuée.

La plupart des résultats du chapitre 5 ont été publiés dans [CMPS17, CMPS19]. Ceux du chapitre 6 ont été publiés dans [CLMP20]. Les résultats obtenus ont également fait l'objet de publications et de communications à LFA (Rencontres Francophones sur la Logique Floue et ses Applications) [CMP17] et à DICE (Developments in Implicit Computational Complexity) [CLM+18].

2

Introduction (English version)

> Πάντες ἄνθρωποι τοῦ εἰδέναι ὀρέγονται φύσει.
> Man naturally has the passion for knowing.
>
> *Metaphysics*, Aristotle ([Ari79]).

Efficient information sharing is an important human problem in this age : we want to communicate faster and faster, in a more and more succinct way, while keeping, as much as possible, the accuracy of the discourse. Another problem consists in summarizing information, though with a potential loss of content, which is controlled by different rules depending on the goal and the type of information to summarize. For instance, majority voting summarizes the opinion of a certain number of people about a set of candidates into a single candidate. We study in this thesis the reunion of these two goals which may seem opposed at first glance : we represent and manipulate logical formulas while keeping their *exact* semantic content intact, but we build them using an aggregation function, the median, which, in the Boolean case, corresponds to the majority vote of three persons, or more when we consider the generalized median – while keeping an odd number of inputs.

Language and representations

So as to explain the subject better, we propose to start by reasoning with the help of an analogy. In the figure 2.1, we have represented in an abstract manner a certain *semantic content*, α. Very generally, α can be a cookbook, a class of functions, a geometry manual, etc. This content α is represented in two different languages, A and B. The book written in the language A is thinner than the one written in the language B. In a way, we can reasonably say that *with regards to the representation of the semantic content α, the language A is more efficient than the language B.* This effect can be witnessed in natural language. The titles from table 2.1 were extracted from the same article [Ish20], translated in French, Portuguese, English, Russian, Polish and Japanese, as well as in a fictional artificial ad-hoc language, the only symbol of which, " $\star$ ", corresponds exactly to the meaning we want. In this case, Japanese, with only 11 characters, expresses this meaning in the most succinct manner, according to the measure that consists in counting the number of characters in the title. The fictional conlang, built with the express purpose of expressing the title, only requires a single symbol.

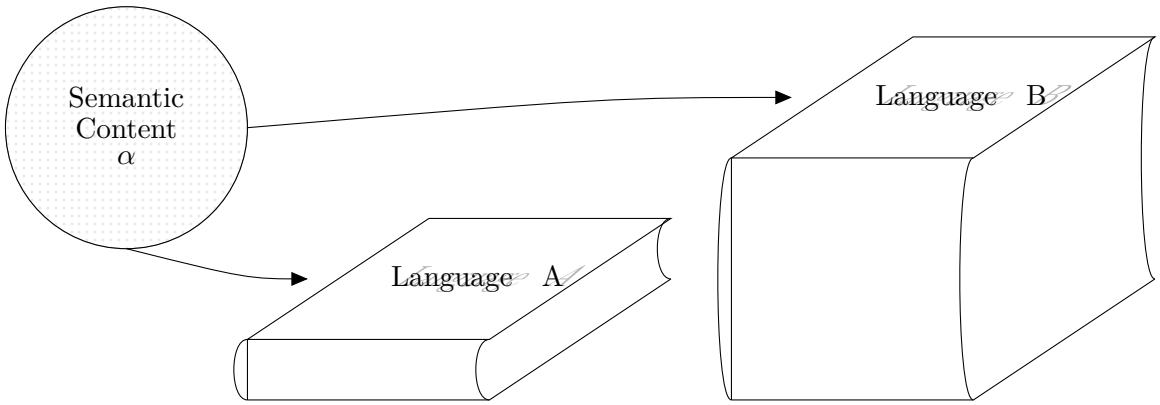


FIGURE 2.1 – Different representations for the same semantic content.

Title	Size	Language
La taille des textes dans les traductions	41	French
O tamanho dos textos nas traduções	34	Portuguese
Dimensiunea textului in traduceri	33	Polish
Размер текста в переводе	24	Russian
Text size in translation	24	English
訳文における文字サイズ	11	Japanese
$\star$	1	artificial conlang

TABLE 2.1 – Comparing the same sentence translated in different languages.

If we universally quantify all possible semantic contents, by which we mean we consider all of them, we can compare languages themselves, by measuring the minimal representations of these contents. In other words, we ask ourselves the following question.

> Given two languages A and B and a certain definition of efficiency, and for every possible semantic content, is the representation of this content in the language A more efficient than the representation of this content in the language B ?

We shall answer this question in the particular cases of Boolean functions and multi-valued lattice functions in chapters 5 and 6. Despite the very general presentation that we give in this introduction, we need to precisely define the notions that we use, in order to avoid as much as possible paradoxes related to the description and the representation of mathematical objects. For instance, Berry's paradox, that was attributed to a London librarian by Russell in [Rus06, Rus08], can be given as follows : consider the expression

"The smallest positive integer not definable in under sixty letters".

This paradox is similar to Epimenide of Cnossos' paradox, Cretan poet Paul of Tarse mentions in the Epistle to Titus :

> "A certain one of them, a prophet of their own, said – 'Cretans ! always liars, evil beasts, lazy bellies !'" ([You98]).

The simple version of this so-called "liar's paradox" involves a man who says : "I'm lying !" : if he tells the truth, then he lies (as he says), but if he lies, then he doesn't (as he says) and thus tells the truth. The paradox here comes from self-reference, the confusion between the discourse,

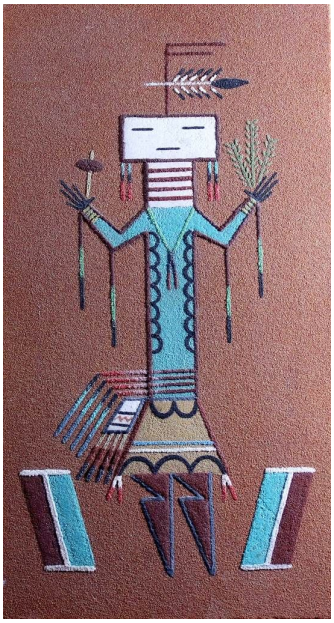


FIGURE 2.2 – *Iikáàh* drawn by Mitchell Silas.

that is, the meaning carried out by the sentence, and a meta-discourse on this discourse, which concerns the sentence itself : the fact that it is a lie.

In natural language, the existence of a meaning that has an efficient representation in a certain language can be, in part, the result of its historical and cultural use. To conserve this exact meaning, some terms can only be translated, from one language to another, by a paraphrase. This is the case for terms that correspond to a precise cultural or artistic practice. For instance, Navajo Indians from the South-West of the United States have a sacred ritual that consists in drawing, on the ground, codified images, using sand that has been previously colored using coal powder, blue or white gypsum, sandstone, etc., or that are otherwise naturally colored : see, for instance, the figure 2.2. This description, relatively long in English and far from capturing the complexity of the practice, is directly expressed by the much more succinct Navajo word *iikáàh* ([HGC79, Wym83]).

Of course, this analogy is not complete. Recognizing a term depends, among other things, on the context of the communication and the cultures of the communicators. Such is the case of scientific, technical, artistic, etc. jargon, that has a certain meaning within a particular context and between communicators who know this jargon, and is hermetic to anyone from the outside.

> Can we classify all languages thusly ?

Different situations can make the comparison difficult. For instance, the language A can be more efficient than the language B for a certain content, but, for another one, the language B can be more efficient. However, even if the languages in a set of languages cannot be all compared, we can still ask the following question.

> Are there languages that are more efficient than others ?

The notion of *efficiency* can be discussed. When we compared natural languages, we chose the size as a measure of efficiency : a representation is more efficient than another if it is smaller. Many other notions of efficiency can be considered :

x	y	$\min(x, y)$
0	0	0
0	1	0
0	2	0
1	0	0
1	1	1
1	2	1
2	0	0
2	1	1
2	2	2

TABLE 2.2 – Truth table of the binary minimum over $\{0, 1, 2\}$.

— reading speed,
— the degree to which a circuit or a program can be parallelized,
— the support of transmission, etc.

We will focus on the measure that consists in counting the number of *important* symbols in our representations, that is, in the Boolean case for instance, logical connectives, just as we would count the number of electronic components in a circuits instead of counting the number of screws necessary to fix the circuit to a board or the quantity of welding wire to use to connect all the components together.

Operations

In this thesis, we focus more particularly on the representation of operations. Given a set A with at least two elements, an *operation* f is a function

$$f\colon \underbrace{A \times A \times \cdots \times A}_{n \text{ times}} \to A,$$

for a certain n called the *arity* of the operation f. In what follows we will only consider the set A to be finite. Very often, we consider the representations of all operations on A.

One way of representing an operation is to build its *truth table*. We give an example in the table 2.2. Building a truth table requires calculating all the values of the operation over every tuple of A^n. With $|A|$ the cardinality of A, this means we need to write down a table of size $|A|^n$, which is generally costly in both time and space.

Normal forms

With the formalization of propositional logic ([Boo47, CL61]), a clear distinction between *syntax* and *semantics* appears. Here, the latter corresponds to the truth table. Syntax corresponds to formulas which, once they are interpreted, correspond to the semantic operations they describe. Here, Table 2.2 is expressed formula

$$\min(x, y),$$

which is much more compact thanks to the symbol min.

Name	Connectives used	Example
Disjunctive normal form (DNF)	$\wedge, \vee$	$(x_1 \wedge x_2) \vee (\overline{x_1} \wedge x_3 \wedge 1)$
Conjunctive normal form (CNF)	$\wedge, \vee$	$1 \wedge (x_1 \vee x_2)$
Polynomial normal form (PNF)	$\wedge, \oplus$	$1 \oplus (x_1 \wedge \overline{x_2})$
Polynomial dual normal form (PNF^{d})	$\vee, \oplus$	$x_1 \oplus (x_1 \vee x_2) \oplus x_2$
Median normal form (MNF)	$\mathtt{m}$	$\mathtt{m}(\mathtt{m}(x_1, 0, 1), x_2, \overline{x_3})$
$\vdots$	$\vdots$	$\vdots$

TABLE 2.3 – Formes normales.

To the same semantic content potentially correspond an infinite number of representations. The expression $\min(x, y$ for the minimum is equivalent to, i.e., has the same semantic interpretation as, the representation of arbitrary size

$$\min(\ldots \min(\min(x, y), y) \ldots, y).$$

> For a given language, i.e., a set of formulas built using a set of symbols, and a given formula, are there formulas for minimal size in this language that represent this function ?

This is the case for normal forms, that usually represent Boolean functions such as the disjunctive or the conjunctive normal forms. In Chapter 5, we discuss the definition of an appropriate normal form to describe formulas built using the median connective, in particular through the definition of a structural order which allows us to reach a notion of *formula of minimal size*, in Section 5.2.

Median-based calculus

In the context of Boolean functions, i.e., when $A = \{0, 1\}$, *normal forms* are codified sets of formulas that are able to represent any function. The *disjunctive normal form* is a disjunction of conjunctions of litterals, that is, variables x_i and negated variables $\overline{x_i}$, and constants, that is, 0 and 1. Its dual, the *conjunctive normal form*, is a conjunction of disjunctions of litterals and constants. The *polynomial normal form*, *sum-of-products*, *Reed-Muller* ([Ree53, Mul54]), or *Zhegalkine normal form* ([Gé27]), is a sum modulo 2 (logical `XOR`) of conjunctions of literals and constants, or disjunctions in the case of its dual.

Table 2.3 summarizes these normal forms.

These four canonical normal forms are commonly used in the domains of social choice, electronics, artificial intelligence, etc. In electronics in particular, the use of simple logical connectives such as `OR` and `AND` is partly due to the simplicity with which they can be simulated with simple circuits. In [CFL06], following an exhaustive study of composition of Boolean clones, the authors defined and introduced the *median normal form*, which uses a singular ternary connective $\mathtt{m}$, defined by the following equivalence :

$$\mathtt{m}(x_1, x_2, x_3) \equiv (x_1 \wedge x_2) \vee (x_2 \wedge x_3) \vee (x_3 \wedge x_1).$$

This connective can be seen as the one that gives the results of a majority vote between three people. If at least two vote for the same candidate, then this candidate will be elected. Despite the use of a single connective, the normal forms obtained are strictly more efficient, in a sense that we define precisely in Definition 112, page 95, than the DNF, CNF, PNF and PNF^{d} that

are, however, pairwise uncomparable ([CFL06]). Intuitively, this means that there is no function with a minimal representation in MNF that is exponential and there are no minimal representations in other forms that are polynomial ; furthermore, there exist functions for which there are polynomial representations in one of the DNF, CNF, PNF et PNF^{d} and exponential in the others.

> Are there other normal forms that share the MNF's efficiency ?

In Chapter 6, we show that there indeed exist other normal forms (an infinity !) that we characterize. For instance, all the $2n+1$-ary medians generate normal forms that are equivalent, in terms of complexity, to the MNF (Proposition 133).

Rewriting formulas in the context of the usual normal forms has been extensively explored from both theoretical and practical standpoints, with such goals as the rewriting of formulas or their simplification. We discuss rewriting in general in Section 3.4.2. For instance, converting a DNF formula into a CNF formula may require the use of De Morgan rules ([DM47])

$$\overline{x_1 \wedge x_2} \equiv \overline{x_1} \vee \overline{x_2},$$
$$\overline{x_1 \vee x_2} \equiv \overline{x_1} \wedge \overline{x_2},$$

and distributivity rules

$$x_1 \wedge (x_2 \vee x_3) \equiv (x_1 \wedge x_2) \vee (x_1 \wedge x_3),$$
$$x_1 \vee (x_2 \wedge x_3) \equiv (x_1 \vee x_2) \wedge (x_1 \vee x_3).$$

Similar rules exist for the MNF, such as the majority rule

$$\mathtt{m}(x_1, x_1, x_2) \equiv x_1$$

and the distributivity rule

$$\mathtt{m}(\mathtt{m}(x_1, x_2, x_4), \mathtt{m}(x_1, x_2, x_5), x_3) \equiv \mathtt{m}(\mathtt{m}(x_1, x_2, x_3), x_4, x_5).$$

> Are these rules sufficient to rewrite any median formula into any other ? If so, is it difficult to simplify a median formula ?

In Chapter 5, we show that it is the case for a certain system of equivalences (System 62, result that has been known at least since Birkhoff ([Bir40]).

The median is an *aggregation* function (see, e.g., [GMMP11]) : it allows us to combine different numerical values in a single one, just as we do by computing a geometrical or arithmetical mean. The goal is to be able to summarize numerical values or to reach a consensus between different choices. In decision theory, statistics, economy, finance, etc., these functions can be used to take into account individual preferences ([Mar09a]). The data to aggregate is then *satisfaction degrees* of *preferences*, i.e., they translate how much an alternative is preferable to another, such as the qualitative appreciation of the food of a restaurant ("terrible < mediocre < good < delicious"), or how much an option is preferable to another one (salad over meatloaf). Marichal ([Mar09b]) showed that discreet *Sugeno integrals* ([Sug74, Mar00]) over distributive lattices can be characterized by a system of equations based on the median. These integrals are studied in the domains of multi-criteria integration and evaluation whenever there is not a total order between preferences. This would be the case if we added an appreciation called "strange" on the restaurant's food in the order above, without knowing anything more than "terrible < strange" and "strange < delicious". Learning these integrals ([BC17]) from real data can be computationally costly and is an ongoing research topic.

Thesis plans and contributions

— In Chapter 3 we begin with a presentation of the theoretical preliminaries we need to approach these questions. Ordered sets and lattice theory allows us to compare the mathematical objects that we manipulate, such as lattice polynomials, Boolean functions, classes of functions, and normal forms themselves. Boolean clones, which are classes of functions that contain the projections and are closed by composition, orient our search of, among other things, candidates for the most efficient connective. In particular, thanks to the complete description of Boolean clones in [Pos41] and of their composition in [CFL06], we can study all possible normal forms, since they correspond to compositions of clones, and, thus, produce interesting connectives, the interpretations of which are clone generators. For instance, the median corresponds of the generator of minimal arity of the clone SM of all self-dual and monotone functions (see Definition 101).
Some models of computations, such as Turing machines, terms, and circuits, as well as some notions of computational complexity allow us to determine the difficulty of solving some of these questions. In particular, Turing machines help us formalize the notion of cost in time or in space of an algorithm ; we also give a Turing machine that solves one of our formula simplification problems (Figure 5.3). Terms and rewriting systems allow us to formalize the notion of formula and function representation, that we use throughout this thesis.

— In Chapter 4, we present the median more precisely with an historical and geometrical point of view, as a point for which the sum of the distance to a family of points forms a median for this family, and with a statistical point of view, as a characteristic of a sequence.
We mention the algebraic and combinatory studies of the median over ordered sets, trees, etc., that was born in the 20th century.
Finally, in Section 4.4, we mention some very recent works that have been done on the median in circuits and the benefits that it brings in terms of, e.g., minimization and logical synthesis.

— In Chapter 5, after we precisely defined median terms and lattice polynomials, that is, the objects we want to represent using those median terms, we recall an equational specification of median formulas that contains, in particular, both rules (majority and distributivity) given above (System 62). This specification is both *sound* and *complete* (Theorem 64), a consequence of which is that any rewriting of a formula keeps logical equivalence, and that any formula can be rewritten in any other that is equivalent to it. The latter point is particularly interesting : indeed, it means that this specification alone allows us to simplify a formula as much as possible.
We then approach the proper and precise definition of a median normal form in Section 5.2. We propose a structural description of formulas (Definition 65) which gives a well order on formulas and thus allows us to give a definition of median normal forms based on minimality (Definition 69). This description consists in counting the number of medians that appear in each depth of a formula.
Then, we study the computational complexity of the problem of simplifying median formulas, both for lattice polynomials over distributive lattices and for the representation of Boolean functions. Thus, the decision problem Monotone Smallmed (Definition 91), on lattice polynomials consists in determining, given a median formula and a structural representation, whether there exists another median formula that is equivalent to the input formula but has a structural representation strictly smaller than the input one.

We show that this problem is moderately intractable, that is, it is at least in NP ; in particular, we bound its difficulty with Σ_2^{P} (Theorem 73). We study the similar problem MONOTONE SMALLMED$_S$, which consists in fixing the structural representation beforehand. This problem is simpler and belongs to the class coNP (Theorem 76). So as to make this problem easier to solve, that is, with polynomial-sized rewriting sequences, (Lemma 78), we study System 77) in which the rules are orientated. We lose completeness, however (Proposition 81). Finally, we discuss of the possibility of adding more rules to a rewriting system to keep this completeness.
We naturally extend those results to the non-monotone case, i.e., to represent Boolean functions, by adding the negation to our median formulas and by modifying the system accordingly (Système 85). Surprisingly, allowing the negation in median formulas can produce representations of *monotone* functions that are strictly smaller than if we didn't allow it (Proposition 89).

— In Chapter 6, we study the representation of Boolean functions, this time by authorizing other connectives than the median, and by extending the notion of normal form given in [CFL06]. Boolean clones theory orients and guides this systematic study, for instance by producing connectives and normal forms explicitly thanks to Post's classification ([Pos41], Figure 3.6).
We start with proposing a definition of normal forms systems (NFSs) as sets of stratified terms that can represent all Boolean functions (Definition 100). We prove some properties on these NFSs, most notably on the nature of the connectives they depend on, and reach a classification of Boolean connectives in two classes : *quasi-Sheffer* connectives who, alone with constants and literals, can represent every Boolean function, such as the median or the Sheffer stroke (logical `NAND`) and others, such as the conjunction or the disjunction (Proposition 105). We also prove that the number of necessary connectives in an NFS is less or equal to 2 (Proposition 108), and we determine all NFSs similar to the median normal form system in the sense that they only require a singular connective (or a connective and the negation).
We then study the efficiency of normal forms (Definition 112) to represent functions. We formalize the way terms can be rewritten from one NFS to another, so as to show, for instance, that it is possible to transform a term efficiently, that is, without an exponential explosion of its size, like it might happen sometimes when we convert a DNF into a CNF. Thanks to this formalization we show that adding more connectives doesn't significantly improves the efficiency of the representations, that the arity of connectives doesn't matter either, and that the median normal form, for a certain subset of normal forms, is still the most efficient way of representing the functions corresponding to this subset (Theorem 135 and 137).

— In the last Chapter 7, we discuss open and future research, on circuits that require different complexity measure, or in normal forms in multi-valued logic.

Most of the results of Chapter 5 have been published in [CMPS17, CMPS19]. Those of Chapter 6 have been published in [CLMP20]. The results we have obtained have also been published and communicated at LFA (Rencontres Francophones sur la Logique Floue et ses Applications) [CMP17] and DICE (Developments in Implicit Computational Complexity) [CLM$^+$18].

3

Préliminaires

Ἐγὼ οὖν μοι δοκῶ, ὦ Ἱππία, ὠφελῆσθαι ἀπὸ τῆς ἀμφοτέρων ὑμῶν ὁμιλίας· τὴν γὰρ παροιμίαν ὅτι ποτὲ λέγει, τὸ « χαλεπὰ τὰ καλά, » δοκῶ μοι εἰδέναι.
Il me semble du moins, Hippias, que ta conversation et la sienne ne m'ont point été inutiles, puisque je crois y avoir appris le sens du proverbe : les belles choses sont difficiles.

Hippias Majeur, Platon ([Pla27]).

Sommaire

3.1 Ensembles ordonnés

Dans cette section, nous rappelons certaines définitions et notations concernant les ordres, en particulier les ordres partiels, qui nous serviront durant toute cette étude pour formaliser la comparaison entre les objets considérés : termes, descriptions structurelles de ces termes (

définition 114), systèmes de formes normales (par exemple, théorèmes 115 et 135), etc. Nous adoptons la terminologie de [SB81, DP92].

Une relation binaire $\preceq$ sur un ensemble S est un *quasi-ordre*, ou un *préordre*, si elle est réflexive et transitive, c'est-à-dire si pour tous $x, y, z \in S$

$$x \preceq x \qquad \text{(réflexivité)}$$
$$\text{si } x \preceq y \text{ et } y \preceq z \text{ alors } x \preceq z. \qquad \text{(transitivité)}$$

Un quasi-ordre est un *ordre partiel* s'il est *anti-symétrique*, c'est-à-dire si pour tous $x, y \in S$

$$\text{si } x \preceq y \text{ et } y \preceq x \text{ alors } x = y. \qquad \text{(anti-symétrie)}$$

Si $\preceq$ est un ordre partiel, la structure $\langle S, \preceq \rangle$ est un *ensemble partiellement ordonné*, ou *poset* (de l'anglais *partially ordered set*).

Une relation d'ordre $\preceq$ sur S induit une relation d'*inégalité stricte* $\prec$, définie ainsi : $x \prec y$ si et seulement si $x \preceq y$ et $x \neg y$. Nous utilisons les notations $x \preceq y$ et $y \succeq x$ de manière interchangeable, et nous utilisons aussi le symbole $\|$ qui indique l'incomparabilité de deux éléments : $x \| y$ si et seulement si $x \not\preceq y$ et $y \not\preceq x$ ou, exprimé autrement, si x et y sont incomparables.

Un ensemble ordonné S est une *chaîne* si, pour tous éléments x, y de S, soit $x \preceq y$ soit $y \preceq x$ ou, exprimé autrement, si tous les éléments de S sont comparables deux-à-deux. Par contre, S est une *antichaîne* si tous ses éléments sont incomparables, c'est-à-dire si pour tous $x, y \in S$, $x \preceq y$ seulement si $x = y$. Par exemple, l'ensemble des entiers naturels $\mathbb{N}$ et l'ensemble des réels $\mathbb{R}$, munis de leur ordres naturels, sont des chaînes. Par contre, l'ensemble $\mathbb{N}$ muni de l'ordre « x divise y » n'est pas une chaîne car il existe des paires d'éléments incomparables, tels que 12 et 5. L'ensemble des nombres premiers $\mathbb{P}$, muni de ce même ordre, est une antichaîne.

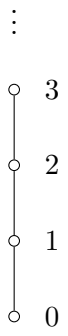
Les ensembles ordonnés peuvent être « dessinés » sous forme de diagrammes. La relation de *couverture* d'un élément par un autre est nécessaire pour les dessiner. L'élément x est *couvert* par l'élément y, ce que nous exprimons aussi par le fait que y *couvre* x, ce que nous notons $x \prec\!\!\!\cdot\; y$ ou $y \succ\!\!\!\cdot\; x$, si $x \prec y$ et si $x \preceq z \prec y$ implique que $z = x$. Exprimé autrement, cette dernière condition requiert qu'il n'existe pas d'élément z tel que $x \prec z \prec y$.

Dans $\mathbb{N}$, $m \prec\!\!\!\cdot\; n$ si et seulement si $n = m + 1$. En revanche, dans $\mathbb{R}$, et dans l'ensemble des rationnels $\mathbb{Q}$, il n'existe pas de couple x, y tel que $x \prec\!\!\!\cdot\; y$; on dit que ces ensembles sont *denses en eux-mêmes*.

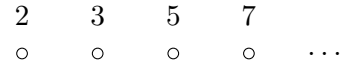
Soit S un ensemble ordonné fini. Nous pouvons représenter S comme une configuration de cercles, qui représentent les éléments de S, et de segments qui les relient, qui représentent la relation de couverture. La construction est la suivante.

1. Associer à chaque élément $x \in S$ un point $P(x)$ du plan euclidien $\mathbb{R}^2$, que l'on signifie par un petit cercle dont le centre est $P(x)$.
2. Pour chaque paire couvrante $x \prec\!\!\!\cdot\; y$ de S, dessiner un segment $\ell(x, y)$ qui joint le cercle associé à $P(x)$ au cercle associé à $P(y)$.
3. Appliquer 1 et 2 en respectant les conditions suivantes :
   a. si $x \prec\!\!\!\cdot\; y$, alors $P(x)$ est « plus bas » que $P(y)$;
   b. le cercle au point $P(z)$ ne coupe pas le segment $\ell(x, y)$ si $x \neq z$ et $z \neq y$.

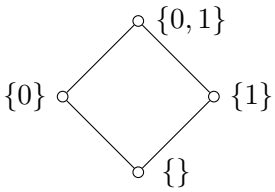
Une configuration qui satisfait les conditions ci-dessus est un *diagramme*, ou *diagramme de Hasse* (voir [Bir40], ainsi que l'utilisation qui en est faite dans [Vog95, Has48, Has53]). Dans la Figure 3.1, nous donnons quelques exemples de diagrammes de Hasse : la chaîne des entiers naturels $\mathbb{N}$ ordonnés par l'ordre naturel sur les entiers $\preceq$, l'antichaîne des nombres premiers $\mathbb{P}$



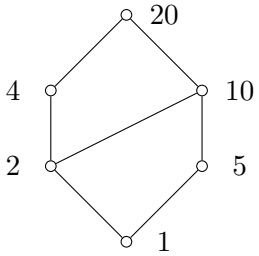
Chaîne des entiers naturels $(\mathbb{N}, \leq)$.



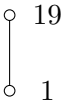
Antichaîne des nombres premiers $(\mathbb{P}, |)$.



Ensemble $(\mathcal{P}(\{0,1\}), \subseteq)$.



Diviseurs de 20.



Diviseurs de 19.

FIGURE 3.1 – Quelques exemples de diagrammes de Hasse.

ordonnés par la relation de divisibilité | ($x|y$ si et seulement si x divise y), et l'ensemble des sous-ensembles de $\{0,1\}$, $\mathcal{P}(\{0,1\})$, ordonnés par l'inclusion ensembliste $\subseteq$. Notons que le nom *chaîne* convient particulièrement bien à l'aspect graphique de la chaîne des entiers naturels.

Un ensemble quasiment ordonné $\langle \mathcal{T}, \leq \rangle$ est *bien-fondé* s'il satisfait la condition des chaînes descendantes, c'est-à-dire s'il n'existe pas de séquence infiniment décroissante

$$\cdots < t_2 < t_1$$

d'éléments de $\mathcal{T}$; exprimé autrement, toute partie non-vide de $\mathcal{T}$ doit posséder un plus petit élément. C'est le cas pour tout $\mathcal{T}$ de cardinalité finie, mais pas, par exemple, de l'ensemble des entiers $\mathbb{Z}$ ou des réels $\mathbb{R}$ munis de l'ordre naturel. Si, pour toute paire (a, b), soit $a \leq b$ soit $b \leq a$, alors $\langle \mathcal{T}, \leq \rangle$ est *totalement ordonné*. Un ensemble totalement ordonné et bien-fondé est un *ensemble bien ordonné*.

Soit n un entier strictement positif, et T_n l'ensemble de tous les n-tuples ordonnés sur S. Posons $T = \bigcup_{n \geq 1} T_n$. L'*extension lexicographique* sur T, que nous notons $\preceq_{lex}$, est définie comme suit :

$$(x_1, \ldots, x_m) \preceq_{lex} (y_1, \ldots, y_n)$$

si

- $m \leq n$ et pour tout $k \in [m]$, $x_k = y_k$ [1], ou
- il existe un entier $k \in [\min(m, n)]$ tel que pour tout $j \in [k-1]$, $x_j = y_j$ et $x_k \prec y_k$ (c'est-à-dire que $x_k \preceq y_k$ et que $x_k \neq y_k$).

1. C'est-à-dire que $(x_1, \ldots, x_m)$ est un *préfixe* de $(y_1, \ldots, y_n)$.

Cet ordre est aussi appelé l'*ordre du dictionnaire* lorsqu'il porte, par exemple, sur des mots du langage naturel :

$$\text{écorce} \preceq \text{orange} \preceq \text{orangeade} \preceq \text{verre}.$$

Exemple 1. L'*ordre lexicographique* défini sur un ensemble de mots fini est bien-fondé. Par contre, pour $S = \{a, b\}$ avec $a \preceq b$, l'extension lexicographique sur le produit infini S^* n'est pas un bon ordre :

$$\cdots \preceq aaab \preceq aab \preceq ab \preceq b.$$

■

3.2 Polynômes latticiels

Dans cette section, nous rappelons certaines définitions et notations concernant les treillis et les fonctions définies sur ceux-ci, en particulier les polynômes latticiels (en anglais, treillis se dit *lattice*), en adoptant la terminologie de [Bir40, SB81, CM10]. En particulier, la représentation de ces fonctions par des formules médianes est un des sujets du chapitre 5.

Un *treillis* est une structure algébrique $\langle L, \wedge_L, \vee_L \rangle$ où L est un ensemble non vide appelé *univers*, et où $\wedge_L$ et $\vee_L$ sont deux opérations binaires qui satisfont les lois suivantes :

1. $x \wedge_L y = y \wedge_L x$, et $x \vee_L y = y \vee_L x$ (commutativité) ;
2. $x \wedge_L (y \wedge_L z) = (x \wedge_L y) \wedge_L z$, et $x \vee_L (y \vee_L z) = (x \vee_L y) \vee_L z$ (associativité) ;
3. $x \wedge_L x = x$, et $x \vee_L x = x$ (idempotence) ;
4. $x = x \vee_L (x \wedge_L y)$, et $x = x \wedge_L (x \vee_L y)$ (absorption).

Remarque 2. Les symboles $\wedge_L$ et $\vee_L$ pour dénoter les opérations sur des treillis sont souvent indiqués par les symboles $\wedge$ et $\vee$, respectivement. Pour ne pas les confondre avec les connecteurs binaires `ET` et `OU` dans le contexte de la logique Booléenne, nous avons indiqué l'univers L en indice sur ces opérations, convention que nous adoptons pour la plupart des notions relatives aux treillis. Dans son ouvrage [Bir40], Birkhoff utilise les notations $\frown$ et $\smile$ respectivement, qui peuvent plutôt induire une interprétation ensembliste des objets manipulés.

Remarque 3. Un treillis peut également être défini comme un ensemble ordonné vérifiant certaines propriétés de la manière suivante. Soit P un ensemble ordonné et soit $S \subseteq P$ un sous-ensemble de P. Un élément $x \in P$ est une *borne supérieure* de S si $s \leq x$ pour tout $s \in S$. Une *borne inférieure* est définie de manière duale. L'ensemble de toutes les bornes supérieures (respectivement inférieures) de S est noté S^u pour *"upper"* (respectivement S^ℓ pour *"lower"*) :

$$S^u := \{x \in P \colon \forall s \in S, s \leq x\}, \quad \text{et}$$
$$S^\ell := \{x \in P \colon \forall s \in S, s \geq x\}.$$

Si S^u (respectivement S^ℓ) a un plus petit (respectivement plus grand) élément x, alors il est unique, et x est appelé *la plus petite borne supérieure* ou *le supremum* de S, noté $\sup S$ (respectivement *la plus grande borne inférieure* ou *l'infimum* de S, noté $\inf S$).

Soit P un ensemble ordonné non vide. Si $\inf\{x, y\}$ et $\sup\{x, y\}$ existent pour tous $x, y \in P$, alors $\langle P, \wedge_L, \vee_L \rangle$ est un treillis avec les notations $x \wedge_L y = \inf\{x, y\}$ et $x \vee_L y = \sup\{x, y\}$. De même que pour les ensembles ordonnés, nous pouvons représenter certains treillis par des diagrammes de Hasse, et, par exemple, lire immédiatement sur un diagramme le *join* et le *meet* de deux éléments.

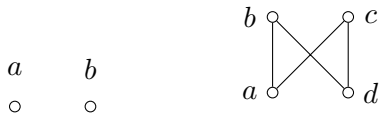


FIGURE 3.2 – Exemples d'ensembles ordonnés qui ne sont pas des treillis.

Exemple 4. Nous donnons ici un exemple de treillis et de calcul du join et du meet de deux éléments. Dans la figure 3.1, nous avons représenté un diagramme de Hasse de l'ensemble des diviseurs de 20, ordonnés par la division $\mid$. Ici, $\wedge_L$ est le plus petit commun diviseur, et $\vee_L$ est le plus grand commun multiple ; par exemple, $4 \wedge_L 10 = 2$, $4 \wedge_L 2 = 2$, $2 \vee_L 5 = 10$, et $2 \vee_L 10 = 10$. ■

Un treillis est dit *distributif* si les deux opérations sont distributives l'une par rapport à l'autre. Lorsqu'il n'y a pas d'ambiguïté, nous désignons un treillis $\langle L, \wedge_L, \vee_L \rangle$ par son univers L.

Dans ce chapitre, L désigne toujours un treillis arbitraire, distributif, et borné, dont le maximum et le minimum sont respectivement $\top$ et $\bot$.

Étant donnés $a, b \in L$, $a \leq b$ signifie que $a \wedge_L b = a$ ou, de manière équivalente, que $a \vee_L b = b$. Pour tout entier $n \geq 1$, on note $[n] = \{1, \ldots, n\}$. Étant donné un ensemble arbitraire non vide A et un treillis L, l'ensemble L^A de toutes les fonctions de A dans L est aussi un treillis sous les opérations

$$(f \wedge_L g)(x) = f(x) \wedge_L g(x)$$

et

$$(f \vee_L g)(x) = f(x) \vee_L g(x)$$

pour tous $f, g \in L^A$. En particulier, tout treillis L induit une structure de treillis sur le produit cartésien L^n, $n \geq 1$, en définissant les opérations $\wedge_L$ et $\vee_L$ composante par composante, c'est-à-dire de la manière suivante :

$$(a_1, \ldots, a_n) \wedge_L (b_1, \ldots, b_n) = (a_1 \wedge_L b_1, \ldots, a_n \wedge_L b_n),$$

$$(a_1, \ldots, a_n) \vee_L (b_1, \ldots, b_n) = (a_1 \vee_L b_1, \ldots, a_n \vee_L b_n).$$

Par la suite, nous désignons les éléments de L par des lettres minuscules $a, b, c, \ldots$, et les éléments de L^n, $n \geq 1$ par des lettres minuscules en gras $\mathbf{a}, \mathbf{b}, \mathbf{c}, \ldots$.

Nous rappelons à présent la notion de fonctions latticielles polynomiales. La classe des *fonctions latticielles polynomiales sur des treillis* (ou, plus simplement, *fonctions polynomiales*) de L^n dans L est définie inductivement comme l'ensemble des fonctions représentées par des expressions construites dans le langage des treillis. Ainsi, sont des fonctions polynomiales :

1. les projections $\mathbf{x} \mapsto x_i$ et les fonctions constantes $\mathbf{x} \mapsto c$ avec $c \in L$, et
2. les fonctions $f \wedge_L g$ et $f \vee_L g$, si f et g sont des fonctions polynomiales.

Le qualificatif « polynomial » provient de la similarité des formes obtenues avec les polynômes usuels, en remplaçant les occurrences de $\vee_L$ par l'addition $+$ et celles de $\wedge_L$ par la multiplication $\times$.

Exemple 5. Soit u_L la fonction latticielle polynomiale définie par

$$u_L(x_1, x_2, x_3) := (x_1 \vee_L x_2) \wedge_L x_3$$

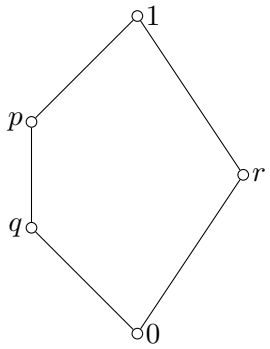


FIGURE 3.3 – Diagramme de Hasse du pentagone N_5.

sur le treillis N_5 représenté dans la figure 3.3. Alors

$$\begin{aligned} u_L(p,q,r) &= (p \vee_L q) \wedge_L r && \text{par définition} \\ &= p \wedge_L r && \text{car } q \preceq p \\ &= 0. \end{aligned}$$

■

3.3 Fonctions Booléennes et théorie des clones

Dans cette section, nous rappelons quelques notions de base sur la théorie des clones ainsi que sur les systèmes de formes normales dans le contexte des fonctions Booléennes. Pour plus d'informations sur la théorie des clones, nous nous référons à [Lau06]. Nous utilisons la théorie des clones Booléens dans le chapitre 6, comme un guide et une manière de déterminer les fonctions et les formes normales qui nous intéressent.

3.3.1 Fonctions Booléennes

Dans ce chapitre, nous indiquons par $\mathbb{B}$ l'ensemble de cardinal 2, $\{0,1\}$. Nous rappelons que les tuples sont souvent indiqués par des caractères gras et leurs éléments par des lettres en italiques indicées ; par exemple, $\mathbf{a} = (a_1, \ldots, a_n)$. La *distance de Hamming* entre deux tuples $\mathbf{a}$ et $\mathbf{b}$, notée $\delta(\mathbf{a}, \mathbf{b})$, est le nombre de positions en lesquelles ils sont différents. Par exemple,

$$\begin{aligned} &\delta((0,0,0,1,1), \\ &\quad (0,0,\underset{\uparrow}{1},1,\underset{\uparrow}{0})) = 2. \end{aligned}$$

Le *poids de Hamming* d'un tuple $\mathbf{a}$, noté $\mathrm{w}(\mathbf{a})$ (de l'anglais *weight*), est défini comme le nombre d'entrées de $\mathbf{a}$ non nulles, c'est-à-dire que $\mathrm{w}(\mathbf{a}) := \delta(\mathbf{a}, (0, \ldots, 0))$. Par exemple,

$$\mathrm{w}((1,1,0,0,1,1,0)) = 4.$$

L'ensemble $\mathbb{B}$ est doté de l'ordre naturel $0 \leq 1$. L'ensemble $\mathbb{B}^n$ peut donc être doté de l'ordre composante-par-composante des tuples, c'est-à-dire que $(a_1, \ldots, a_n) \leq (b_1, \ldots, b_n)$ si et seulement si pour tout $i \in [n]$ on a $a_i \leq b_i$. On dit que le tuple $\mathbf{a}$ *couvre* le tuple $\mathbf{b}$ si $\mathbf{a} < \mathbf{b}$ et s'il

a	$\neg(a)$
0	1
1	0

a	b	$a \vee b$	$a \wedge b$	$a \oplus b$	$a \uparrow b$	$a \downarrow b$
0	0	0	0	0	1	1
0	1	1	0	1	1	0
1	0	1	0	1	1	0
1	1	1	1	0	0	0

a	b	c	$\mu(a,b,c)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

TABLE 3.1 – Quelques fonctions Booléennes usuelles.

n'existe pas d'autre tuple $\mathbf{c}$ tel que $\mathbf{a} < \mathbf{c} < \mathbf{b}$; cette définition de couverture est à rapprocher de la définition donnée dans la section 3.1.

Une *fonction Booléenne* est une opération $f : \mathbb{B}^n \to \mathbb{B}$, pour un entier donné $n \geq 0$ que l'on appelle l'*arité* de f. L'ensemble des fonctions Booléennes de $\mathbb{B}^n \to \mathbb{B}$ est noté $\mathbb{B}^{\mathbb{B}^n}$. L'arité de f est notée $\mathrm{ar}(f)$. Pour une arité donnée n, les n différentes *projections* sont les fonctions définies par

$$e_i^{(n)} : (a_1, \ldots, a_n) \mapsto a_i, \qquad 1 \leq i \leq n.$$

Les opérations d'arité nulle sont les constantes qui correspondent aux éléments de $\mathbb{B}$. Sans qu'il n'y ait d'ambiguïté, nous notons les fonctions constantes d'arité quelconque par $\mathbf{0}$ (respectivement $\mathbf{1}$) pour les fonctions qui prennent la valeur 0 (respectivement 1) en tous points.

Nous donnons dans la Table 3.1 les définitions par tables de vérité de certaines fonctions Booléennes usuelles : la fonction unaire $\neg$ (négation, `NON`, `NOT`), les fonctions binaires $\vee$ (disjonction, `OU`, `OR`) et $\wedge$ (conjonction, `ET`, `AND`), $\oplus$ (addition modulo 2, ou exclusif, `XOR`), $\uparrow$ (*Sheffer stroke*, opérateur de Sheffer, conjonction niée, `NAND`), $\downarrow$ (*Peirce's arrow*, opérateur de Peirce, disjonction niée, `NOR`), et la fonction ternaire μ (majorité).

Nous utilisons parfois la notation préfixée (ou notation polonaise) ou infixée ; par exemple, $\wedge(a_1, a_2) = a_1 \wedge a_2$. Étant donnée une fonction binaire f, soit f_n la fonction définie de manière inductive par :

— $f_2(a_1, a_2) = f(a_1, a_2)$, et
— $f_{n+1}(a_1, \ldots, a_{n+1}) = f(a_1, f_n(a_1, \ldots, a_n))$.

Par exemple, $\wedge_3(a_1, a_2, a_3) = \wedge(a_1, \wedge(a_2, a_3))$.

Un tuple $\mathbf{a}$ est un *point vrai* (respectivement un *point faux*) d'une fonction f si $f(\mathbf{a}) = 1$ (respectivement si $f(\mathbf{a}) = 0$). Si $\mathbf{a}$ est un point vrai de f et qu'il n'existe pas d'autre point vrai $\mathbf{b}$ de f tel que $\mathbf{b} < \mathbf{a}$ alors on dit que $\mathbf{a}$ est un *point vrai minimal*. Si $\mathbf{a}$ est un point faux de f et qu'il n'existe pas d'autre point faux $\mathbf{b}$ de f tel que $\mathbf{a} < \mathbf{b}$ alors on dit que $\mathbf{a}$ est un *point faux maximal*.

Étant donné un point $a \in \mathbb{B}$, on dit qu'une fonction $f : \mathbb{B}^n \to \mathbb{B}$ est *a-préservatrice* si $f(a, \ldots, a) = a$. Une fonction *préserve les constantes* si elle est à la fois 0- et 1-préservatrice. Étant donnée une fonction $f : \mathbb{B}^n \to \mathbb{B}$, le *dual* de f est défini par

$$f^{\mathrm{d}}(a_1, \ldots, a_n) := \neg(f(\neg(a_1), \ldots, \neg(a_n))).$$

Une fonction $f : \mathbb{B}^n \to \mathbb{B}$ est *auto-duale* si $f = f^{\mathrm{d}}$. Une fonction $f : \mathbb{B}^n \to \mathbb{B}$ est *symétrique* si, pour toute permutation π de $\{1, \ldots, n\}$,

$$f(a_1, \ldots, a_n) = f(a_{\pi(i)}, \ldots, a_{\pi(n)})$$

pour tous $a_1, \ldots, a_n \in \mathbb{B}$. Une fonction $f : \mathbb{B}^n \to \mathbb{B}$ est *monotone* si pour tous $\mathbf{a}, \mathbf{b} \in \mathbb{B}^n$, $\mathbf{a} \leq \mathbf{b}$ induit $f(\mathbf{a}) \leq f(\mathbf{b})$.

Une fonction $f : \mathbb{B}^n \to \mathbb{B}$ est *croissante* (respectivement *décroissante* en son i-ème argument si pour tous $a_1, \ldots, a_n, b_i \in \mathbb{B}$, $a_i \leq b_i$ induit

$$f(a_1, \ldots, a_{i-1}, a_i, a_{i+1}, \ldots, a_n) \leq f(a_1, \ldots, a_{i-1}, b_i, a_{i+1}, \ldots, a_n)$$
$$(f(a_1, \ldots, a_{i-1}, a_i, a_{i+1}, \ldots, a_n) \geq f(a_1, \ldots, a_{i-1}, b_i, a_{i+1}, \ldots, a_n) \text{ respectivement}).$$

Une fonction est *pseudo-monotone* si elle est croissante ou décroissante en chaque argument.

Fait 6. Une fonction $f : \mathbb{B}^n \to \mathbb{B}$ est pseudo-monotone si et seulement si il existe une fonction monotone $g : \mathbb{B}^n \to \mathbb{B}$ et un sous-ensemble $S \subseteq \{1, \ldots, n\}$ tels que pour tous $a_1, \ldots, a_n \in \mathbb{B}$,

$$f(a_1, \ldots, a_n) = g(l_1, \ldots, l_n)$$

où $l_i = a_i$ si $i \in S$ et $l_i = \neg(a_i)$ si $i \notin S$.

Le i-ème argument d'une fonction $f : \mathbb{B}^n \to \mathbb{B}$ est dit *essentiel* s'il existe $(a_1, \ldots, a_n) \in \mathbb{B}^n$ tel que

$$f(a_1, \ldots, a_{i-1}, 0, a_{i+1}, \ldots, a_n) \neq f(a_1, \ldots, a_{i-1}, 1, a_{i+1}, \ldots, a_n).$$

Deux fonctions f et g sont *équivalentes*, ce que l'on note $f \cong g$, si l'on peut obtenir l'une à partir de l'autre par des permutations des arguments et par addition ou suppression d'arguments non-essentiels.

Fait 7. Le nombre d'arguments essentiels est un invariant par dualité et équivalence des fonctions.

Pour plus d'informations sur les arguments essentiels et non-essentiels d'une fonction, nous nous référons à [Sal63, Wil96, CLW12, CLW15].

Exemple 8. Afin d'illustrer les notions présentées dans cette section, pour toutes les fonctions Booléennes $\mathbf{0}, \mathbf{1}, \neg, \vee, \wedge, \oplus, \uparrow, \downarrow$, et μ, nous donnons dans la Table 3.2 leur dual, puis nous indiquons si elles sont auto-duales, symétriques, monotones, croissantes ou décroissantes en l'argument i (les propriétés indiquées sont valables pour tous i), ou pseudo-monotone. La projection $e_i^{(n)}$ est à la fois 0- et 1-préservatrice et auto-duale. Elle est symétrique si et seulement si $n = 1$. Elle est monotone, croissante en chaque argument, décroissante en chaque argument sauf le $i$ème, et pseudo-monotone.

Chaque argument est essentiel pour les fonctions $\neg, \vee, \wedge, \oplus, \uparrow, \downarrow$, et μ. Aucun argument n'est essentiel pour les fonctions $\mathbf{0}$ et $\mathbf{1}$. Le seul argument essentiel de $e_i^{(n)}$ est le $i$ème.

Toutes les projections sont équivalentes. Les fonctions constantes qui prennent la même valeur en tous points sont équivalentes. Les fonctions de la Table 3.2 sont deux-à-deux non équivalentes. ■

Grâce aux diagrammes de Hasse, nous pouvons naturellement représenter des fonctions Booléennes d'arité raisonnable (par exemple, à moins de 5 entrées). Cette représentation a la même « complexité » que les tables de vérités, puisque le domaine entier de la fonction est représenté, mais a l'avantage d'être graphique, et d'afficher ainsi directement certaines propriétés intéressantes telles que, par exemple, la couverture de points, la croissance, ou l'auto-dualité de la

	0	**1**	$\neg$	$\vee$	$\wedge$	$\oplus$	$\uparrow$	$\downarrow$	μ
dual	**1**	**0**	$\neg$	$\wedge$	$\vee$	$\neg \circ \oplus$	$\downarrow$	$\uparrow$	μ
0-préservatrice	oui	non	non	oui	oui	oui	non	non	oui
1-préservatrice	non	oui	non	oui	oui	oui	non	non	oui
auto-duale	non	non	oui	non	non	non	non	non	oui
symétrique	oui	oui	oui	oui	oui	oui	oui	oui	oui
monotone	oui	oui	non	oui	oui	non	non	non	oui
croissante en l'argument i	oui	oui	non	oui	oui	non	non	non	oui
décroissante en l'argument i	oui	oui	oui	non	non	non	oui	oui	non
pseudo-monotone	oui	oui	oui	oui	oui	non	oui	oui	oui

TABLE 3.2 – Propriétés de quelques fonctions Booléennes connues.

fonction. Étant donnée une fonction Booléenne f que nous voulons représenter ainsi, nous pouvons dessiner un diagramme de Hasse d'un treillis Booléen de taille $2^{\mathrm{ar}(f)}$. Chaque point $P(x)$ du diagramme correspond à un point du domaine $\mathbb{B}^{\mathrm{ar}(f)}$ de f, et est coloré suivant la valeur de $f(x)$: si $f(x) = 1$ (respectivement 0) alors $P(x)$ est représenté par un disque plein (respectivement un cercle vide). Nous donnons quelques exemples dans la Figure 3.4. En particulier, la monotonie (la croissance, plus précisément) des fonctions $x \wedge y$ et $(x \wedge y) \vee (z \wedge t)$ y est visible directement : aucun point faux ne couvre un point vrai. De même, la non-monotonie de la fonction $(x \wedge y) \vee (\overline{x} \wedge z)$ est visible directement, car un point faux couvre un point vrai, situation que nous avons grisée : ce sont les points $(0, 0, 1)$ et $(1, 0, 1)$. Notons que dans le cas des fonctions (Booléennes) ternaires, la représentation est très similaire à celle du cube unité dans l'espace $\mathbb{R}^3$, illustré dans la Figure 3.5. Pour des dimensions supérieures, c'est-à-dire pour des fonctions d'arité supérieure à 4, les représentations deviennent plus difficiles à dessiner.

Une *classe* de fonctions Booléennes est un sous-ensemble $\mathcal{C} \subseteq \bigcup_{n \geq 1} \mathbb{B}^{\mathbb{B}^n}$. Si f est n-aire et si $g_1, \ldots, g_n$ sont toutes m-aires, alors leur *composition* est la fonction m-aire obtenue par

$$f(g_1, \ldots, g_n)(a_1, \ldots, a_m) \;=\; f(g_1(a_1, \ldots, a_m)), \ldots, g_n(a_1, \ldots, a_m))),$$

pour tout $(a_1, \ldots, a_m) \in \mathbb{B}^m$. Cette notion s'étend naturellement aux classes de fonctions. La *composition de la classe $\mathcal{I}$ avec la classe $\mathcal{J}$*, que l'on note $\mathcal{I} \circ \mathcal{J}$, est définie par

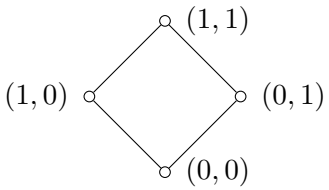
$$\mathcal{I} \circ \mathcal{J} := \{f(g_1, \ldots, g_n) \mid n, m \geq 1, f \text{ } n\text{-ary in } \mathcal{I}, g_1, \ldots, g_n \text{ } m\text{-ary in } \mathcal{J}\}.$$

3.3.2 Clones de fonctions Booléennes

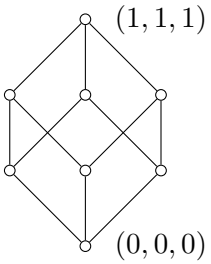
Un *clone* (de l'anglais *closed one*, c'est-à-dire « qui est fermé », en un sens que nous précisons ci-après) est une classe $\mathcal{C}$ de fonctions Booléennes qui contient toutes les projections et qui satisfait

$$\mathcal{C} \circ \mathcal{C} = \mathcal{C},$$

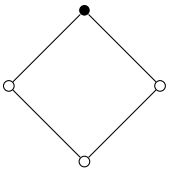
c'est-à-dire qu'elle est fermée par composition. Dans ce document, nous parlons surtout de clones Booléens car ici les fonctions considérées ont pour domaine $\{0, 1\}$; mais les clones en général peuvent être définis pour un ensemble fini contenant au moins 2 éléments. Voir, par exemple, [Csá05, KPS14]. Le nom de "clones" leur a été attribué par Philip Hall, selon Paul Cohn [Coh81]. Leur étude en tant qu'objets mathématiques est plus ancienne : Émile Post [Pos41] a dressé, à partir des années 20, une description complète des « classes de fonctions fermées » sur un ensemble à deux éléments. Associé à l'ordre de l'inclusion ensembliste $\subset$, l'ensemble des clones



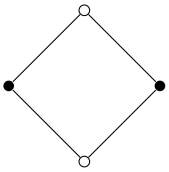
Treillis Booléen à 2^2 éléments.



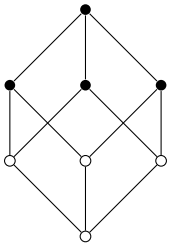
Treillis Booléen à 2^3 éléments.



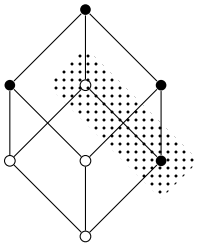
Représentation de $f(x, y) = x \wedge y$.



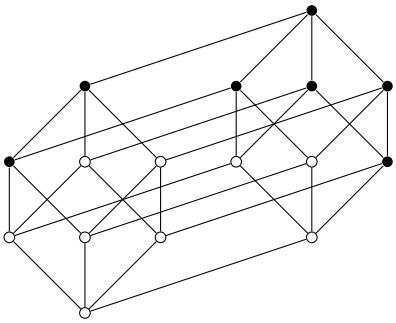
Représentation de $f(x, y) = x \oplus y$.



Représentation de $f(x, y, z) = \mu(x, y, z)$.



Représentation de $f(x, y, z) = (x \wedge y) \vee (\overline{x} \wedge z)$.



Représentation de $f(x, y, z, t) = (x \wedge y) \vee (z \wedge t)$.

FIGURE 3.4 – Représentation de fonctions Booléennes par des diagrammes de Hasse.

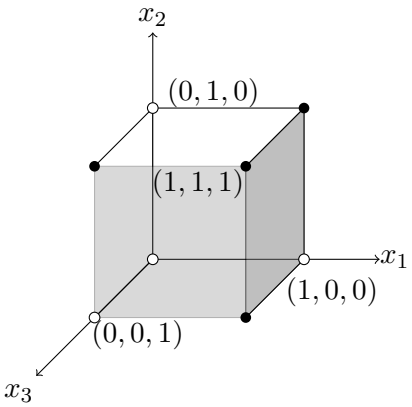


FIGURE 3.5 – Représentation de la fonction médiane sur les sommets du cube unité de $\mathbb{R}^3$.

des fonctions Booléennes constitue un *treillis algébrique*, dont le plus grand clone est l'ensemble de toutes les fonctions Booléennes et dont le plus petit est l'ensemble de toutes les projections ; et où le *meet* $\wedge_L$ de deux clones est leur intersection tandis que le *join* $\vee_L$ de deux clones est le plus petit clone qui contient leur union. Par exemple, le meet du clone des fonctions Booléennes monotones M et du clone des fonctions Booléennes auto-duales S est le clone des fonctions Booléennes auto-duales monotones :

$$M \wedge_L S = M \cap S = SM.$$

Ce treillis, appelé le *Treillis de Post*, a été entièrement décrit dans [Pos41]. Nous reproduisons dans la figure 3.6 un diagramme de Hasse de ce treillis. La séparation en trait pointillés indique les clones *précomplets* qui vérifient $\mathcal{C} \circ \Omega(1)$ et ceux qui ne vérifient pas cette propriété ; voir, ci-dessous, la définition 11, ainsi que la proposition 105 au chapitre 6. Dans la table 3.3, nous reproduisons la table de composition de clones, entièrement déterminée dans [CFL06]. Dans cette table, le résultat de $\mathcal{C}_1 \circ \mathcal{C}_2$ est indiqué. Les lignes correspondent à $\mathcal{C}_1$ et les colonnes à $\mathcal{C}_2$. Lorsque $\mathcal{C}_1 \circ \mathcal{C}_2$ n'est pas un clone, nous avons écrit $\mathcal{C}_1 \vee \mathcal{C}_2$ entre crochets. Les compositions marquées du symbole † sont des clones seulement si $m = 2$. Nous désignons par p le minimum $\min(m, n)$.

Nous utilisons la nomenclature de [FP04, CFL06].

— Le clone de toutes les fonctions Booléennes est noté Ω.
— Pour $a \in \mathbb{B}$, le clone des fonctions a-préservatrices est noté T_a, et $T_c := T_0 \cap T_1$ est le clone des fonctions qui préservent les constantes.
— Le clone de toutes les fonctions monotones est noté M, et $M_x := M \cap T_x$ pour $x \in \{0, 1, \mathrm{c}\}$.
— Le clone de toutes les fonctions auto-duales est noté S, et $S_\mathrm{c} := S \cap T_\mathrm{c}$, et $SM := S \cap M$.
— Le clone de toutes les fonctions linéaires est noté L, c'est-à-dire que

$$L := \{f \in \Omega \mid f \cong \oplus_n \text{ ou } f \cong \neg(\oplus_n) \text{ pour } n \geq 2\} \cup \{e_i^{(n)}, \neg(e_i^{(n)}) \mid 1 \leq i \leq n\} \cup \{\mathbf{0}, \mathbf{1}\},$$

$L_x := L \cap T_x$, pour $x \in \{0, 1, \mathrm{c}\}$, et $LS := L \cap S$.

Un ensemble $A \subseteq \mathbb{B}^n$ est dit *a-séparant* pour un certain $a \in \mathbb{B}$ s'il existe un i, $1 \leq i \leq n$ tel que pour tout $(a_1, \ldots, a_n) \in A$ on a $a_i = a$. Une fonction f est dite *a-séparatrice* si $f^{-1}(a)$ est a-séparant. La fonction f est dite *a-séparatrice de rang $k \geq 2$* si chaque sous-ensemble $a \subseteq f^{-1}(a)$ de taille au plus k est a-séparateur.

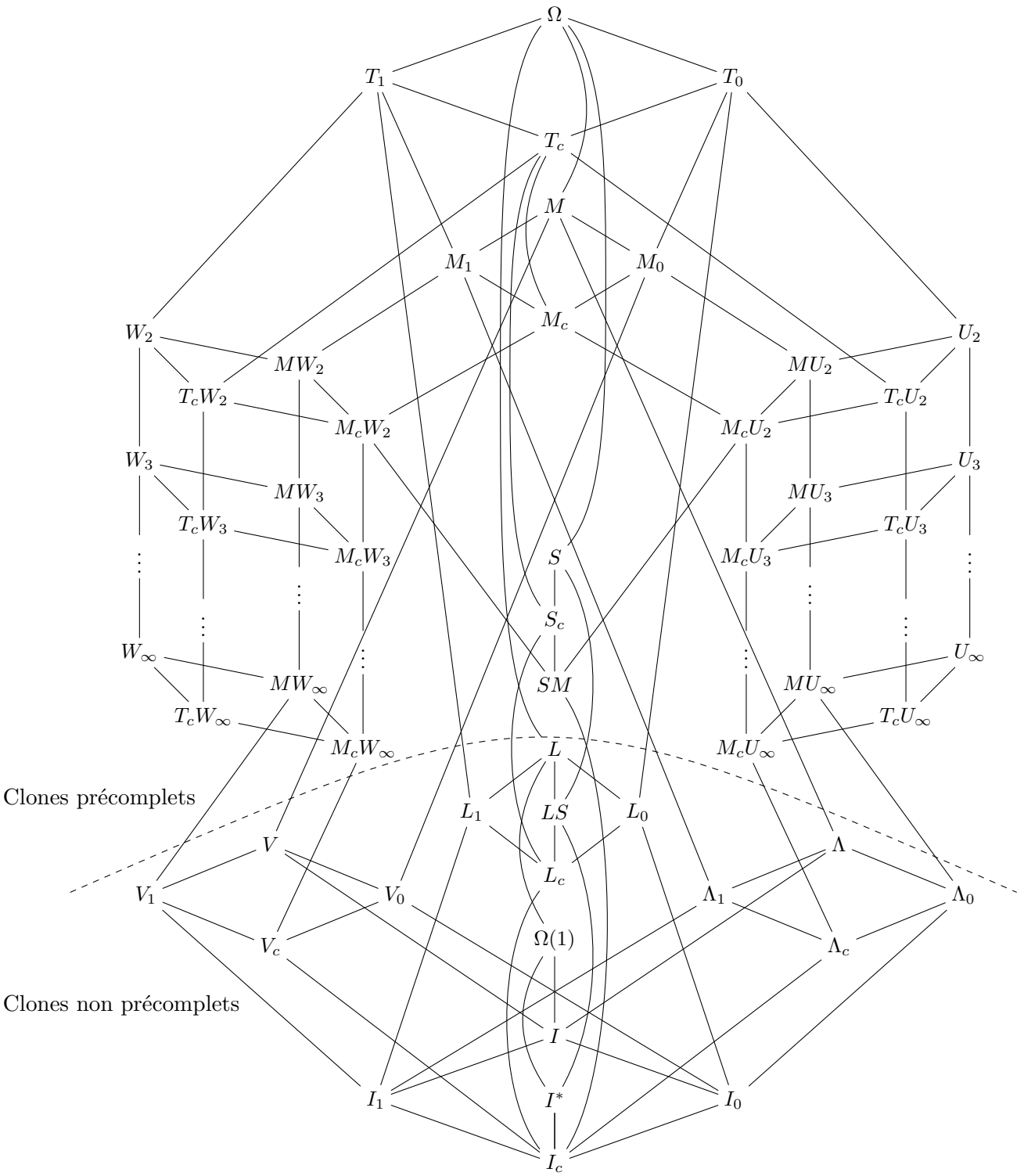


FIGURE 3.6 – Treillis de Post.

$\circ$	Ω	T_0	T_1	T_c	M	M_0	M_1	M_c	S	S_c	SM	L	L_0	L_1	LS	L_c	Λ	Λ_0	Λ_1	Λ_c	V	V_0	V_1	V_c	U_n	MU_n	T_cU_n	M_cU_n	W_n	MW_n	T_cW_n	M_cW_n	$\Omega(1)$	I	I_0	I_1	I^*	I_c
Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω
T_0	Ω	T_0	Ω	T_0	Ω	T_0	Ω	T_0	Ω	T_0	T_0	Ω	T_0	Ω	Ω	T_0	Ω	T_0	Ω	T_0	Ω	T_0	Ω	T_0	T_0	T_0	T_0	T_0	Ω	Ω	T_0	T_0	Ω	Ω	T_0	Ω	Ω	T_0
T_1	Ω	Ω	T_1	T_1	Ω	Ω	T_1	T_1	Ω	T_1	T_1	Ω	Ω	T_1	Ω	T_1	Ω	Ω	T_1	T_1	Ω	Ω	T_1	T_1	Ω	Ω	T_1	T_1	T_1	T_1	T_1	T_1	Ω	Ω	Ω	T_1	Ω	T_1
T_c	Ω	T_0	T_1	T_c	Ω	T_0	T_1	T_c	Ω	T_c	T_c	Ω	T_0	T_1	Ω	T_c	Ω	T_0	T_1	T_c	Ω	T_0	T_1	T_c	T_0	T_0	T_c	T_c	T_1	T_1	T_c	T_c	Ω	Ω	T_0	T_1	Ω	T_c
M	Ω	$[\Omega]$	$[\Omega]$	$[\Omega]$	M	M	M	M	Ω	$[\Omega]$	M	Ω	$[\Omega]$	$[\Omega]$	Ω	$[\Omega]$	M	M	M	M	M	M	M	M	$[\Omega]$	M	$[\Omega]$	M	$[\Omega]$	M	$[\Omega]$	M	Ω	M	M	M	Ω	M
M_0	Ω	T_0	$[\Omega]$	$[T_0]$	M	M_0	M	M_0	Ω	$[T_0]$	M_0	Ω	T_0	$[\Omega]$	Ω	$[T_0]$	M	M_0	M	M_0	M	M_0	M	M_0	T_0	M_0	$[T_0]$	M_0	$[\Omega]$	M	$[T_0]$	M_0	Ω	M	M_0	M	Ω	M_0
M_1	Ω	$[\Omega]$	T_1	$[T_1]$	M	M	M_1	M_1	Ω	$[T_1]$	M_1	Ω	$[\Omega]$	T_1	Ω	$[T_1]$	M	M	M_1	M_1	M	M	M_1	M_1	$[\Omega]$	M	$[T_1]$	M_1	T_1	M_1	$[T_1]$	M_1	Ω	M	M	M_1	Ω	M_1
M_c	Ω	T_0	T_1	T_c	M	M_0	M_1	M_c	Ω	T_c	M_c	Ω	T_0	T_1	Ω	T_c	M	M_0	M_1	M_c	M	M_0	M_1	M_c	T_0	M_0	T_c	M_c	T_1	M_1	T_c	M_c	Ω	M	M_0	M_1	Ω	M_c
S	Ω	Ω	Ω	$[\Omega]$	Ω	Ω	Ω	$[\Omega]$	S	S	S	Ω	Ω	Ω	S	S	Ω	Ω	Ω	$[\Omega]$	Ω	Ω	Ω	$[\Omega]$	Ω	Ω	$[\Omega]$	$[\Omega]$	Ω	Ω	$[\Omega]$	$[\Omega]$	Ω	Ω	Ω	Ω	S	S
S_c	Ω	T_0	T_1	T_c	Ω	T_0	T_1	T_c	S	S_c	S_c	Ω	T_0	T_1	S	S_c	Ω	T_0	T_1	T_c	Ω	T_0	T_1	T_c	T_0	T_0	T_c	T_c	T_1	T_1	T_c	T_c	Ω	Ω	T_0	T_1	S	S_c
SM	Ω	T_0	T_1	T_c	M	M_0	M_1	M_c	S	S_c	SM	Ω	T_0	T_1	S	S_c	M	MU_2	M_1	M_cU_2	M	M_0	MW_2	M_cW_2	U_2	MU_2	T_cU_2	M_cU_2	W_2	MW_2	T_cW_2	M_cW_2	Ω	M	MU_2	MW_2	S	SM
L	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	$[\Omega]$	$[\Omega]$	$[\Omega]$	L	L	L	L	L	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	Ω	L	L	L	L	L	L
L_0	Ω	T_0	Ω	T_0	Ω	T_0	Ω	T_0	$[\Omega]$	$[T_0]$	$[T_0]$	L	L_0	L	L	L_0	Ω	T_0	Ω	T_0	Ω	T_0	Ω	T_0	T_0	T_0	T_0	T_0	Ω	Ω	T_0	T_0	L	L	L_0	L	L	L_0
L_1	Ω	Ω	T_1	T_1	Ω	Ω	T_1	T_1	$[\Omega]$	$[T_1]$	$[T_1]$	L	L	L_1	L	L_1	Ω	Ω	T_1	T_1	Ω	Ω	T_1	T_1	Ω	Ω	T_1	T_1	T_1	T_1	T_1	T_1	L	L	L	L_1	L	L_1
LS	Ω	Ω	Ω	$[\Omega]$	Ω	Ω	Ω	$[\Omega]$	S	S	S	L	L	L	LS	LS	Ω	Ω	Ω	$[\Omega]$	Ω	Ω	Ω	$[\Omega]$	Ω	Ω	$[\Omega]$	$[\Omega]$	Ω	Ω	$[\Omega]$	$[\Omega]$	L	L	L	L	LS	LS
L_c	Ω	T_0	T_1	T_c	Ω	T_0	T_1	T_c	S	S_c	S_c	L	L_0	L_1	LS	L_c	Ω	T_0	T_1	T_c	Ω	T_0	T_1	T_c	T_0	T_0	T_c	T_c	T_1	T_1	T_c	T_c	L	L	L_0	L_1	LS	L_c
Λ	Ω	$[\Omega]$	$[\Omega]$	$[\Omega]$	M	M	M	M	$[\Omega]$	$[\Omega]$	$[M]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	Λ	Λ	Λ	Λ	M	M	M	M	$[\Omega]$	$[M]$	$[\Omega]$	$[M]$	$[\Omega]$	M	$[\Omega]$	M	$[\Omega]$	Λ	Λ	Λ	$[\Omega]$	Λ
Λ_0	Ω	T_0	$[\Omega]$	$[T_0]$	M	M_0	M	M_0	$[\Omega]$	$[T_0]$	MU_2	$[\Omega]$	$[T_0]$	$[\Omega]$	$[\Omega]$	$[T_0]$	Λ	Λ_0	Λ	Λ_0	M	M_0	M	M_0	U_n	MU_n	$[U_n]$	MU_n	$[\Omega]$	M	$[T_0]$	M_0	$[\Omega]$	Λ	Λ_0	Λ	$[\Omega]$	Λ_0
Λ_1	Ω	$[\Omega]$	T_1	$[T_1]$	M	M	M_1	M_1	$[\Omega]$	$[T_1]$	$[M_1]$	$[\Omega]$	$[\Omega]$	$[T_1]$	$[\Omega]$	$[T_1]$	Λ	Λ	Λ_1	Λ_1	M	M	M_1	M_1	$[\Omega]$	$[M]$	$[T_1]$	$[M_1]$	T_1	M_1	$[T_1]$	M_1	$[\Omega]$	Λ	Λ	Λ_1	$[\Omega]$	Λ_1
Λ_c	Ω	T_0	T_1	T_c	M	M_0	M_1	M_c	$[\Omega]$	$[T_c]$	M_cU_2	$[\Omega]$	$[T_0]$	$[T_1]$	$[\Omega]$	$[T_c]$	Λ	Λ_0	Λ_1	Λ_c	M	M_0	M_1	M_c	U_n	MU_n	T_cU_n	M_cU_n	T_1	M_1	T_c	M_c	$[\Omega]$	Λ	Λ_0	Λ_1	$[\Omega]$	Λ_c
V	Ω	$[\Omega]$	$[\Omega]$	$[\Omega]$	M	M	M	M	$[\Omega]$	$[\Omega]$	$[M]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	M	M	M	M	V	V	V	V	$[\Omega]$	M	$[\Omega]$	M	$[\Omega]$	$[M]$	$[\Omega]$	$[M]$	$[\Omega]$	V	V	V	$[\Omega]$	V
V_0	Ω	T_0	$[\Omega]$	$[T_0]$	M	M_0	M	M_0	$[\Omega]$	$[T_0]$	$[M_0]$	$[\Omega]$	$[T_0]$	$[\Omega]$	$[\Omega]$	$[T_0]$	M	M_0	M	M_0	V	V_0	V	V_0	T_0	M_0	$[T_0]$	M_0	$[\Omega]$	$[M]$	$[T_0]$	$[M_0]$	$[\Omega]$	V	V_0	V	$[\Omega]$	V_0
V_1	Ω	$[\Omega]$	T_1	$[T_1]$	M	M	M_1	M_1	$[\Omega]$	$[T_1]$	MW_2	$[\Omega]$	$[\Omega]$	$[T_1]$	$[\Omega]$	$[T_1]$	M	M	M_1	M_1	V	V	V_1	V_1	$[\Omega]$	M	$[T_1]$	M_1	W_n	MW_n	$[W_n]$	MW_n	$[\Omega]$	V	V	V_1	$[\Omega]$	V_1
V_c	Ω	T_0	T_1	T_c	M	M_0	M_1	M_c	$[\Omega]$	$[T_c]$	M_cW_2	$[\Omega]$	$[T_0]$	$[T_1]$	$[\Omega]$	$[T_c]$	M	M_0	M_1	M_c	V	V_0	V_1	V_c	T_0	M_0	T_c	M_c	W_n	MW_n	T_cW_n	M_cW_n	$[\Omega]$	V	V_0	V_1	$[\Omega]$	V_c
U_m	Ω	T_0	Ω	T_0	Ω	T_0	Ω	T_0	$[\Omega]$	$[T_0]$	U_2	Ω	$T_0\dagger$	Ω	$[\Omega]$	$[T_0]$	Ω	U_m	Ω	U_m	Ω	T_0	Ω	T_0	U_p	U_p	U_p	U_p	Ω	Ω	T_0	T_0	Ω	Ω	U_m	Ω	$[\Omega]$	U_m
MU_m	Ω	T_0	$[\Omega]$	$[T_0]$	M	M_0	M	M_0	$[\Omega]$	$[T_0]$	MU_2	Ω	$T_0\dagger$	$[\Omega]$	$[\Omega]$	$[T_0]$	M	MU_m	M	MU_m	M	M_0	M	M_0	U_p	MU_p	$[U_p]$	MU_p	$[\Omega]$	M	$[T_0]$	M_0	Ω	M	MU_m	M	$[\Omega]$	MU_m
T_cU_m	Ω	T_0	T_1	T_c	Ω	T_0	T_1	T_c	$[\Omega]$	$[T_c]$	T_cU_2	Ω	$T_0\dagger$	T_1	$[\Omega]$	$[T_c]$	Ω	U_m	T_1	T_cU_m	Ω	T_0	T_1	T_c	U_p	U_p	T_cU_p	T_cU_p	T_1	T_1	T_c	T_c	Ω	Ω	U_m	T_1	$[\Omega]$	T_cU_m
M_cU_m	Ω	T_0	T_1	T_c	M	M_0	M_1	M_c	$[\Omega]$	$[T_c]$	M_cU_2	Ω	$T_0\dagger$	T_1	$[\Omega]$	$[T_c]$	M	MU_m	M_1	M_cU_m	M	M_0	M_1	M_c	U_p	MU_p	T_cU_p	M_cU_p	T_1	M_1	T_c	M_c	Ω	M	MU_m	M_1	$[\Omega]$	M_cU_m
W_m	Ω	Ω	T_1	T_1	Ω	Ω	T_1	T_1	$[\Omega]$	$[T_1]$	W_2	Ω	Ω	$T_1\dagger$	$[\Omega]$	$[T_1]$	Ω	Ω	T_1	T_1	Ω	Ω	W_m	W_m	Ω	Ω	T_1	T_1	W_p	W_p	W_p	W_p	Ω	Ω	Ω	W_m	$[\Omega]$	W_m
MW_m	Ω	$[\Omega]$	T_1	$[T_1]$	M	M	M_1	M_1	$[\Omega]$	$[T_1]$	MW_2	Ω	$[\Omega]$	$T_1\dagger$	$[\Omega]$	$[T_1]$	M	M	M_1	M_1	M	M	MW_m	MW_m	$[\Omega]$	M	$[T_1]$	M_1	W_p	MW_p	$[W_p]$	MW_p	Ω	M	M	MW_m	$[\Omega]$	MW_m
T_cW_m	Ω	T_0	T_1	T_c	Ω	T_0	T_1	T_c	$[\Omega]$	$[T_c]$	T_cW_2	Ω	T_0	$T_1\dagger$	$[\Omega]$	$[T_c]$	Ω	T_0	T_1	T_c	Ω	T_0	W_m	T_cW_m	T_0	T_0	T_c	T_c	W_p	W_p	T_cW_p	T_cW_p	Ω	Ω	T_0	W_m	$[\Omega]$	T_cW_m
M_cW_m	Ω	T_0	T_1	T_c	M	M_0	M_1	M_c	$[\Omega]$	$[T_c]$	M_cW_2	Ω	T_0	$T_1\dagger$	$[\Omega]$	$[T_c]$	M	M_0	M_1	M_c	M	M_0	MW_m	M_cW_m	T_0	M_0	T_c	M_c	W_p	MW_p	T_cW_p	M_cW_p	Ω	M	M_0	MW_m	$[\Omega]$	M_cW_m
$\Omega(1)$	Ω	Ω	Ω	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	L	L	L	$[L]$	$[L]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$\Omega(1)$	$\Omega(1)$	$\Omega(1)$	$\Omega(1)$	$\Omega(1)$	$\Omega(1)$
I	Ω	$[\Omega]$	$[\Omega]$	$[\Omega]$	M	M	M	M	$[\Omega]$	$[\Omega]$	$[M]$	L	$[L]$	$[L]$	$[L]$	$[L]$	Λ	Λ	Λ	Λ	V	V	V	V	$[\Omega]$	$[M]$	$[\Omega]$	$[M]$	$[\Omega]$	$[M]$	$[\Omega]$	$[M]$	$\Omega(1)$	I	I	I	$\Omega(1)$	I
I_0	Ω	T_0	$[\Omega]$	$[T_0]$	M	M_0	M	M_0	$[\Omega]$	$[T_0]$	$[MU_2]$	L	L_0	$[L]$	$[L]$	$[L_0]$	Λ	Λ_0	Λ	Λ_0	V	V_0	V	V_0	U_n	MU_n	$[U_n]$	MU_n	$[\Omega]$	$[M]$	$[T_0]$	$[M_0]$	$\Omega(1)$	I	I_0	I	$[\Omega(1)]$	I_0
I_1	Ω	$[\Omega]$	T_1	$[T_1]$	M	M	M_1	M_1	$[\Omega]$	$[T_1]$	$[MW_2]$	L	$[L]$	L_1	$[L]$	$[L_1]$	Λ	Λ	Λ_1	Λ_1	V	V	V_1	V_1	$[\Omega]$	$[M]$	$[T_1]$	$[M_1]$	W_n	MW_n	$[W_n]$	MW_n	$\Omega(1)$	I	I	I_1	$[\Omega(1)]$	I_1
I^*	Ω	Ω	Ω	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	S	S	$[S]$	L	L	L	LS	LS	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$[\Omega]$	$\Omega(1)$	$\Omega(1)$	$\Omega(1)$	$\Omega(1)$	I^*	I^*
I_c	Ω	T_0	T_1	T_c	M	M_0	M_1	M_c	S	S_c	SM	L	L_0	L_1	LS	L_c	Λ	Λ_0	Λ_1	Λ_c	V	V_0	V_1	V_c	U_n	MU_n	T_cU_n	M_cU_n	W_n	MW_n	T_cW_n	M_cW_n	$\Omega(1)$	I	I_0	I_1	I^*	I_c

TABLE 3.3 – Table de composition de clones.

Par exemple, $\wedge$ est 1-séparatrice mais pas 0-séparatrice. En effet,

$$\wedge^{-1}(1) = \{(1,1)\} \qquad \text{et}$$
$$\wedge^{-1}(0) = \{(0,0),(0,1),(1,0)\}.$$

De manière duale, $\vee$ est 0-séparatrice mais pas 1-séparatrice. Pour tout $n \geq 3$, la fonction $f : \mathbb{B}^n \to \mathbb{B}$ définie par la règle $f(\mathbf{x}) = 1 \Leftrightarrow \mathrm{w}(\mathbf{x}) = n-1$ est 1-séparatrice de rang $n-1$ mais n'est pas 1-séparatrice de rang n. Par exemple, pour $n = 4$,

$$\begin{aligned} f^{-1}(1) = \{&(0,1,1,1),\\ &(1,0,1,1),\\ &(1,1,0,1),\\ &(1,1,1,0)\}. \end{aligned}$$

— Pour $m \geq 2$, les clones des fonctions 1- et 0-séparatrices de rang m sont respectivement notées U_m et W_m. Les clones de toutes les fonctions 1- et 0-séparatrices sont respectivement notées U_∞ et W_∞. Pour $m = 2, \ldots, \infty$, $T_\mathrm{c}U_m := T_\mathrm{c} \cap U_m$, $T_\mathrm{c}W_m := T_\mathrm{c} \cap W_m$, $MU_m := M \cap U_m$, $MW_m := M \cap W_m$, $M_\mathrm{c}U_m := M_\mathrm{c} \cap U_m$, $M_\mathrm{c}W_m := M_\mathrm{c} \cap W_m$.
— Le clone de toutes les conjonctions et constantes est noté Λ, c'est-à-dire que

$$\Lambda := \{f \in \Omega \mid f \cong \wedge_n \text{ pour un certain } n \geq 2\} \cup \{e_i^{(n)} \mid 1 \leq i \leq n\} \cup \{\mathbf{0}, \mathbf{1}\},$$

et $\Lambda_x := \Lambda \cap T_x$, pour $x \in \{0, 1, \mathrm{c}\}$.
— Le clone de toutes les conjonctions et constantes est noté V, c'est-à-dire que

$$V := \{f \in \Omega \mid f \cong \vee_n \text{ pour un certain } n \geq 2\} \cup \{e_i^{(n)} \mid 1 \leq i \leq n\} \cup \{\mathbf{0}, \mathbf{1}\},$$

et $V_x := V \cap T_x$, pour $x \in \{0, 1, \mathrm{c}\}$.
— Le clone de toutes les projections, projections niées, et constantes est noté $\Omega(1)$, le clone de toutes les projections et projections niées est noté I^*, le clone de toutes les projections et constantes est noté I, et $I_x := I \cap T_x$, pour $x \in \{0, 1, \mathrm{c}\}$.

Soit $\mathcal{F}$ un ensemble de fonctions Booléennes. Le clone *généré par* $\mathcal{F}$, noté $\mathcal{C}(\mathcal{F})$, est le plus petit clone qui contient $\mathcal{F}$, c'est-à-dire que

$$\mathcal{C}(\mathcal{F}) \quad = \quad \bigcap_{\mathcal{C} \text{ un clone, } \mathcal{F} \subseteq \mathcal{C}} \mathcal{C}.$$

Lorsque $\mathcal{F} = \{f\}$, c'est-à-dire qu'il est réduit à une seule fonction f, nous écrivons tout simplement $\mathcal{C}(f)$ et nous disons que f *est un générateur de* $\mathcal{C}(f)$.

Exemple 9. Le clone SM des fonctions monotones et auto-duales est généré, pour un $k \geq 1$ quelconque, par la *fonction majorité* $(2k+1)$-aire $\mu_{2k+1} : \mathbb{B}^{2k+1} \to \mathbb{B}$ définie par la règle

$$\mu_{2k+1}(\mathbf{x}) = 1 \;\Leftrightarrow\; \mathrm{w}(\mathbf{x}) \geq k+1.$$

Notons que $\mu_3 = \mu$; voir la Table 3.1. Les clones $M_\mathrm{c}U_\infty$ et $M_\mathrm{c}W_\infty$ sont générés respectivement par les fonctions ternaires u et w, qui sont définies par

$$u(a_1, a_2, a_3) := (a_1 \vee a_2) \wedge a_3,$$
$$w(a_1, a_2, a_3) := (a_1 \wedge a_2) \vee a_3,$$

pour tous $a_1, a_2, a_3 \in \mathbb{B}$. ∎

Dans le chapitre 6, nous examinons des ensembles de connecteurs que nous déterminons par rapport aux clones que leurs interprétations génèrent ; par exemple, le clone SM et son générateur d'arité minimale μ. Pour cette raison, le résultat suivant d'Émile Post est particulièrement intéressant.

Théorème 10 ([Pos41])**.** Chaque clone Booléen est généré par un ensemble fini de fonctions.

En revanche, tout clone n'est pas généré par une *unique* fonction. Certains clones, tels que le clone M des fonctions monotones, nécessitent l'ajout des fonctions constantes en plus d'une fonction non constante, telle que

$$f(x_1, x_2, x_3, x_4) := (x_1 \wedge x_2) \vee (x_3 \wedge x_4).$$

Par identification de variables, nous pouvons récupérer la conjonction et la disjonction :

$$f(x_1, x_2, x_1, x_2) = x_1 \wedge x_2;$$
$$f(x_1, x_1, x_2, x_2) = x_1 \vee x_2.$$

Par contre, par seule identification, il est impossible de récupérer les fonctions constantes, qui doivent donc être ajoutées à f, pour générer M tout entier.

Définition 11 (Fonctions Sheffer et quasi-Sheffer)**.** Une fonction f est *Sheffer*, respectivement *quasi-Sheffer*, si $\Omega = \mathcal{C}(f)$, respectivement si $\Omega = \mathcal{C}(f) \circ \Omega(1)$. Un clone est *précomplet* s'il contient au moins une fonction quasi-Sheffer.

L'appellation « fonction Sheffer » provient du fait que la fonction $\uparrow$, appelée *Sheffer stroke* en anglais, vérifie $\mathcal{C}(\uparrow) = \Omega$. Cette appellation a parfois été utilisée pour indiquer en effet les fonctions qui, par *superposition* (dans ce contexte, pris dans le sens de la composition d'opérations comme nous en avons donné la définition plus haut), engendrent toutes les fonctions d'un ensemble donné ; voir, par exemple, [Mar52, F$^+$62, Sto88].

Exemple 12. Toute fonction Sheffer est également quasi-Sheffer, mais l'inclusion réciproque est fausse. En effet, la fonction $\uparrow$ est Sheffer et donc quasi-Sheffer, tandis que la fonction majorité μ est quasi-Sheffer mais pas Sheffer. En effet, $\mathcal{C}(\mu) = SM \neq \Omega$. En particulier, SM est précomplet. ■

Dans la figure 3.6, nous avons représenté la séparation entre les clones précomplets et les clones qui ne le sont pas ; voir, également, la proposition 105.

3.4 Modèles de calcul

Dans cette section, nous introduisons quelques modèles de calculs qui nous permettrons d'étudier la complexité de certains problèmes de manière formelle. En particulier, nous parlons de machines de Turing, de termes et de circuits. Les termes nous seront utiles dans tout cet ouvrage ; ils nous permettent de formaliser la notion de représentation de fonctions. Les circuits remplissent également ce rôle de représentation, avec des différences que nous précisons, telle que le partage de sous-circuits. Les machines de Turing nous permettent de définir précisément les classes de complexités classiques dans lesquelles nous placerons nos problèmes, dans le chapitre 5.

Pour plus d'information sur les modèles de calculs, nous nous référons aux ouvrages [Pap03, Sav98] ainsi qu'aux ouvrages plus récents [AB09, Per14].

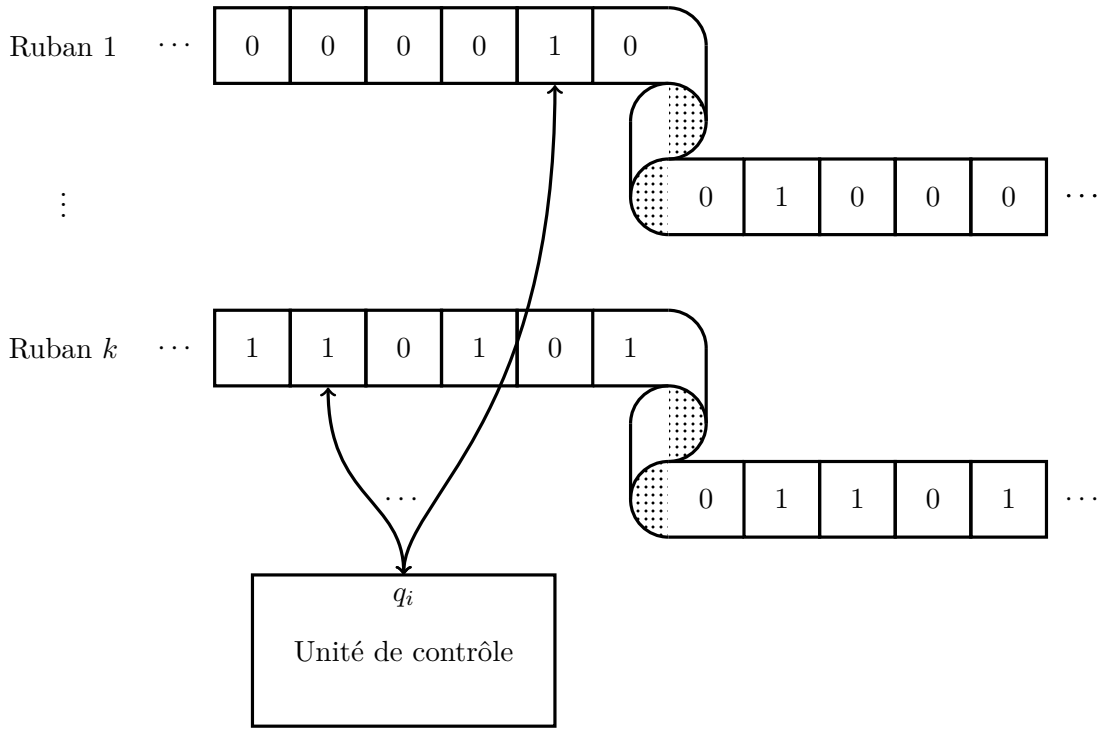


FIGURE 3.7 – Représentation d'une machine de Turing.

3.4.1 Machine de Turing

La *machine de Turing* (MT) est un modèle de calcul discret classique qui a été introduit par Alan Turing en 1936 [Tur36, Tur38]. Il en existe différentes descriptions, qui sont équivalentes. Nous adoptons ici le formalisme et les notations de [Per14]. Auparavant, David Hilbert et Wilhelm Ackermann avaient étudié dès les années 20 le problème consistant à trouver un « algorithme » (avant qu'il n'existe une définition formelle de la notion) qui déciderait si une proposition en logique du premier ordre est une conséquence des axiomes de l'arithmétique ([HA28]). Une solution, tant sur le plan conceptuel (la définition d'un algorithme) que sur le plan calculatoire a été proposée indépendamment par Church ([Chu36]) puis par Turing ([Tur36]) à quelques mois d'intervalle.

Une machine de Turing peut être vue comme un ensemble de k rubans de tailles infinies, découpés en cellules, et manipulés par une unité de contrôle, comme sur la figure 3.7. À chaque étape de temps et pour chaque ruban, l'unité de contrôle reçoit une entrée de ce ruban et lui fournit une sortie. La tête de lecture, symbolisée par une flèche sur la figure, pointe vers la cellule à lire et sur laquelle écrire.

Définition 13 (Machine de Turing, [Per14])**.** Une *machine de Turing* M à $k \geq 2$ rubans est un octuplet

$$M = (\Sigma, \Gamma, B, Q, q_0, q_a, q_r, \delta)$$

où

— Σ est un ensemble fini non vide appelé *alphabet d'entrée*. Ce sont les symboles utilisés pour écrire un mot $m \in \Sigma^*$ en entrée.

— Γ est un ensemble fini appelé *alphabet de travail*, tel que $\Sigma \subset \Gamma$. Ce sont les symboles utilisés par la machine au cours du calcul.

— $B \in \Gamma \backslash \Sigma$ est un symbole spécial blanc qui représente une case vide.
— Q est un ensemble fini appelé *ensemble des états*. Ce sont les états que peuvent prendre l'unité de contrôle et ses têtes de lecture.
— $q_0 \in Q$ est l'*état initial* ; $q_a \in Q$ est l'*état final d'acceptation* et $q_r \in Q$ est l'*état final de rejet*. Les états q_a et q_r sont aussi appelés *états terminaux*.
— δ est la *fonction de transition* qui décrit le comportement de la machine :

$$\begin{aligned} \delta \colon (Q \backslash \{q_a, q_r\}) \times \Gamma^k &\to Q \times \Gamma^k \times \{\leftarrow, \downarrow, \rightarrow\}^k \\ (q, a_1, \ldots, a_k) &\mapsto (r, b_2, \ldots, b_k, d_1, \ldots, d_k). \end{aligned}$$

Si les têtes sont dans l'état q et que la i-ème tête de lecture lit le symbole a_i, alors le nouvel état dans lequel se trouve les têtes est r, la tête i écrit le symbole b_i à la place de a_i, et elle se déplace dans la direction d_i : à gauche, en restant sur place, ou à droite.

Une *configuration* d'une machine est la description du contenu des rubans, de la position des têtes, et de leur état. Une *configuration finale* est une configuration pour laquelle l'état est terminal. Le *calcul* d'une machine de Turing est une suite potentiellement infinie de configurations régie par la fonction de transition δ. La machine de Turing ne s'arrête pas nécessairement. On dit que la machine de Turing M *s'arrête* sur l'entrée x, ou que $M(x)$ s'arrête, si le calcul de M sur x atteint une configuration finale. Dans ce cas, si l'état de la configuration finale est q_a, alors on dit que M *accepte* x ; sinon, M *rejette* x. À la fin du calcul, c'est-à-dire après l'arrêt de $M(x)$, le contenu des rubans constitue le *résultat du calcul de M sur x*, également noté $M(x)$. Par convention, un ruban peut faire office de ruban de sortie, et alors on lira le résultat du calcul sur celui-ci.

Définition 14. Soit M une machine de Turing. Le *langage accepté* par M est l'ensemble des mots $x \in \Sigma^*$ acceptés par M. La *fonction calculée* par M est la fonction partielle $f_M : \Sigma^* \to \Gamma^*$ définie par $f_M(x) = M(x)$ si $M(x)$ s'arrête.

Les machines de Turing que nous avons considérées ci-dessus sont *déterministes* ([Hoe16]) : à aucun moment, la transition δ ne dépend d'un *choix* extérieur à la machine ; le passage d'une configuration à une autre ne dépend que de l'état de l'unité de contrôle et de ses têtes et des cases lues par ces têtes. Dans le cas d'une machine *non déterministe*, la transition δ est différente, en ce sens qu'à chaque étape plusieurs transitions sont possibles ; les configurations ne forment alors plus une suite mais un *arbre* de configurations, dont les noeuds ont un ou plusieurs fils suivant leurs successeurs.

Définition 15 (Machine de Turing non déterministe)**.** Une *machine de Turing*

$$N = (\Sigma, \Gamma, B, Q, q_0, q_a, q_r, \delta)$$

est dite *non déterministe* si la transition δ n'est plus une fonction mais une relation :

$$\delta \subseteq ((Q \backslash \{q_a, q_r\}) \times \Gamma^k) \times (Q \times \Gamma^k \times \{\leftarrow, \downarrow, \rightarrow\}^k).$$

Tout élément

$$((q, a_1, \ldots, a_k), (r, b_2, \ldots, b_k, d_1, \ldots, d_k)) \in \delta$$

est ainsi un couple : si la machine se trouve dans l'état q et lit $(q, a_1, \ldots, a_k)$, alors elle peut aller dans l'état r, écrire $(b_2, \ldots, b_k)$ sur ses rubans, et déplacer ses têtes dans les directions $(d_1, \ldots, d_k)$.

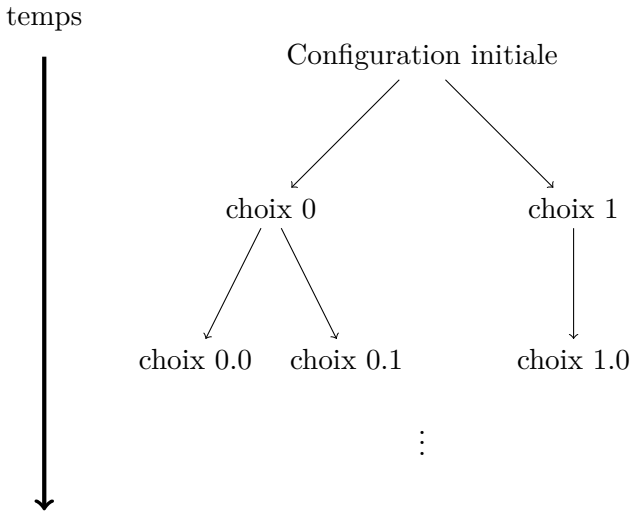


FIGURE 3.8 – Arbre de configurations d'une machine de Turing non déterministe.

Une *exécution* d'une machine non déterministe N est une suite de configurations compatible avec la relation de transition δ allant d'une configuration initiale à une configuration finale. On l'appelle également un *chemin* si l'on considère l'arbre des calculs possibles et si l'on considère les chemins qui vont de la racine à la feuille. Le temps (respectivement l'espace) de calcul de N est le temps (respectivement l'expace) maximal d'une de ses exécutions.

Définition 16. Le langage reconnu par une machine de Turing non déterministe N sur l'alphabet Σ est l'ensemble des mots $x \in \Sigma^*$ tels qu'il existe un chemin acceptant dans le calcul $N(x)$.

Dans la section 3.5, nous donnons la définition des classes de complexités usuelles P, NP, PSPACE, etc. avec pour modèle de calcul les machines de Turing.

3.4.2 Systèmes de réécriture et spécifications équationnelles

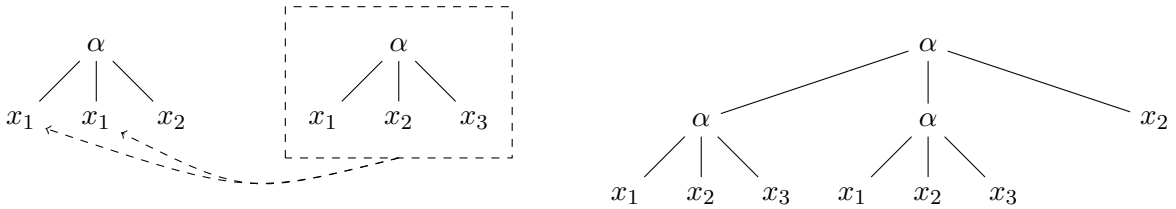
Nous rappelons brièvement les notions de termes, algèbres, et opérations de termes de la théorie de l'Algèbre Universelle [SB81]. Soit F un ensemble de *symboles d'opérations* ou *connecteurs*, et soit $\tau\colon F \to \mathbf{N}$ une opération, appelée un *type (algébrique de similitude)* qui associe à chaque symbole d'opération son *arité*. Une *algèbre* est une paire $\mathbf{A} = (A, F^{\mathbf{A}})$, dans laquelle A est un ensemble non-vide, appelé le *domaine* ou l'*univers* de $\mathbf{A}$, et $F^{\mathbf{A}} = (f^{\mathbf{A}}\colon f \in F)$ est une famille indexée d'opérations sur A, avec chaque $f^{\mathbf{A}}$ d'arité $\tau(f^{\mathbf{A}})$.

Définition 17 (Termes)**.** Soit $\tau\colon F \to \mathbb{N}$ un type, et soit X un ensemble disjoint de F. Les éléments de X sont appelés *variables*. Les *termes de type τ sur X* sont définis de manière inductive de la manière suivante.

— Chaque variable $x \in X$ est un terme.
— Si $c \in F$ et $\tau(c) = 0$, alors c est un terme.
— Si $f \in F$, $\tau(f) > 0$ et $t_1, \ldots, t_{\tau(f)}$ sont des termes, alors

$$f(t_1, \ldots, t_{\tau(f)})$$

est un terme.



Terme t. Terme t_1. Terme $t\{t_1/x_1\}$.

FIGURE 3.9 – Exemple de substitution.

L'ensemble de tous les termes de type τ sur X est noté $T_\tau(X)$.

Sauf mention contraire explicite, nous considérons des termes sur l'ensemble standard des variables, c'est-à-dire $X = \{x_i \colon i \in \mathbb{N}\}$. Un terme $t \in T_\tau(X)$ est n-aire si les variables qui apparaissent dans t sont prises parmi $x_1, \ldots, x_n$.

Nous utiliserons parfois la notation infixe pour certains symboles de fonctions binaires. Ainsi nous écrirons $t_1 \alpha t_2$ au lieu de $\alpha(t_1, t_2)$ si α est un symbole de fonction.

Définition 18 (Opérations de termes)**.** Soit $\mathbf{A}$ une algèbre de type τ. Chaque terme n-aire $\tau \in T_\tau(X)$ induit une opération n-aire $t^{\mathbf{A}}$ sur A de la manière suivante.

— Si $t = x_i \in X$, alors $t^{\mathbf{A}}$ est la i-ème projection n-aire $e_i^{(n)}$.

— Si $t = c \in F$ avec $\tau(c) = 0$, alors $t^{\mathbf{A}}$ est une fonction constante identiquement égale à $c^{\mathbf{A}}$.

— Si $t = f(t_1, \ldots, t_{\tau(f)})$, alors $t^{\mathbf{A}} = f^{\mathbf{A}}(t_1^{\mathbf{A}}, \ldots, t_{\tau(f)}^{\mathbf{A}})$.

L'opération $t^{\mathbf{A}}$ est appelée *l'opération de terme* induite par t sur $\mathbf{A}$.

On dit aussi que $t^{\mathbf{A}}$ est l'*interprétation* de t dans $\mathbf{A}$, ou encore que le terme t *représente* la fonction $t^{\mathbf{A}}$.

Notons que si un terme est d'arité n, alors il est aussi d'arité n' pour tout $n' \geq n$. Par conséquent, l'arité n'est pas une propriété inhérente à un terme et devrait être précisée à chaque fois que nous considérons des opérations sur des termes. Toutefois, dans la plupart des cas, l'arité des opérations peut être clairement déduite du contexte.

Un terme t est *linéaire* si aucune variable n'apparaît plus d'une fois dans t. Tout sous-mot d'un terme t qui est lui-même un terme est un *sous-terme* de t. Étant donné un terme t contenant les variables $x_1, \ldots, x_n$ et des termes $t_1, \ldots, t_n$, le terme

$$t\{t_1/x_i \ldots t_n/x_n\}$$

est obtenu à partir de t en remplaçant chaque instance de x_i dans t par t_i.

Exemple 19. Si $t = \alpha(x_1, x_1, x_2)$ et $t_1 = \alpha(x_1, x_2, x_3)$, alors

$$t\{t_1/x_1\} \;=\; \alpha(\alpha(x_1, x_2, x_3), \alpha(x_1, x_2, x_3), x_2).$$

Voir la Figure 3.9. ■

Dans cet ouvrage, nous considérons un type algébrique particulier τ et une algèbre particulière de type τ. En effet, l'ensemble des symboles d'opérations est l'ensemble de toutes les fonctions Booléennes, c'est-à-dire que $F = \Omega$. Nous prenons

$$\begin{aligned} \tau \colon \Omega &\to \mathbb{N} \\ f &\mapsto \mathrm{ar}(f). \end{aligned}$$

Soit $\mathbf{B} = (\mathbb{B}, \Omega^B)$ l'algèbre de type τ telle que pour tout $f \in \Omega$, $f^{\mathbf{B}} = f$. Ainsi, nous pouvons construire des termes en utilisant des fonctions Booléennes comme symboles d'opération, qui seront interprétés dans $\mathbf{B}$ de manière naturelle comme des fonctions Booléennes.

Nous utiliserons les lettres $s, s', t, t', \ldots$ pour désigner des termes dans $T_\tau(X)$. Les variables et les termes de la forme $\neg(x_i)$ pour une certaine variable x_i sont appelés des *littéraux*. Étant donné un terme t et un entier $k > 0$, soit t^k la chaîne de caractères définie inductivement par $t^1 := t$ et $t^{n+1} := tt^n$.

Deux termes n-aires $s, t \in T_\tau(X)$ sont *équivalents*, ce que l'on note $s \equiv t$, si $s^{\mathbf{B}} = t^{\mathbf{B}}$. Étant donné un terme $t \in T_\tau(X)$, nous désignons souvent l'opération de termes $t^{\mathbf{B}}$ par $[t]$, lorsqu'il n'y a pas d'ambiguïté. Pour un ensemble $S \subseteq T_\tau(X)$, l'*interprétation* de S est définie par $[S] := \{[t] \mid t \in S\}$.

Exemple 20. Considérons les termes binaires $\mathrm{m}(x_1, x_2, 1)$ et $x_1 \vee x_2$ dans $T_\tau(X)$. Alors,

$$[\mathrm{m}(x_1, x_2, 1)] = \mathrm{m}(e_1^{(2)}, e_2^{(2)}, 1) = \vee(e_1^{(2)}, e_2^{(2)}) = [x_1 \vee x_2].$$

Autrement dit, $\mathrm{m}(x_1, x_2, 1) \equiv x_1 \vee x_2$ et les deux termes représentent la même fonction. ∎

Nous empruntons les notations de [Klo92] et de [BN99] pour introduire la notion de spécification équationnelle. Une *spécification équationnelle*, parfois appelée *système équationnel* dans ce document, est une paire (Σ, E) constituée d'un *alphabet* Σ, appelé aussi *signature*, et d'un ensemble d'équations E. L'alphabet Σ est défini comme

$$\Sigma := X \cup F,$$

c'est-à-dire qu'il consiste en un nombre infini mais dénombrable de variables $x_1, x_2, \ldots$, et d'un nombre non nul de *symboles de fonctions* ou *symboles d'opérations* ; Dans le chapitre 5, qui concerne la manipulation de formules médianes, cet ensemble contient m et les constantes de L.

Nous rappelons que l'ensemble des *termes*, ou *expressions*, sur Σ, est noté $T_\tau(\Sigma)$. Une *équation* est une expression de la forme $s = t$ dans laquelle $s, t \in T_\tau(\Sigma)$. Nous désignons l'ensemble de toutes les équations par E.

Un *contexte* est un terme qui contient une occurrence d'un symbole spécial $\diamond$ qui signifie un espace vide. Un contexte est généralement désigné par $C[\diamond]$. Si $t \in T_\tau(\Sigma)$ et que t est substitué dans $\diamond$, alors le terme résultant est $C[t] \in T_\tau(\Sigma)$.

Une *substitution* est une application σ de $T_\tau(\Sigma)$ vers $T_\tau(\Sigma)$ qui satisfait

$$\sigma(F(t_1, \ldots, t_n)) = F(\sigma(t_1), \ldots, \sigma(t_n))$$

pour tout symbole de fonction n-aire pour $n \geq 0$.

La règle de substitution, associée aux règles que nous rappelons dans la table 3.4, permet d'obtenir ce que l'on nomme les *équations dérivables*, c'est-à-dire les équations que l'on peut obtenir en appliquant une combinaison finie de ces règles. Lorsqu'une équation $s = t$ est dérivable à partir d'un système E, nous écrivons $(\Sigma, E) \vdash s = t$ ou encore $\Sigma \vdash_E s = t$.

Définition 21. Un *Système de Réécriture de Termes* (*TRS*, de l'anglais *Term Rewriting System*) est une spécification équationnelle dont toutes les équations ont été orientées. Une paire (l, r) de termes de $T_\tau(\Sigma)$, que l'on note $l \longrightarrow r$, est une *règle de réduction* si r n'est pas une variable et si toutes les variables dans r sont contenues dans l.

Tout système de réécriture de termes

$$R = \{l_1 \longrightarrow r_1, \ldots, l_n \longrightarrow r_n\}$$

engendre des relations de réécriture sur les termes :

$$(\Sigma, E) \vdash t = t \qquad \text{si } t \in T_\tau(\Sigma)$$

$$(\Sigma, E) \vdash s = t \qquad \text{si } s = t \in E$$

$$\frac{(\Sigma, E) \vdash s = t}{(\Sigma, E) \vdash t = s}$$

$$\frac{(\Sigma, E) \vdash t_1 = t_2, (\Sigma, E) \vdash t_2 = t_3}{(\Sigma, E) \vdash t_1 = t_3}$$

$$\frac{(\Sigma, E) \vdash s = t}{(\Sigma, E) \vdash \sigma(s) = \sigma(t)} \qquad \text{pour toute substitution } \sigma$$

$$\frac{(\Sigma, E) \vdash s_1 = t_1, \ldots, (\Sigma, E) \vdash s_n = t_n}{(\Sigma, E) \vdash F(s_1, \ldots, s_n) = F(t_1, \ldots, t_n)} \qquad \text{pour toute fonction } n\text{-aire } F \in \Sigma$$

TABLE 3.4 – Système d'inférence équationnel.

— on a $s \longrightarrow_R t$ si $s \longrightarrow t$ grâce à l'une des règles $l_i \longrightarrow r_i$;
— la relation $\longrightarrow_R^*$ est obtenue par clôture réflexive et transitive de $\longrightarrow_R$;
— la relation $\twoheadrightarrow$ est obtenue à partir de $\longrightarrow_R^*$ par clôture par substitution et contexte : si $l \longrightarrow_R^* r$, alors $\forall \sigma, \sigma(l) \twoheadrightarrow \sigma(r)$, et $\forall C[\diamond], C[l] \twoheadrightarrow \sigma C[r]$. Par abus de notation et lorsque l'ensemble R est clair à partir du contexte, nous désignons simplement par $\longrightarrow$ cette relation, et par $\longrightarrow^*$ la clôture réflexive et transitive de $\longrightarrow_R$.

Exemple 22. Les systèmes de réécritures permettent, par exemple, de formaliser le calcul d'opérations arithmétiques sur les entiers naturels, en s'inspirant de la construction axiomatique que Peano en a faite (voir [Pea89, Pla93]). Les axiomes qui nous intéressent ici sont les axiomes suivants, décrits de manière informelle :
— il existe un entier naturel, nommé 0 ;
— tout entier naturel a un entier naturel qui est son "successeur" (existence d'une fonction successeur, $\mathtt{S}$) ;
— deux entiers naturels différents ont des successeurs différents ($\mathtt{S}$ est injective) ;
— 0 n'est le successeur d'aucun entier naturel ;
— tout sous-ensemble d'entiers naturels qui contient 0 et qui est fermé par $\mathtt{S}$ est l'ensemble des entiers naturels (axiome de l'induction).

Nous avons noté cet ensemble bien connu $\mathbb{N}$:

$$\mathbb{N} = \{0, \mathtt{S}(0), \mathtt{S}(\mathtt{S}(0)), \ldots\}.$$

Une graphie alternative pour les éléments de $\mathbb{N}$ est $1 := \mathtt{S}(0)$, $2 := \mathtt{S}(\mathtt{S}(0))$, etc. Considérons à présent le système Add, constitué de deux règles Add_1 et Add_2, qui permet de décrire l'addition des entiers naturels, notée $+$:

$$Add = \{x + 0 \longrightarrow x, x + \mathtt{S}(y) \longrightarrow \mathtt{S}(x + y)\}.$$

Ainsi, par exemple, l'addition $2 + 3$ se réécrit ainsi :

$$\begin{aligned}
\mathtt{S}(\mathtt{S}(0)) + \mathtt{S}(\mathtt{S}(\mathtt{S}(0))) &\longrightarrow \mathtt{S}(\mathtt{S}(\mathtt{S}(0)) + \mathtt{S}(\mathtt{S}(0))) && (Add_2)\\
&\longrightarrow \mathtt{S}(\mathtt{S}(\mathtt{S}(\mathtt{S}(0)) + \mathtt{S}(0)) && (Add_2)\\
&\longrightarrow \mathtt{S}(\mathtt{S}(\mathtt{S}(\mathtt{S}(\mathtt{S}(0))) + 0) && (Add_2)\\
&\longrightarrow \mathtt{S}(\mathtt{S}(\mathtt{S}(\mathtt{S}(\mathtt{S}(0))))). && (Add_1)
\end{aligned}$$

Puisque $5 := \mathtt{S}(\mathtt{S}(\mathtt{S}(\mathtt{S}(\mathtt{S}(0)))))$, nous obtenons le résultat escompté. ∎

L'étude des systèmes de réécriture de manière abstraite (*ARS, Abstract Rewriting Systems*), c'est-à-dire d'un simple ensemble A muni d'une relation binaire $\longrightarrow$, permet de mettre en évidence des problèmes généraux liés à la réécriture.

— Un élément $x \in A$ est-il *réductible* ? Existe-t-il un élément $y \in A$ tel que $x \longrightarrow y$? Un tel élément est alors appelé une *forme normale*, qui peut être unique. Dans l'exemple 22, nous avons donné un exemple de forme normale : l'entier $\mathtt{S}(\mathtt{S}(\mathtt{S}(\mathtt{S}(\mathtt{S}(0)))))$.

— Étant donné x et y, sont-ils équivalents selon la plus petite relation d'équivalence tirée de la relation $\longrightarrow$? Ce problème est appelé, en anglais, le *word problem* (voir, par exemple, [KB70]). En général, ce problème est indécidable. Dans l'exemple 22, les mots $\mathtt{S}(0) + 0$ et $0 + \mathtt{S}(0)$ ($1 + 0$ et $0 + 1$) sont équivalents, car

$$\begin{aligned} \mathtt{S}(0) + 0 &\longrightarrow \mathtt{S}(0) && (Add_1) \\ &\longrightarrow \mathtt{S}(0 + 0) && (Add_1^{-1}) \\ &\longrightarrow 0 + \mathtt{S}(0). && (Add_2^{-1}) \end{aligned}$$

— L'ARS considéré est-il *confluent* ? La confluence, nom inspiré de la confluence des cours d'eau, signifie, pour un élément $a \in A$, que si pour toute paire d'éléments différents $b, c \in A$ tels que $a \longrightarrow^* b$ et $a \longrightarrow^* c$, alors il existe un élément $d \in A$ tel que $b \longrightarrow^* d$ et $c \longrightarrow^* d$. Si tous les éléments de A sont confluents, alors A est confluent ; on dit aussi qu'il est Church-Rosser ([CR36]). Cette propriété exprime le fait qu'un élément peut être réécrit de différentes manières pour obtenir le même résultat. C'est le cas de la somme d'entiers naturels, qui exprime d'ailleurs son associativité :

$$\begin{aligned} \mathtt{S}(0) + \mathtt{S}(0) + \mathtt{S}(0) &\longrightarrow \mathtt{S}(\mathtt{S}(0) + 0) + \mathtt{S}(0) \\ &\longrightarrow \mathtt{S}(\mathtt{S}(0)) + \mathtt{S}(0) \\ &\longrightarrow \mathtt{S}(\mathtt{S}(\mathtt{S}(0)) + 0) \\ &\longrightarrow \mathtt{S}(\mathtt{S}(\mathtt{S}(0)), \end{aligned}$$

et

$$\begin{aligned} \mathtt{S}(0) + \mathtt{S}(0) + \mathtt{S}(0) &\longrightarrow \mathtt{S}(0) + \mathtt{S}(\mathtt{S}(0) + 0) \\ &\longrightarrow \mathtt{S}(0) + \mathtt{S}(\mathtt{S}(0)) \\ &\longrightarrow \mathtt{S}(\mathtt{S}(0) + \mathtt{S}(0)) \\ &\longrightarrow \mathtt{S}(\mathtt{S}(\mathtt{S}(0) + 0)) \\ &\longrightarrow \mathtt{S}(\mathtt{S}(\mathtt{S}(0))). \end{aligned}$$

— L'ARS considéré est-il *noethérien* ? Un ARS est noethérien s'il n'existe pas de séquence infinie de réécritures. C'est le cas pour l'exemple 22 ; mais si nous avions ajouté l'axiome de la commutativité de l'addition

$$x + y \longrightarrow y + x,$$

alors nous aurions autorisé des séquences de réécritures telles que

$$\mathtt{S}(0) + 0 \longrightarrow 0 + \mathtt{S}(0) \longrightarrow \mathtt{S}(0) + 0 \longrightarrow 0 + \mathtt{S}(0) \longrightarrow \dots.$$

3.4.3 Circuits

À l'instar des machines de Turing, les *circuits*, ou *circuits logiques* sont un modèle de calcul, toutefois sans mémoire, qui permettent de calculer des fonctions, de résoudre des problèmes, de reconnaître des langages.

Un *graphe* $G = (V, E)$ consiste en un ensemble fini de noeuds V (pour *vertices*) et en un ensemble fini E (pour *edges*) de paires de noeuds $E \subseteq V \times V$ qui sont appelées les arêtes. Une arête e est *incidente* au noeud v si e contient v. Un graphe est *non-dirigé* si pour toute arête $(v_1, v_2) \in E$ on a aussi $(v_2, v_1) \in E$. Dans un graphe *dirigé*, une arête (v_1, v_2) est dirigée de v_1 vers v_2, ce que l'on symbolise par une flèche si l'on veut le dessiner : voir, par exemple, la figure 4.7. Le *degré entrant* (*in-degree*) d'un noeud est le nombre d'arêtes qui portent sur lui ; le *degré sortant* (*out-degree*) d'un noeud est le nombre d'arêtes qui en sortent ; son *degré* est la somme de ses degrés sortants et entrants. Dans un graphe dirigé, un *noeud d'entrée* ou *entrée* a pour degré entrant 0, tandis qu'un *noeud de sortie* ou *sortie* a pour degré sortant 0 ou est conçu comme tel (par exemple dans le cas des automates à états finis où une sortie peut aussi être l'origine d'autres arêtes). Un *chemin*, dans un graphe dirigé ou non, est un tuple d'arêtes $(v_1, \ldots, v_p)$ tel que $(v_i, v_{i+1}) \in E$ pour tout $1 \leq i \leq p-1$. Un chemin $(v_1, \ldots, v_p)$ est un *cycle* si $v_1 = v_p$ et $p \geq 2$. La *longueur* d'un chemin est le nombre d'arêtes qu'il contient : le chemin $(v_1, \ldots, v_p)$ est de taille $p - 1$.

Un *graphe acyclique dirigé* ou *orienté* (*DAG*, *Directed Acyclic Graph*) est un graphe dirigé qui ne contient pas de cycle.

Définition 23. Un *circuit logique* est un DAG dans lequel tous les noeuds, mis à part les noeuds de sortie, portent des étiquettes de fonctions Booléennes, aussi appelées *portes logiques* dans ce contexte. Les noeuds d'entrées portent des étiquettes de variables Booléennes. L'ensemble des étiquettes de portes logiques utilisées dans un tel graphe s'appelle la *base* du circuit.

Souvent, ces portes logiques sont $\vee$, $\wedge$, et $\neg$; notons que la base $\{\vee, \wedge, \neg\}$ est complète vis-à-vis des fonctions Booléennes. La *taille* d'un circuit est le nombre de noeuds qui ne sont pas des entrées qu'il contient. La *profondeur* d'un circuit est la taille du plus grand chemin dirigé d'un noeud d'entrée vers un noeud de sortie.

Dans la figure 3.10, nous avons représenté un circuit qui calcule la fonction $f^{(n)}_{\mathrm{mod}\ 3}$ pour $n = 3$, qui vaut 1 si la somme modulo 3 de ses n variables vaut 0 modulo 3 et 0 sinon.

Sa taille est de 8 et sa profondeur de 4. Les noeuds entrants sont indiqués par des carrés. Le noeud sortant est indiqué par un cercle double, comme il en est l'usage pour les automates cellulaires. Notons le partage possible de sous-circuits, ici au niveau des variables, en contraste avec la représentation de fonctions avec des formules.

3.5 Complexité

Dans cette section, nous rappelons quelques notions de complexité calculatoire (*computational complexity*), qui nous permettrons d'étudier et de comparer la complexité de certains algorithmes et problèmes en rapport avec, par exemple, la simplification de formules médianes, dans le chapitre 5. Certains exemples de problèmes proviennent du *Complexity Zoo* ([AKGR20]).

Nous rappelons la notation $\mathcal{O}$, utilisée pour décrire le comportement asymptotique de fonctions. Soient f et g deux fonctions dont les domaines et co-domaines sont les entiers naturels ou les réels. S'il existe des constantes strictement positives n_0 et A telles que pour tout $n > n_0$,

$$f(n) \leq Ag(n),$$

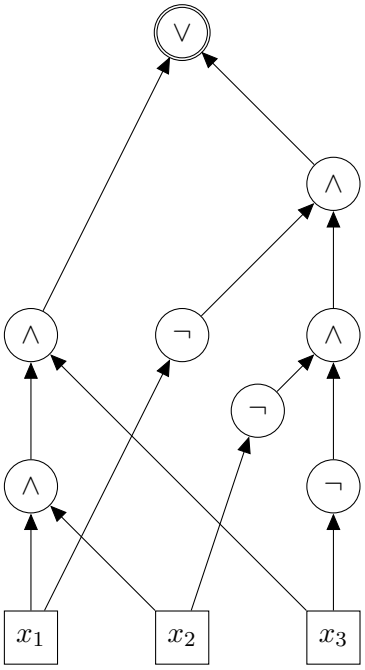


FIGURE 3.10 – Représentation par un DAG d'un circuit logique qui calcule $f^{(3)}_{\text{mod } 3}$.

alors nous écrivons

$$f(n) = \mathcal{O}(g(n))$$

et nous disons que « $f(n)$ est un grand O de $g(n)$ », ou encore que f ne croît pas plus vite, en n, que g.

3.5.1 Classes de complexité

Nous référons à la section 3.4 pour une courte introduction au modèle de calcul des machines de Turing. Par la suite, M désigne une machine de Turing déterministe, et N désigne une machine de Turing non déterministe.

Remarque 24. Un langage peut être considéré de manière équivalente comme un problème de décision. Par exemple, le problème PREMIER qui consiste à déterminer si un entier est premier ou non correspond au langage qui contient tout les nombres premiers, c'est-à-dire l'ensemble

$$\{2, 3, 5, 7, 11, \ldots\}.$$

Définition 25. Si $M(x)$ s'arrête, c'est-à-dire si la machine de Turing M s'arrête sur l'entrée x, alors le *temps de calcul* $t_M(x)$ est le nombre d'étapes effectuées par le calcul $M(x)$. L'*espace* $s_M(x)$ utilisé par le calcul $M(x)$ est le nombre total de cases différentes visitées par les têtes.

Définition 26 (Complexité en temps déterministe)**.** Étant donnée une fonction $t : \mathbb{N} \to \mathbb{N}$, la classe

$$\mathsf{DTIME}(t(n))$$

est l'ensemble des langages reconnus par une machine de Turing M telle qu'il existe une constante positive $k \in \mathbb{N}$ pour laquelle $M(x)$ fonctionne en temps inférieur à $kt(|x|)$ sur toute entrée x,

avec $|x|$ la taille de x. Étant donné un ensemble de fonctions $\mathcal{T}$,

$$\mathsf{DTIME}(\mathcal{T}) := \bigcup_{t \in \mathcal{T}} \mathsf{DTIME}(t(n)).$$

Nous définissons à présent les classes de langages reconnus en temps polynomial ou exponentiel.

Définition 27. La classe P est l'ensemble des langages reconnus en temps polynomial :

$$\mathsf{P} := \mathsf{DTIME}(n^{\mathcal{O}(1)}) = \bigcup_{k \in \mathbb{N}} \mathsf{DTIME}(n^k).$$

Exemple 28. Nous donnons ici quelques exemples de problèmes qui sont dans P.

— PGCD (Plus grand commun diviseur)
Entrée : Deux entiers naturels non nuls p, q ;
Sortie : Le PGCD (Plus Grand Commun Diviseur) de p, q.
L'algorithme d'Euclide, par exemple, résout ce problème en temps quadratique.

— ESTPREMIER (Test de primalité, [AKS04])
Entrée : Un entier naturel non nul n ;
Sortie : Réussite si n est premier, Échec sinon.

— PROGLIN (Programmation linéaire, [Kha79, GL81])
Entrée : Un système de n inéquations linéaires à m variables ;
Sortie : Réussite si le système admet une solution, Échec sinon.
La méthode du Simplexe, qui divise effectivement l'espace des solutions à chaque étape ou chaque inéquation, en est une méthode de résolution efficace.

— MAXCOUP (Couplage maximal dans un graphe, [Edm65])
Entrée : Un graphe G composé de deux ensembles finis, ses *points* et ses *arêtes* ;
Sortie : Un couplage maximal pour G, c'est-à-dire un sous-ensemble d'arêtes qui ne partagent pas de point deux-à-deux (couplage), de cardinalité maximale.

■

Définition 29. La classe EXP est l'ensemble des langages reconnus en temps exponentiel :

$$\mathsf{EXP} := \mathsf{DTIME}(2^{n^{\mathcal{O}(1)}}) = \bigcup_{k \in \mathbb{N}} \mathsf{DTIME}(2^{n^k}).$$

Exemple 30. Nous donnons ici un exemple de problème de EXP.

— $\mathsf{ARRÊT}_k$ (Problème de l'Arrêt majoré ; *Halting problem*, [Tur38, Coo71, Lev73])
Entrée : Une machine de Turing déterministe, un entier k.
Sortie : Réussite si la machine s'arrête en au plus k étapes, pour toute entrée ; Échec sinon.
Le Problème de l'Arrêt est un problème indécidable connu depuis Turing, et sous ce nom depuis au moins Davis [Dav58], mais sa restriction à un nombre fini d'étapes k est décidable, et dans EXP.

■

De manière similaire, la classe PSPACE est l'ensemble des langages reconnus en espace polynomial.

Exemple 31. Nous donnons ici un exemple de problème de PSPACE.

— QBF (Formule Booléenne quantifiée ; *Quantified Boolean Formula*, [SM73])

Entrée : Une formule en logique du premier ordre dont toutes les variables sont quantifiées universellement ou existentiellement.
Sortie : RÉUSSITE s'il existe un modèle, c'est-à-dire une interprétation de variables, dans lequel la formule est vraie ; ÉCHEC sinon.

■

Jusqu'à présent, nous avons considéré des classes de complexité basées sur les machines de Turing déterministes (définition 13). Nous pouvons définir de manière similaire des classes de complexités basées sur les machines de Turing non déterministes (définition 15).

Définition 32 (Complexité en temps non déterministe)**.** Étant donnée une fonction $t : \mathbb{N} \to \mathbb{N}$, la classe

$$\mathsf{NTIME}(t(n))$$

est l'ensemble des langages reconnus par une machine de Turing non déterministe N telle qu'il existe une constante positive $k \in \mathbb{N}$ pour laquelle $N(x)$ fonctionne en temps inférieur à $kt(|x|)$ sur toute entrée x, avec $|x|$ la taille de x, c'est-à-dire que toute branche de son arbre de calcul sur l'entrée x est de taille majorée par $kt(|x|)$. Étant donné un ensemble de fonctions $\mathcal{T}$,

$$\mathsf{NTIME}(\mathcal{T}) := \bigcup_{t \in \mathcal{T}} \mathsf{NTIME}(t(n)).$$

Définition 33. La classe NP (*Nondeterministic Polynomial*) est l'ensemble des langages reconnus en temps polynomial par des machines non déterministes :

$$\mathsf{NP} := \mathsf{NTIME}(n^{\mathcal{O}(1)}) = \bigcup_{k \in \mathbb{N}} \mathsf{NTIME}(n^k).$$

Une caractérisation équivalente de NP est qu'il est l'ensemble des langages dont on peut vérifier l'appartenance en temps polynomial. Formellement, un langage $\mathcal{L}$ est dans NP si et seulement si il existe un polynôme $p(n)$ et un langage $\mathsf{B} \in \mathsf{P}$ tels que

$$x \in \mathsf{A} \quad \text{si et seulement si} \quad \exists x \in \{0,1\}^{p(|x|)}, (x, y) \in \mathsf{B}.$$

Le mot y, de taille polynomiale, encode un chemin acceptant dans l'arbre de calcul de la machine non déterministe qui reconnaît le langage A.

Exemple 34. Nous donnons ici quelques exemples de problèmes de NP.
— SAT (Satisfiabilité, [Coo71, Lev73])
Entrée : Une formule de la logique propositionnelle.
Sortie : RÉUSSITE s'il existe une interprétation des variables de la formule qui l'évalue à 1, c'est-à-dire si la formule est satisfiable ; ÉCHEC sinon.
— 3-COL (3-Colorabilité, [GJS74])
Entrée : Un graphe.
Sortie : RÉUSSITE s'il est possible, avec trois couleurs différentes, de colorer chaque point du graphe de telle sorte que deux mêmes couleurs ne soient pas adjacentes ; ÉCHEC sinon.

■

Étant donnée une classe C, nous pouvons considérer son complément coC. Si $A \subseteq \Sigma^+$ est un sous-ensemble de mots non vides, alors

$$\overline{A} := \Sigma^+ - A = \{x \in \Sigma^+ : x \notin A\}.$$

Si C est une classe de langages, alors

$$\mathsf{coC} := \{A \colon \overline{A} \in \mathsf{C}\}.$$

Exemple 35. La classe de complexité coNP contient les problèmes dont les contre-exemples peuvent être vérifiés en temps polynomial.
TAUT (Tautologie)
Entrée : Une formule de la logique propositionnelle.
Sortie : RÉUSSITE si la formule est une *tautologie*, c'est-à-dire si elle est logiquement équivalente à 1 ; ÉCHEC sinon.

■

Théorème 36 (Théorème 8.5.4, [Sav98])**.**

$$\mathsf{P} \subseteq \mathsf{NP} \subseteq \mathsf{PSPACE} \subseteq \mathsf{EXP} \subseteq \mathsf{NEXP}.$$

Les problèmes qui consistent à savoir si $\mathsf{P} = \mathsf{NP}$ ou si $\mathsf{NP} = \mathsf{coNP}$, et même comment aborder le problème lui-même (par des méthodes algébriques, de comptage, de diagonalisation, etc.), sont encore ouverts, mais l'opinion standard est que $\mathsf{P} \neq \mathsf{NP}$ ([HMU01, CJW06, Ins20]) et que ce problème est difficile à résoudre ([Aar05]). Pour se faire une bonne idée de la pensée actuelle sur ce problème, nous nous référons aux enquêtes de Gasarch [Gas19].

3.5.2 Réductibilité polynomiale et complétude

Un langage $\mathcal{L}_1 \subseteq \Sigma^*$ est *réductible polynomialement* vers un langage $\mathcal{L}_1 \subseteq \Sigma^*$, ce que nous écrivons

$$\mathcal{L}_1 \preceq_P \mathcal{L}_2,$$

s'il existe une machine de Turing M qui s'exécute en temps polynomial telle que

$$x \in \mathcal{L}_1 \text{ si et seulement si } M(x) \in \mathcal{L}_2.$$

De manière intuitive, cette définition traduit le fait que s'il existe une réduction polynomiale de $\mathcal{L}_1$ vers $\mathcal{L}_2$, alors $\mathcal{L}_1$ ne peut pas être plus difficile (long, coûteux, etc.) que $\mathcal{L}_2$. Nous retrouverons cette idée à plusieurs reprises dans cet ouvrage, notamment lorsque nous comparerons les systèmes de représentations de formules (les *systèmes de formes normales*) en fonction de leur efficacité à représenter des fonctions, au chapitre 6 ; voir la définition 112.

Théorème 37. Si $\mathcal{L} \in \mathcal{P}$ et $\mathcal{L}' \preceq_P \mathcal{L}$, alors $\mathcal{L}' \in \mathcal{P}$.

Examinons à présent les langages les plus durs d'une classe de complexité donnée selon l'ordre $\preceq_P$. Soit C un ensemble de langages sur 01. Un langage $\mathcal{L} \subseteq \Sigma^*$ est *difficile pour* C ou C-*difficile* si $\mathcal{L}' \preceq_P \mathcal{L}$ pour tout $\mathsf{L}' \in \mathsf{C}$. Un langage $\mathcal{L} \subseteq \Sigma^*$ est *complet pour* C ou C-*complet* s'il est C-difficile et appartient à C.

Une stratégie typique pour démontrer qu'un problème $\mathcal{L}$ est complet pour une classe C est de démontrer tout d'abord que le problème appartient à la classe, c'est-à-dire $\mathcal{L} \in \mathsf{C}$, et ensuite de trouver un problème $\mathcal{L}^*$ dont la complétude est connue et de montrer que $\mathcal{L}^* \preceq_P \mathcal{L}$. Voir, par exemple, [Kar72].

Exemple 38. Le problème QBF est PSPACE-complet. Les problèmes SAT et 3COL sont NP-complets. ■

3.5.3 Hiérarchie polynomiale

L'un des problèmes que nous allons étudier se situe dans une des classes de complexité qui forme la hiérarchie polynomiale (théorème 73) ; nous la présentons dans cette section.

Nous adoptons les notations de [SU02] et présentons la notion d'*oracle* de manière informelle. Bien que, dans le monde réel, tout calcul ait un coût (en temps, en espace, etc.), nous pouvons imaginer un algorithme qui, lors de son exécution, fait appel à un autre sous-algorithme qui, lui, renvoie sa réponse de manière *instantanée*, sans aucun coût. Ce sous-algorithme est appelé un *oracle*. En termes de machines de Turing, un oracle est une sous-machine dont la réponse influence le comportement de la machine globale qui l'utilise. Ainsi, une requête à l'oracle ne coûte, finalement, qu'une seule étape lors du déroulement d'un calcul par la machine globale.

Étant données deux classes de complexité C et A, nous indiquons par C^{A} l'ensemble des problèmes de décision qui peuvent être résolus par une machine de Turing augmentée d'un oracle pour un certain problème A-complet.

Définition 39. La *hiérarchie polynomiale*, parfois appelée PH (pour *polynomial hierarchy*) est la suite de classes suivante.

1. $\Delta_0^{\mathsf{P}} = \Sigma_0^{\mathsf{P}} = \Pi_0^{\mathsf{P}} = \mathsf{P}$, et
2. pour tout $i \geq 0$,
   — $\Delta_{i+1}^{\mathsf{P}} = \mathsf{P}^{\Sigma_i^{\mathsf{P}}}$,
   — $\Sigma_{i+1}^{\mathsf{P}} = \mathsf{NP}^{\Sigma_i^{\mathsf{P}}}$,
   — $\Pi_{i+1}^{\mathsf{P}} = \mathsf{coNP}^{\Sigma_i^{\mathsf{P}}}$.

Notons que nous retrouvons, au premier niveau de cette hiérarchie, les classes de complexité familières dont nous avons rappelé les définitions ci-haut :
— $\Delta_1^{\mathsf{P}} = \mathsf{P}^{\Sigma_0^{\mathsf{P}}} = \mathsf{P}^{\mathsf{P}} = \mathsf{P}$,
— $\Sigma_1^{\mathsf{P}} = \mathsf{NP}^{\Sigma_0^{\mathsf{P}}} = \mathsf{NP}^{\mathsf{P}} = \mathsf{NP}$,
— $\Pi_1^{\mathsf{P}} = \mathsf{coNP}^{\Sigma_0^{\mathsf{P}}} = \mathsf{coNP}^{\mathsf{P}} = \mathsf{coNP}$.
Dans la figure 3.11, nous avons représenté les premiers niveaux de la hiérarchie polynomiale, ordonnés par inclusion.

Les problèmes de $\mathsf{EXPTIME}$ sont parfois dits *intractables* [Sav98]. Les problèmes qui se trouvent entre NP et Σ_2^{P} sont parfois dits *modérément intractables*. C'est le cas d'un des problèmes que nous allons étudier dans le chapitre 5 (théorème 73).

Dans le chapitre 5, nous utiliserons une caractérisation alternative et pratique des classes Σ_i^{P} par des formules Booléennes quantifiées. Une relation n-aire sur les mots est un sous-ensemble de $\Sigma^+ \times \cdots \times \Sigma^+$ n fois. Si R est une relation et C une classe de langages, alors la notation $R \in \mathsf{C}$ signifie que le langage

$$\{x_1 \# x_2 \# \ldots \# x_n : R(x_1, x_2, \ldots, x_n)\}$$

est dans C pour un certain symbole $\# \notin \Sigma$.

Théorème 40 ([SM73, Sto76, Wra76])**.** Soit $\mathcal{L} \subseteq \Sigma^+$. Le langage $\mathcal{L}$ est dans Σ_i^{P} si et seulement si il existe un polynôme p, un alphabet Γ, et une relation $(i+1)$-aire $R \in \mathsf{P}$ tels que pour tout $x \in \Sigma^+$, $x \in \mathcal{L}$ si et seulement si

$$\exists y_1 \forall y_2 \exists y_3 \ldots Q_i y_i R(x, y_1, y_2, \ldots, y_i),$$

où les quantificateurs apparaissent de manière alternée, et donc où $Q_i = \exists$ si i est impair et $Q_i = \forall$ si i est pair, et où les y_j sont des mots de Γ^+ de taille majorée par $p(|x|)$.

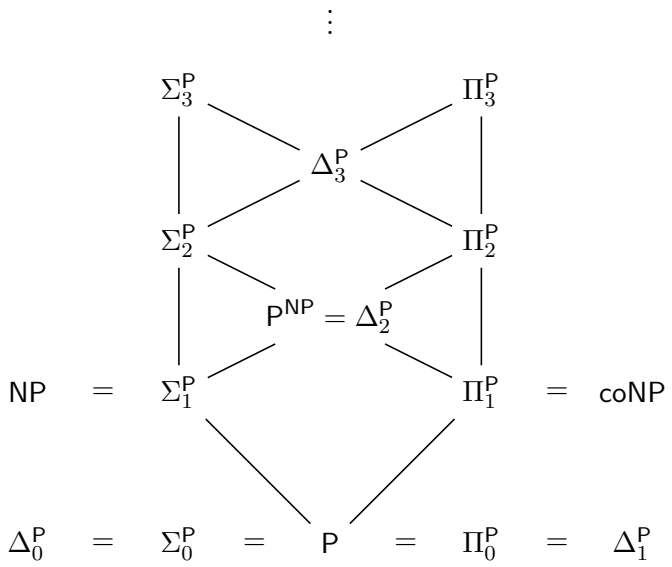


FIGURE 3.11 – Hiérarchie polynomiale : les classes sont ordonnées par inclusion.

Dans le chapitre 5, nous étudions le problème de minimisation de formules médianes. En ce qui concerne les fonctions Booléennes en général, ainsi que pour les fonctions Booléennes monotones, c'est-à-dire qui ne contiennent que les opérateurs de conjonction et de disjonction sans négation, les problèmes de minimisation ou de comptage d'assignements qui rendent une formule valide amènent à l'étude de la classe DP, qui se trouve au troisième niveau de la hiérarchie, entre NP et Σ_2^{P}.

Nous nous référons à [GHM08] pour une étude plus approfondie des classes de complexités et des problèmes que nous mentionnons ci-après. En particulier, nous adoptons les mêmes notations ainsi que les noms anglais donnés aux problèmes.

Définition 41. Un *monôme* est une conjonction de littéraux, c'est-à-dire de variables et de variables niées. Un *implicant* d'une formule Booléenne f est un monôme C tel que C implique f. Un monôme C est un *implicant premier* (*prime implicant*) d'une formule Booléenne f si et seulement si C est un implicant premier de f et si aucun sous-ensemble propre de C n'est un implicant de f.

Exemple 42. Le monôme $x_1 \wedge x_2$ est un implicant de la formule $f := (x_1 \wedge x_2) \vee x_2$. Ce n'est pas un implicant premier de f, car $x_2 \subset x_1 \wedge x_2$, et x_2 est déjà un implicant premier de f.

Définition 43 ([Pap03])**.** La classe DP est l'ensemble des langages $\mathcal{L}$ qui peuvent s'exprimer comme une intersection d'un langage de NP et un langage de coNP ; c'est-à-dire tels qu'il existe deux langages $\mathcal{L}_1 \in \mathsf{NP}$ et $\mathcal{L}_2 \in \mathsf{coNP}$ tels que

$$\mathcal{L} = \mathcal{L}_1 \cap \mathcal{L}_1.$$

Proposition 44. Nous avons les inclusions suivantes :

$$\mathsf{NP} \subset \mathsf{DP};$$
$$\mathsf{coNP} \subset \mathsf{DP};$$
$$\mathsf{DP} \subset \Sigma_2^{\mathsf{P}}.$$

Considérons à présent les problèmes suivants, qui sont particulièrement intéressants pour notre étude de minimisation de formules.

— IsPrimi (Implicant premier) :
  Entrée : une formule Booléenne f et un monôme C ;
  Sortie : Réussite si C est un implicant premier de f. Échec sinon.
— PrimiSize (Taille d'implicant premier) :
  Entrée : une formule Booléenne f et un entier k ;
  Sortie : Réussite si f a un implicant premier de taille inférieure à k. Échec sinon.
— MinDNFSize (Taille de DNF) :
  Entrée : une formule Booléenne en forme normale disjonctive et un entier k ;
  Sortie : Réussite s'il existe une DNF pour f avec au plus k occurrences de variables. Échec sinon.

Théorème 45 ([GHM08])**.** Si f est une formule Booléenne arbitraire, alors IsPrimi est DP-complet. Si f est une formule Booléenne monotone, alors IsPrimi $\in$ L.

Théorème 46. Si f est une formule Booléenne arbitraire, alors PrimiSize est Σ_2^{P}-complet ([Uma01]). Si f est une formule Booléenne monotone, alors PrimiSize est NP-complet ([GHM08]).

Théorème 47 ([Uma01])**.** MinDNFSize est Σ_2^{P}-complet.

Ces résultats sont cohérents avec ceux obtenus pour les formules médianes dans le chapitre 5, en ce sens que les problèmes considérés sont modérément intractables, voire intractables : ils font partie des problèmes les plus difficiles de NP ou sont dans des classes de complexité supérieures. En particulier, compter le nombre d'implicants premiers d'une fonction Booléenne (problème MinDNFSize) fournit des informations sur la taille de la forme normale disjonctive de cette fonction, qui est alors la disjonction de ces implicants [CH11].

3.5.4 Uniformité

Nous avons rappelé, dans le cadre des machines de Turing, certaines classes de complexité importantes. De même que pour ce modèle de calcul, il est possible de définir des classes de complexité propres aux circuits. Ce résumé succinct est inspiré de [Ruz81, FH03, Sav98].

Familles de circuits

Un circuit qui contient n entrées peut être considéré comme une machine qui reconnaît des mots de taille n, qui sont ceux pour lesquels le circuit est évalué à 1. Par exemple, le circuit représenté dans la figure 3.10 , page 42, et qui calcule la fonction ternaire $f^{(3)}_{\text{mod } 3}$ qui vaut 1 sur le point $\mathbf{x}$ si et seulement si $\mathrm{w}(\mathbf{x}) = 0$ modulo 3 reconnaît les mots 000 et 111, c'est-à-dire le langage

$$\{000, 111\} \subseteq \Sigma^+.$$

Le circuit représenté dans la figure 4.7, page 62, et qui calcule la majorité ternaire μ reconnaît les mots 011, 101, 110, 111, c'est-à-dire le langage

$$\{011, 101, 110, 111\} \subseteq \Sigma^+.$$

Afin de considérer des ensembles infinis de mots, nous considérons des *familles* de circuits, qui sont des collections infinies de circuits $(C_n)_n$ qui prennent en compte toutes les tailles d'entrées

possibles. Ainsi, une famille de circuits peut reconnaître des langages, tout comme une machine de Turing.

Une famille de circuits est un modèle *non-uniforme*, en ce sens que, pour des tailles d'entrées différentes, les calculs peuvent être très différents : en effet, à une taille d'entrée correspond un circuit en particulier. L'uniformité est alors apportée par des contraintes sur le coût des calculs (par exemple, une garantie sur un temps de calcul polynomial). Les familles non-uniformes de circuits sont un modèle trop puissant : elles peuvent potentiellement calculer des fonctions que les machines de Turing ne peuvent pas. La thèse de Church-Turing (voir, par exemple, [Cop19]) concerne le fait que la notion intuitive de « fonction effectivement calculable » correspond aux calculs effectués par une machine de Turing. Les familles de circuits non-uniformes peuvent calculer des fonctions qui ne sont pas calculables par des machines de Turing. Pour cette raison, on préfère à ces familles d'autres modèles de calculs. Les circuits non-uniformes peuvent être utilisés, par exemple, pour déterminer des bornes inférieures qui seront appliquées aux circuits uniformes en particulier et, ainsi, à d'autres modèles de calculs qui leurs sont équivalents. Pour plus d'informations sur les circuits uniformes, nous nous référons à [Ruz81].

Définition 48. Une *famille de circuits* $C = \{C_1, C_2, \ldots\}$ est une collection de circuits logiques pour lesquels C_n a n entrées et $m(n)$ sorties pour une certaine fonction $m : \mathbb{N} \to \mathbb{N}$. Une *famille de circuits uniforme en temps* $r(n)$ *(respectivement en espace* $r(n)$*)* est une famille de circuits pour laquelle il existe une machine de Turing M telle que pour tout entier n en notation unaire, c'est-à-dire 1^n, inscrit en entrée sur son ruban, M écrit la description de C_n en sortie sur son ruban avec un coût en espace (respectivement en temps) de $r(n)$.

Classes de complexité

Nous donnons ici quelques exemples de classes de complexités importantes dans le domaine des circuits. Certaines sont liées à des problèmes de comptage ou de satisfiabilité.

Définition 49 ([Per14])**.** La classe NC^k (*Nick's Class*, en l'honneur de Nick Pippenger) contient les langages $\mathcal{L}$ reconnus par une famille uniforme de circuits Booléens de taille polynomiale et de profondeur $\mathcal{O}(\log^k(n))$ avec n la taille de l'entrée. La classe NC est l'union des classes NC^k, pour $k \geq 1$:

$$\mathsf{NC} = \bigcup_{k \geq 1} \mathsf{NC}^k.$$

Une *porte à seuil-k* (*k-threshold gate*) est une porte logique de degré entrant arbitraire n et qui renvoie 1 si au moins k entrées valent 1. La majorité μ_{2n+1} est en particulier une porte à seuil. Ces portes logiques nous permettent de définir des sous-classes de NC^k en restreignant le choix des portes logiques ; par exemple, AC^k est une restriction de NC^k sur la base $\{\wedge, \vee\}$.

Définition 50. La classe AC^k (*Alternation Circuits*) est la classe des langages reconnus par une famille de circuits de taille polynomiale et de profondeur $\mathcal{O}(\log^k(n))$ avec n la taille de l'entrée, sur la base $\{\wedge, \vee\}$ de degrés entrants arbitraires. De plus,

$$\mathsf{AC} = \bigcup_{k \geq 1} \mathsf{AC}^k.$$

La classe TC^k (*Threshold Circuits*) est la classe de langages reconnus reconnus par une famille de circuits de taille polynomiale et de profondeur $\mathcal{O}(\log^k(n))$ avec n la taille de l'entrée, sur la base $\{\wedge, \vee, \mu_{2p+1}\}$ de degrés entrants arbitraires. De plus,

$$\mathsf{TC} = \bigcup_{k \geq 1} \mathsf{TC}^k.$$

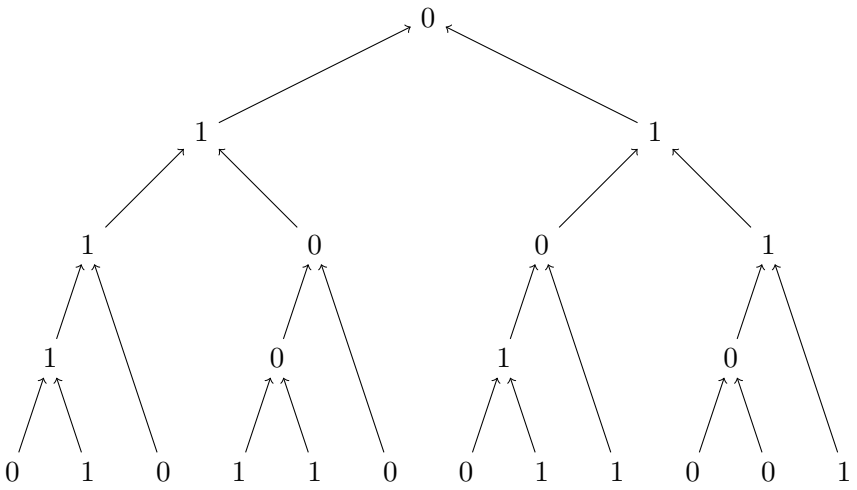


FIGURE 3.12 – Calcul parallélisé de la parité du mot 010110011001.

Proposition 51. Nous avons les inclusions de classes suivantes :

$$\mathsf{NC}^1 \subseteq \mathsf{L} \subseteq \mathsf{NL} \subseteq \mathsf{NC}^2 \subseteq \mathsf{NC} \subseteq \mathsf{P},$$

avec L les langages reconnus par un algorithme déterministe qui fonctionne en espace logarithmique $\mathcal{O}(\log n)$, et NL les langages reconnus par un algorithme non déterministe qui fonctionne en espace logarithmique $\mathcal{O}(\log n)$.

Nous avons également les inclusions suivantes ([AB09]) :

$$\mathsf{NC}^k \subseteq \mathsf{AC}^k \subseteq \mathsf{TC}^k \subseteq \mathsf{NC}^{k+1}.$$

Une conséquence immédiate de cette dernière série d'inclusions est que $\mathsf{NC} = \mathsf{AC} = \mathsf{TC}$. Un intérêt pour l'étude de ces classes est que NC correspond, intuitivement, aux langages pour lesquels il existe des algorithmes parallèles efficaces ; voir, par exemple, [AB09].

Exemple 52. Considérons le langage $\mathsf{PARITÉ}$ qui contient les mots binaires qui ont un nombre pair de 1 :

$$\mathsf{PARITÉ} = \{x \colon \mathrm{w}(x) = 0 \mod 2\}.$$

Ce problème est dans NC^1. En effet, déterminer la parité d'une entrée x de taille n consiste à déterminer la parité p_1 de la première moitié de l'entrée, ainsi que la parité p_2 de la seconde moitié, puis de calculer la parité du mot p_1p_2. En appliquant ce procédé récursivement comme dans la figure 3.12, dans laquelle la parité du mot 001010011111 est testée, nous obtenons un circuit de profondeur $\mathcal{O}(\log n)$ en général (4 dans la figure). ■

Le problème qui consiste à savoir si une majorité de certificats sont valides a été introduite par Gill ([Gil77]) dans le contexte des machines probabilistes, d'où le nom plus connu de PP (*Probabilistic Polynomial time*). Nous nous référons également à [San69] pour une présentation plus complète et formelle des machines de Turing probabilistes.

Une *machine de Turing probabiliste* est une machine de Turing déterministe (voir définition 13) augmentée d'états spéciaux correspondant au lancer d'une pièce bien équilibrée. Pour chaque état de lancer de pièce, la relation de transition spécifie deux états d'arrivée possibles, qui correspondent aux résultats possibles du lancer. Une machine de Turing probabiliste est *bornée polynomialement* s'il existe un polynôme $p(n)$ telle que tout calcul de la machine sur des entrées de taille n s'arrête en au plus $p(n)$ étapes.

Définition 53. La classe PP est la classe des langages reconnus par des machines de Turing probabilistes et bornées polynomialement.

Il en existe une définition équivalente ; une classe A est dans PP si pour tout mot x, $x \in \mathsf{A}$ si et seulement si au moins la moitié des chemins d'une machine N non déterministe polynomiale sont acceptants pour x.

Définition 54. La classe PP est l'ensemble des langages $\mathcal{L}$ tels qu'il existe $\mathsf{B} \in \mathsf{P}$ et un polynôme $p(n)$ tels que pour tout $x \in \Sigma^*$

$$x \in \mathsf{A} \quad \text{si et seulement si} \quad |\{y \in \Sigma^{p(|x|)}, (x,y) \in \mathsf{B}\}| \geq \frac{1}{2} \cdot |\Sigma|^{p(|x|)},$$

c'est-à-dire si la majorité des mots $y \in \Sigma^{p(|x|)}$ vérifie $(x,y) \in \mathsf{B}$.

Exemple 55. Nous donnons ici un exemple de problème PP-complet.
MajSAT (Majorité SAT)
Entrée : Une formule Booléenne non quantifiée $f(x_1, \ldots, x_n)$;
Sortie : Réussite si au moins la moitié des affectations des variables $(a_1, \ldots, a_n) \in \{0,1\}^n$ satisfait $f(a_1, \ldots, a_n)$; Échec sinon.

Nous quittons un instant le domaine des circuits pour nous concentrer sur la fonction médiane ou, dans le domaine Booléen, la majorité à trois entrées.

4

Médiane

οὐ γὰρ ᾠόμην ἔγωγε οὕτω παρ᾽ ὀλίγον ἔσεσθαι ἀλλὰ παρὰ πολύ : νῦν δέ, ὡς ἔοικεν, εἰ τριάκοντα μόναι μετέπεσον τῶν ψήφων, ἀπεπεφεύγη ἄν.
Ce qui me surprend bien plus, c'est le nombre des voix pour ou contre ; j'étais bien loin de m'attendre à être condamné à une si faible majorité ; car, à ce qu'il paraît, il n'aurait fallu que trois voix de plus pour que je fusse absous.

Apologie de Socrate, Platon ([Pla27]).

Sommaire

4.1 Histoire succincte

Nous nous référons aux ouvrages [Har79, LLTdBN11, Mon12] pour une présentation plus détaillée de la médiane et de son histoire.

La médiane est un objet mathématique qui connaît au moins deux définitions très liées. D'une part, la médiane peut être définie de manière géométrique. Dans un espace métrique, c'est-à-dire un ensemble de points muni d'une notion de distance entre ceux-ci, une médiane d'une famille de points est n'importe quel point dont la somme des distances aux points de la famille est minimal [BB84]. Historiquement, cette notion de médiane apparaît dans le calcul de statistiques comme une caractéristique de valeur centrale d'une série statistique. Euler en 1749 ([Eul22]) et Mayer ([May50]) étudient la médiane comme un moyen de diviser les observations d'un ensemble de

données en deux parties égales. Par la suite, Cournot introduit le terme de *médiane* dans [Cou43], et Galton introduit le terme anglais de *median* dans [Gal83], lorsqu'il discute de méthodes statistiques. Boscovich ([BM70]), chargé par le pape Benoît XIV de la mesure d'un méridien afin de corriger la carte géographique terrestre, détermine une approximation géométrique de l'ellipticité de la terre à partir d'observations, qui respecte, en particulier, la contrainte suivante :

> « Étant donné un certain nombre de degrés, trouver la correction qu'il faut faire à chacun d'eux[.] [L]a somme de toutes les corrections, tant positives que négatives, [doit être] la moindre possible[.] »

Laplace présente la méthode géométrique de Boscovich de manière plus analytique et parle de *milieu de probabilité* ([DL74, DL89]) pour désigner un élément minimisant une somme de distances aux éléments de la série statistique considérée. En guise d'exemple de l'utilisation statistique de la médiane, dans la figure 4.1, nous avons représenté le nombre de papes de l'Église catholique en fonction de leur longévité. Sont représentés les 100 derniers papes pour lesquels un âge suffisamment précis (à l'année près) a pu être déterminé [ACM+14], ainsi que la durée de vie médiane (68 ans) et moyenne (70 ans). En analyse de données et en théorie du choix collectif, Monjardet et Barthélemy ([BM81]) utilisent le terme de *procédure médiane* pour désigner l'application qui fait correspondre à un n-uplet d'éléments d'un espace métrique l'ensemble de ses médianes (métriques), qui n'est pas nécessairement un singleton.

D'autre part, la médiane a aussi été étudiée d'un point de vue purement combinatoire, sur des ensembles ordonnés. Sholander ([Sho52, Sho54a, Sho54b]) étudie la médiane dans les treillis, et introduit la structure de *demi-treillis à médiane* (*median semilattice*), qui comprend en particulier les treillis distributifs et les arbres. Dans le cas d'une médiane à 3 éléments d'un treillis distributif, Birkhoff et Kiss ([BK47]) montrent que leur médiane algébrique, définie à partir des opérations $\wedge_L$ et $\vee_L$ du treillis de la manière suivante

$$\begin{aligned}\mathtt{m}_L(x_1, x_2, x_3) &:= (x_1 \wedge_L x_2) \vee_L (x_2 \wedge_L x_3) \vee_L (x_3 \wedge_L x_1) \\ &\equiv (x_1 \vee_L x_2) \wedge_L (x_2 \vee_L x_3) \wedge_L (x_3 \vee_L x_1)\end{aligned}$$

est également une médiane métrique, en considérant la distance sur le graphe du treillis par la taille de chemins entre points.

4.2 Médiane géométrique

Nous adoptons les notations et définitions de [Die63, Mon12] ainsi que l'ouvrage [Eid09] pour la construction de la médiane géométrique dans le plan Euclidien. Étant donné un ensemble E, une *distance sur* E est une application à valeurs réelles

$$\delta : E \times E \to \mathbb{R}$$

qui vérifie les propriétés suivantes.

1. $\delta(x, y) \geq 0$,
2. $\delta(x, y) = 0$ si et seulement si $x = y$,
3. $\delta(x, y) = \delta(y, x)$, et
4. l'inégalité triangulaire
$$\delta(x, z) \leq \delta(x, y) + \delta(y, z)$$

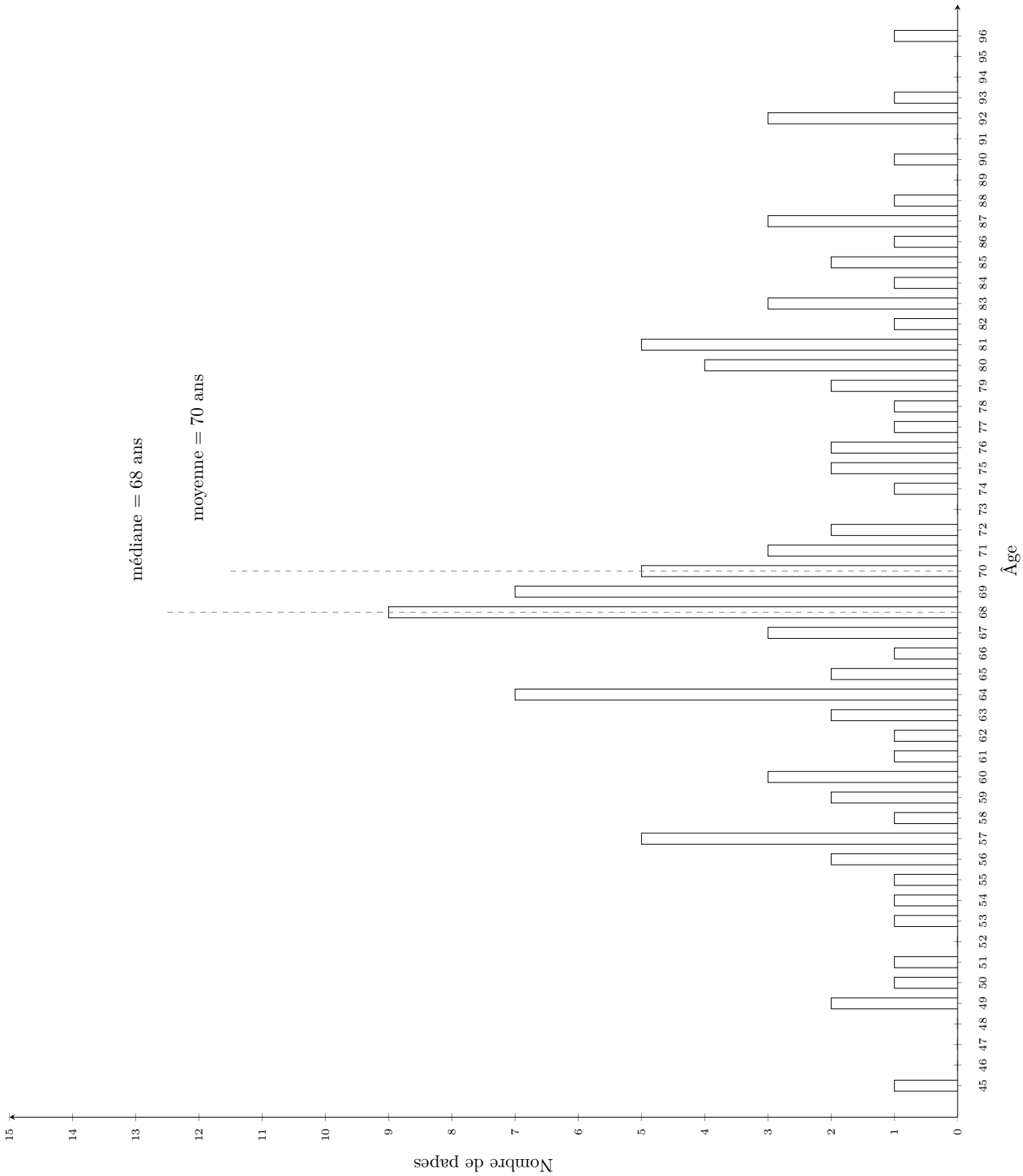


FIGURE 4.1 – Médiane et moyenne sur un histogramme : longévité de papes.

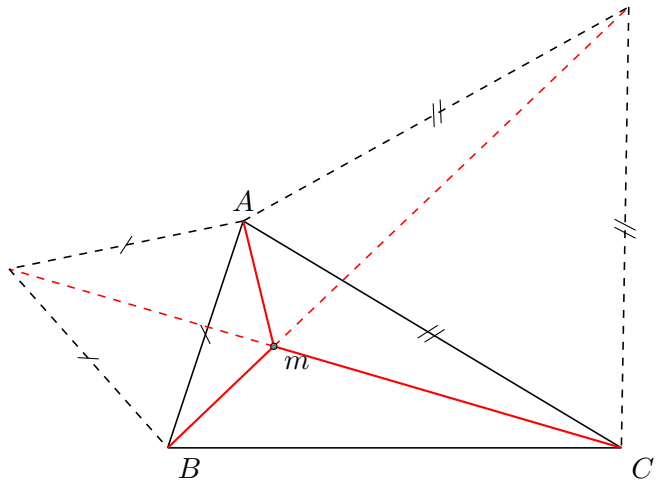


FIGURE 4.2 – Point de Fermat-Torricelli m dans un triangle ABC.

pour tous $x, y, z \in E$.

Dans la figure 4.2, nous donnons une représentation de la médiane géométrique correspondant au problème de minimisation de sommes de distances à un point. Pour le construire, il suffit de tracer deux triangles équilatéraux sur deux faces du triangle ABC ; l'intersection des droites qui passent par le sommet externe des triangles au sommet opposé de ABC correspondant est la médiane du triangle ABC. Pour une démonstration que ce point est unique et minimise la somme des distances aux sommets de ABC, nous nous référons à [Eid09].

Fermat ([DF91]) énonce ce problème pour 3 points, dans un triangle dont les angles sont inférieurs à $2\pi/3$; en géométrie, le point résultant de la construction est appelé le *point de Fermat* ou *point de Torricelli*, du nom du mathématicien qui en fournit une solution ([TL14]).

Difficulté du calcul de la médiane

La *médiane géométrique* (*estimateur* L_1 en statistiques) est une généralisation de ce problème à un nombre quelconque de points. C'est un problème d'optimisation standard du domaine de la recherche opérationnelle, lorsque l'on recherche un point qui permette d'atteindre tout autre le plus rapidement possible. Formellement et dans un espace euclidien de dimension k, la médiane géométrique y d'un ensemble de points $x_1, \ldots, x_n \in \mathbb{R}^k$ est définie par

$$\operatorname*{argmin}_y \sum_{i=1}^{n} \delta(x_i, y)$$

où $\delta(a, b)$ est la distance euclidienne entre deux points a et b de $\mathbb{R}^k$, et où $\operatorname{argmin}_y f(y)$ renvoie l'argument $y \in \mathbb{R}^k$ pour lequel la fonction en $f(y)$ est minimale, unique dès lors que les points ne sont pas collinéaires ([VZ00]). Le *problème du lieu de Fermat-Weber* (*Fermat-Weber location problem*) consiste à trouver ce point. Il existe des algorithmes efficaces (polynomiaux) pour résoudre ce problème de manière approchée, c'est-à-dire se rapprocher du point y avec une

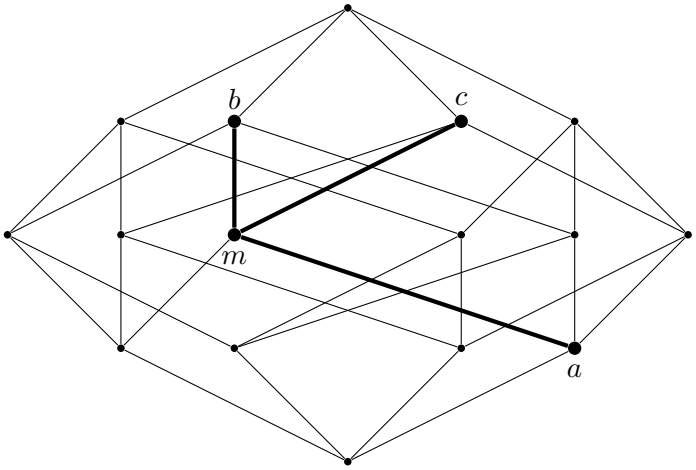


FIGURE 4.3 – Médiane de trois points sur un treillis distributif.

précision $\varepsilon \in \mathbb{Q}$ (voir, par exemple, [CT90, AM12]). Dans certains cas, c'est-à-dire pour des espaces métriques munis d'une distance particulière, le calcul de la médiane est NP-difficile ([BTT89]). Dans d'autres, et c'est le cas pour les espaces à médianes, ce calcul devient beaucoup plus aisé ([Mon12]).

Lecture de la médiane métrique dans un treillis distributif

Nous considérons le treillis distributif (borné) de la figure 4.3. La distance entre deux points est ici définie comme elle l'est typiquement pour les graphes : c'est le nombre d'arêtes du plus court chemin entre les points, aussi appelée *distance géodésique*. Ici, la fonction

$$x \mapsto \delta(a, x) + \delta(b, x) + \delta(c, x)$$

est minimale pour l'unique point $x = m$, et vaut alors 3. Sur la figure, nous avons représenté ces trois distances avec des traits plus gras.

4.3 Espaces à médianes

Nous donnons ici quelques exemples d'espaces dans lesquels la médiane apparaît assez naturellement, et qui forment la base théorique de l'étude des espaces à médianes, enrichis plus tard par des travaux plus théoriques ainsi que des applications plus pratiques dans le domaine de la simplification de circuits, par exemple, dont nous parlerons dans la section 4.4.

Nous utilisons les notations et définitions de [BH83, BB84, Mon12].

4.3.1 Algèbre ternaire

Dans la section précédente, nous avons donné une définition de la médiane ternaire m sur un treillis distributif (borné). Plus généralement, c'est-à-dire sans la condition de distributivité ni les opérations $\wedge_L$ et $\vee_L$ d'un treillis L, nous pouvons définir une *algèbre médiane* comme un ensemble muni d'une seule opération ternaire qui vérifie les identités qui sont vraies pour la médiane sur les treillis distributifs.

Une *algèbre ternaire* (*ternary algebra* en anglais) est un ensemble A muni d'une seule opération ternaire

$$(a, b, c) \to (a\ b\ c).$$

Une algèbre ternaire A est une *algèbre médiane* si elle satisfait les identités suivantes :

$$\begin{aligned}
&(a\ a\ b) = a,\\
&(a\ b\ c) = (b\ a\ c) = (b\ c\ a),\\
&((a\ b\ c)\ d\ e) = (a\ (b\ d\ e)\ (c\ d\ e)).
\end{aligned}$$

Ces identités, comme nous allons le voir dans le Chapitre 5, portent parfois le nom de *majorité*, *symétrie*, et *distributivité*. Tout treillis distributif L peut être vu comme une algèbre médiane en définissant une opération m comme dans la définition 56 :

$$\begin{aligned}
\mathrm{m}(a, b, c) &= (a \wedge_L b) \vee_L (b \wedge_L c) \vee_L (c \wedge_L a)\\
&= (a \vee_L b) \wedge_L (b \vee_L c) \wedge_L (c \vee_L a).
\end{aligned}$$

Une algèbre médiane peut être induite par d'autres structures plus générales, telles que les arbres, dont nous parlons ci-après. Voir également, par exemple, [Zel68].

4.3.2 Graphes à médianes et arbres

Un graphe non orienté est dit *connexe* s'il est d'un seul tenant, c'est-à-dire si chacun de ses sommets est relié à au moins un autre, ou s'il est réduit à un seul sommet. Un *graphe à médianes* (*median graph* en anglais) est graphe non orienté connexe tel que tout triplet a, b, c de sommets admet une médiane métrique unique, c'est-à-dire un unique point m qui minimise la somme

$$\delta(a, m) + \delta(b, m) + \delta(c, m),$$

où δ est la distance géodésique entre deux sommets du graphe. En particulier, le graphe du treillis de la figure 4.3 est un graphe médian. Les *arbres*, qui sont des graphes non orientés, connexes, et acycliques, sont aussi un cas particulier de graphes à médianes.

4.3.3 Demi-treillis à médianes

Un *inf-demi-treillis* (*inf-semilattice*) est un ensemble ordonné tel que tout couple x, y d'éléments admet un infimum $x \wedge_L y$. Un *demi-treillis à médianes* (*median semilattice*) est un inf-demi-treillis D qui vérifie les deux propriétés suivantes :

1. pour tout $x \in D$, l'ensemble des éléments inférieurs ou égaux à x est un treillis distributif ;
2. pour tous $x, y, z \in D$, si $x \vee_L y$, $y \vee_L z$, et $z \vee_L x$ existent alors $x \vee_L y \vee_L z$ existe aussi.

Tout treillis distributif est un demi-treillis à médianes.

Il est possible de passer aisément d'une structure à une autre. Par exemple, l'opération ternaire d'un graphe à médianes qui à trois éléments associe leur (unique) médiane métrique définit une algèbre médiane. Inversement, à partir d'une algèbre médiane et de son opération $(x, y, z) \mapsto (xyz)$, nous pouvons définir un graphe médian en reliant deux éléments x, y par une arête s'il n'existe pas de troisième élément z tel que $(xyz) = z$.

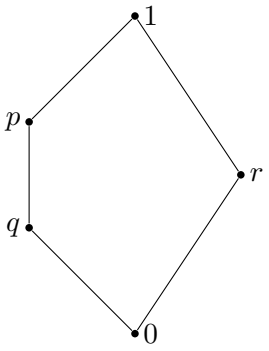


FIGURE 4.4 – Diagramme de Hasse du pentagone N_5.

4.3.4 Médiane sur des treillis non-distributifs

Définition 56. La *médiane* ternaire m, c'est-à-dire la fonction définie par :

$$\begin{aligned} \mathrm{m}(x_1, x_2, x_3) &= (x_1 \wedge_L x_2) \vee_L (x_2 \wedge_L x_3) \vee_L (x_3 \wedge_L x_1) \\ &= (x_1 \vee_L x_2) \wedge_L (x_2 \vee_L x_3) \wedge_L (x_3 \vee_L x_1) \end{aligned}$$

est une fonction polynomiale au sens donné ci-haut. Ces deux définitions sont équivalentes.

La condition de distributivité sur les treillis que nous considérons est importante. Dans un treillis non distributif, nous avons toujours l'inégalité suivante :

$$\mathrm{m}(x_1, x_2, x_3) \leq \mathrm{m}^+(x_1, x_2, x_3),$$

avec

$$\begin{aligned} \mathrm{m}(x_1, x_2, x_3) &:= (x_1 \wedge_L x_2) \vee_L (x_2 \wedge_L x_3) \vee_L (x_3 \wedge_L x_1), \\ \mathrm{m}^+(x_1, x_2, x_3) &:= (x_1 \vee_L x_2) \wedge_L (x_2 \vee_L x_3) \wedge_L (x_3 \vee_L x_1). \end{aligned}$$

Mais un treillis est distributif si et seulement si, pour tout triplet (x, y, z), l'inégalité ci-dessus est une égalité :

$$(x_1 \wedge_L x_2) \vee_L (x_2 \wedge_L x_3) \vee_L (x_3 \wedge_L x_1) = (x_1 \vee_L x_2) \wedge_L (x_2 \vee_L x_3) \wedge_L (x_3 \vee_L x_1).$$

Voir, par exemple, [Men36]. Dans la figure 4.5, nous illustrons le calcul des deux quantités

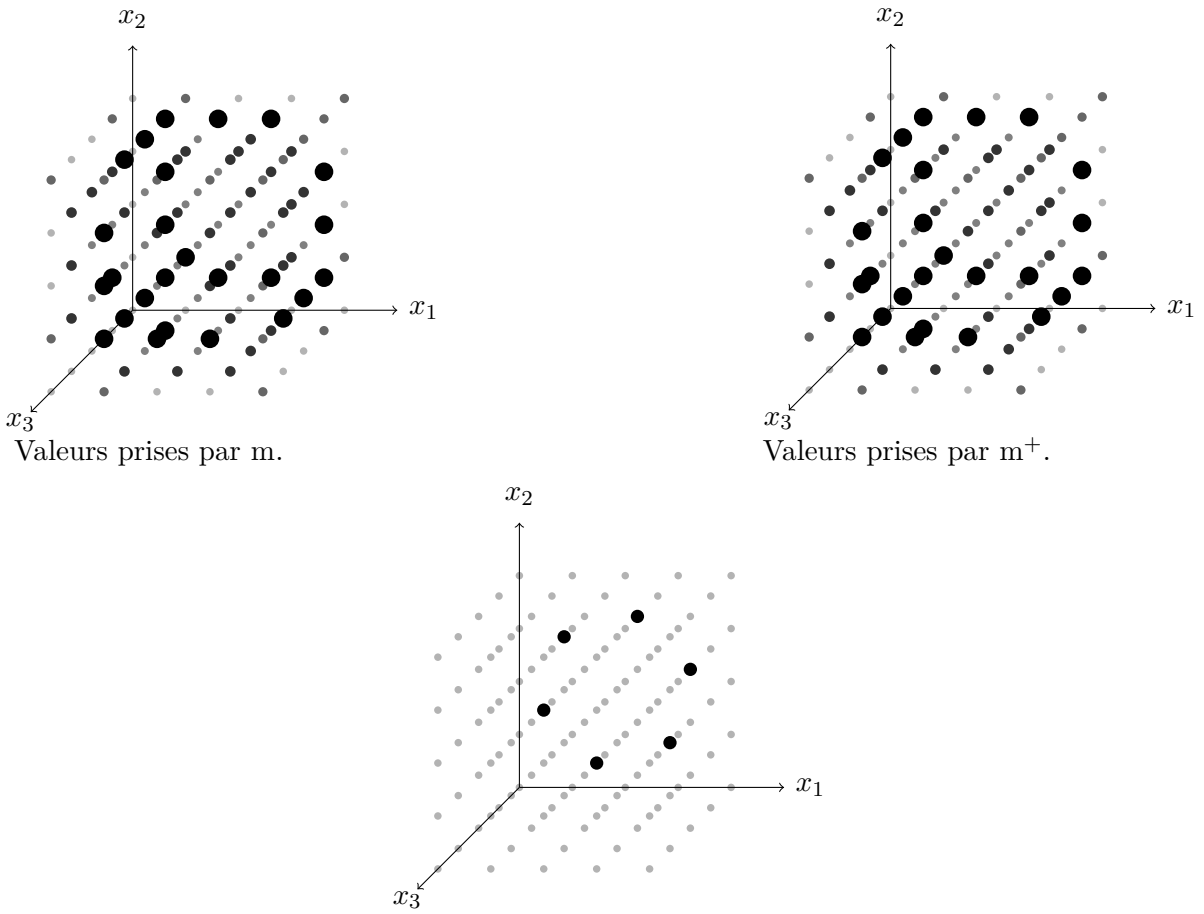
$$\mathrm{m}(x_1, x_2, x_3) = (x_1 \wedge_L x_2) \vee_L (x_2 \wedge_L x_3) \vee_L (x_3 \wedge_L x_1)$$

et

$$\mathrm{m}^+(x_1, x_2, x_3) = (x_1 \vee_L x_2) \wedge_L (x_2 \vee_L x_3) \wedge_L (x_3 \vee_L x_1)$$

sur le treillis non distributif N_5, aussi appelé le *pentagone*, et que nous avons représenté dans la figure 4.4. Par exemple, dans le pentagone,

$$\begin{aligned} \mathrm{m}(p, q, r) &= (p \wedge_L q) \vee_L (q \wedge_L r) \vee_L (r \wedge_L p) \\ &= q \vee_L 0 \vee_L 0 \\ &= q, \end{aligned}$$



Valeurs prises par m.

Valeurs prises par m^+.

Différences entre m et m^+.

FIGURE 4.5 – Comparaison de m et de m^+ sur le pentagone N_5.

mais

$$\begin{aligned}\mathrm{m}^+(p,q,r) &= (p \vee_L q) \wedge_L (q \vee_L r) \wedge_L (r \vee_L p)\\ &= p \wedge_L 1 \wedge_L 1\\ &= p.\end{aligned}$$

Dans la figure 4.5, nous avons représenté l'ensemble des valeurs prises par m et m^+ sur le pentagone N_5. Le codomaine est représenté par des points de tailles et d'opacités différentes. Un troisième graphique met en évidence les points sur lesquels m et m^+ diffèrent : ce sont les points

$$\{(p,q,r),(p,r,q),(q,p,r),(q,r,p),(r,p,q),(r,q,p)\}.$$

4.4 Application : médiane et circuits

Dans cette section, nous présentons les *Graphes Majorité-Inversion* de [AGCDM15], liés à la simplification de circuits basés sur l'opérateur majorité, comme un exemple d'utilisation de

la médiane dans le cadre des circuits, et non des formules.

La *base monotone* $\{\wedge, \vee\}$, avec la négation $\neg$, est la base usuelle pour représenter des fonctions Booléennes à l'aide de circuits et en étudier les caractéristiques telles que la complexité ([Sav98]). La majorité à trois entrées, opération ternaire qui renvoie l'entrée la plus fréquente, a été étudiée comme base dès les années 1960 ; alors, les opérations $\wedge, \vee$ usuelles sont simulées par la majorité ([GMN+60]).

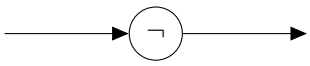
Dans [AGDM14, AGCDM15, CAS+16], les auteurs présentent un type de circuits logiques homogènes, *les graphes Majorité-Inversion* (*Majority-Inverter Graphs (MIGs)*), constitués de portes Majorité à trois entrées et d'entrées qui peuvent être complémentées. Les auteurs proposent des implémentations efficaces (linéaires) de circuits logiques usuels tels que l' *additionneur complet* (*full-adder*), des schémas de correction d'erreurs basés sur la médiane, ainsi que des résultats expérimentaux sur l'efficacité des MIGs pour la représentation de circuits et la simplification de MIGs. En particulier, les MIGs sont $18,6\%$ moins profonds, en moyenne, que les AIGs (*And-Inverter Graphs*), et sont plus aisément simplifiables. Un *additionneur* est un circuit digital qui permet d'effectuer des additions d'entiers. Ils sont utilisés comme une brique de base des microprocesseurs et des unités arithmétiques et logique dans les ordinateurs digitaux. Dans sa forme la plus simple, un *demi-additionneur* à un bit calcule une opération

$$\begin{aligned} \text{ADD}_d : \{0,1\} \times \{0,1\} &\to \{0,1\} \times \{0,1\} \\ (a, b) &\mapsto (s, c_{out}) \end{aligned}$$

qui, à deux entrées $a, b \in \{0,1\} \times \{0,1\}$ associe la somme binaire $s = a \oplus b$ et la retenue $c_{out} = a \wedge b$. Un *additionneur complet*, que l'on peut construire à l'aide de deux demi-additionneurs en série, rajoute la possibilité de propager la retenue, et à trois entrées a, b, c_{in} renvoie la somme $s = a \oplus b \oplus c_{in}$ ainsi que la retenue $c_{out} = m(a, b, c_{in})$.

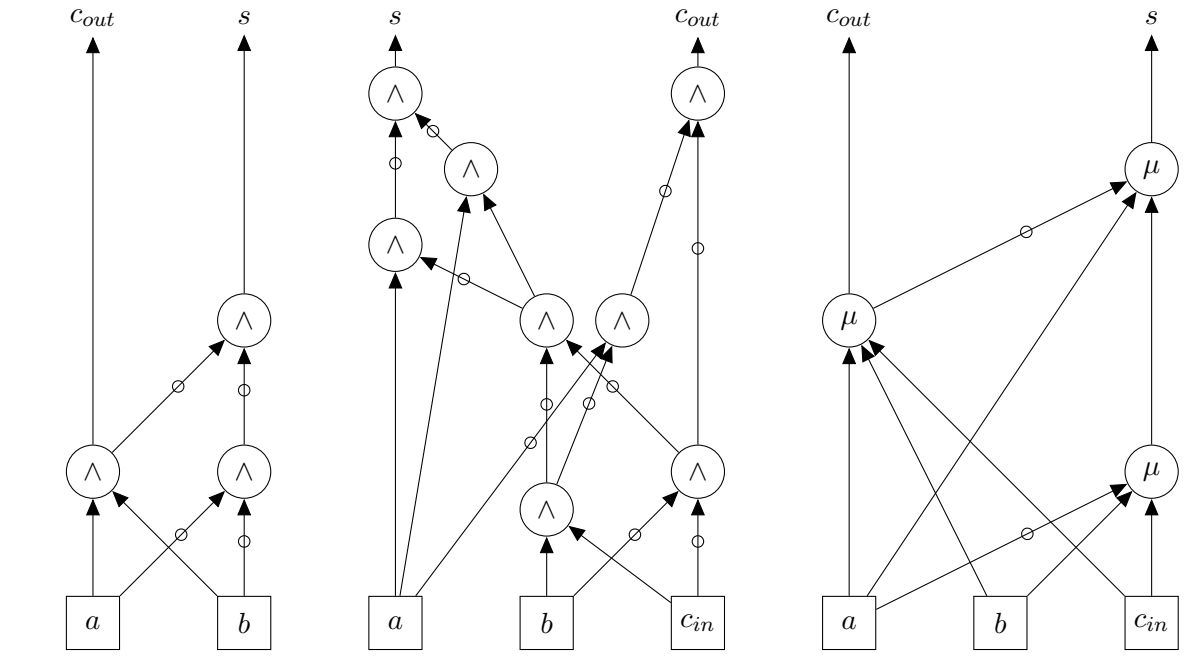
$$\begin{aligned} \text{ADD} : \{0,1\} \times \{0,1\} \times \{0,1\} &\to \{0,1\} \times \{0,1\} \\ (a, b, c_{in}) &\mapsto (s, c_{out}). \end{aligned}$$

Dans la figure 4.6, nous donnons une réalisation de ces circuits dans la base $\{\wedge, \neg\}$, ainsi que dans la base $\{\mu, \neg\}$ pour l'additionneur complet. Pour plus de clarté, les arêtes niées, qui correspondent à la configuration



sont représentées par des flèches décorées d'un cercle, comme représenté ci-dessous.

Dans [CHS+19], les auteurs étudient la synthèse de formules construites, cette fois, à partir de la médiane d'arité 5. En particulier, ils élaborent certaines techniques pour évaluer le nombre de médianes requises pour représenter une fonction donnée, et obtiennent, expérimentalement, une amélioration d'environ 10% en taille et en profondeur des circuits par rapport à ceux obtenus avec les MIGs construits sur la médiane, ternaire. Ces résultats confirment l'intérêt pratique de l'utilisation de connecteurs d'arité différentes, même si, par la suite, nous montrons que les représentations, dans le cadre des formules, sont de complexités équivalentes (proposition 133). Cela nous encourage à travailler, dans le futur, sur des méthodes de comparaison fines des formules médianes.



Demi-additionneur (AIG) Additionneur (AIG) Additionneur (MIG)

FIGURE 4.6 – Circuits logiques pour les additionneurs ([AGCDM15]).

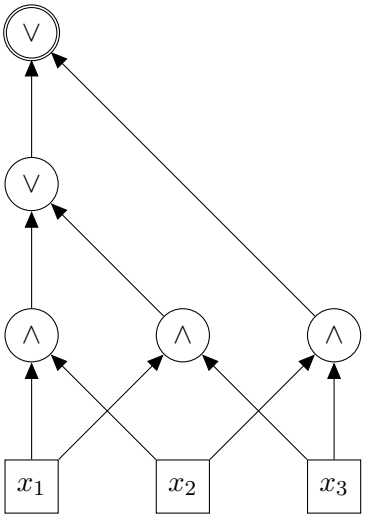


FIGURE 4.7 – Représentation par un DAG d'un circuit logique qui calcule $\mu(x_1, x_2, x_3)$.

5

Formules médianes

Krakowiaczek jeden
Miał koników siedem,
Sześcią do dziewczyny,
W domu został jeden.
Un cracovien avait sept chevaux. Six il en donne à une fille ; à la maison, il lui en reste un.

Comptine polonaise ([Glo92]).

Sommaire

Dans ce chapitre, nous considérons des règles de calcul qui reposent sur le connecteur médiane m, c'est-à-dire le connecteur défini par $(x \wedge_L y) \vee_L (y \wedge_L z) \vee_L (z \wedge_L x)$ sur un treillis distributif $\langle L, \wedge_L, \vee_L \rangle$. Notre but est de représenter de manière efficace certaines classes de fonctions telles que les fonctions polynomiales et les fonctions Booléennes monotones. Ces fonctions peuvent être représentées par des *formules médianes* qui sont des compositions de médianes, de variables, et de constantes dans L. Par exemple, nous pouvons représenter la conjonction Booléenne ternaire $x \wedge y \wedge z$ par la formule

$$\mathrm{m}(\mathrm{m}(x, 0, y), 0, z).$$

Nous étudions une spécification équationnelle des formules médianes, et l'étendons depuis le domaine des fonctions polynomiales sur des treillis distributifs au domaine des fonctions

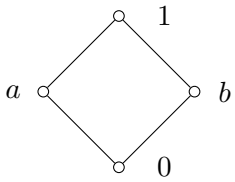


FIGURE 5.1 – Exemple de treillis distributif borné pour $L = \{1, 0, a, b\}$.

Booléennes monotones. Cette spécification contient des équations telles que

$$\mathrm{m}(x, x, y) = x,$$

qui permet en particulier de simplifier des formules médianes, en ce sens que la formule à laquelle on applique cette règle (si possible) a une taille strictement plus petite : elle a un connecteur m en moins.

Cette spécification est *correcte* et *complète*, ce qui est particulièrement intéressant pour la réécriture des formules médianes : la correction implique que l'application de règles conserve l'équivalence logique, et la complétude implique qu'il est possible de réécrire n'importe quelle formule en toute autre qui lui est logiquement équivalente. Nous illustrons l'utilité de cette spécification pour simplifier des formules médianes de manière algébrique.

De plus, nous proposons une définition pour les *formes normales médianes* (*Median Normal Forms* en anglais, ou MNF) qui sont des formules médianes minimales, selon un ordre structurel sur les expressions consistant à compter le nombre de médianes à chaque étage d'une formule vue comme un arbre. Cet ordre permet en particulier de favoriser les formules qui ont le moins de médianes possible.

Nous examinons la complexité calculatoire du problème qui consiste à décider si une formule est, ou non, une MNF, c'est-à-dire le problème qui consiste à minimiser la formule médiane d'une fonction Booléenne monotone donnée. Nous démontrons que ce problème est dans Σ_2^{P} : il est *modérément intractable*. Nous démontrons de plus ce résultat pour les fonctions Booléennes arbitraires, c'est-à-dire sans la condition nécessaire de monotonie.

La plupart des résultats présentés dans ce chapitre ont été publiés dans [CMPS17, CMPS19].

5.1 Axiomatisation des polynômes latticiels

Dans cette section, nous présentons un système équationnel pour le calcul médian qui est à la fois correct et complet. Nous l'utilisons par la suite pour manipuler des expressions médianes. Nous rappelons que $\langle L, \wedge_L, \vee_L \rangle$ est un treillis distributif borné dont le maximum et le minimum sont 1 et 0, respectivement, comme illustré dans la figure 5.1.

5.1.1 Termes médians

L'ensemble des *termes médians* (également appelées *expressions médianes* ou *formules médianes* dans [CFL06]) $\mathbf{M}$ désigne l'ensemble de toutes les formules que l'on peut construire en utilisant des variables, des constantes, et la médiane m. Bien que [CFL06] se concentre sur les formes normales médianes pour représenter des fonctions Booléennes, cette notion s'applique de manière naturelle aux polynômes latticiels, car le langage utilisé pour représenter les fonctions Booléennes monotones et les polynômes latticiels est le même. En effet, l'ensemble de connecteurs $\{\wedge, \vee\}$ est *complet* vis-à-vis des fonctions Booléennes monotones, c'est-à-dire que l'on peut

représenter toute fonction Booléenne monotone par composition de ces deux connecteurs, de variables, et de constantes. Cet ensemble est d'ailleurs parfois appelé la *base monotone* ([Sav98]) dans le domaine des circuits Booléens. Or, nous pouvons aisément exprimer la conjonction et la disjonction par la médiane :

$$\wedge(x, y) \equiv \mathrm{m}(x, 0, y);$$
$$\vee(x, y) \equiv \mathrm{m}(x, 1, y).$$

Donc l'ensemble $\{\mathrm{m}\}$ est également complet vis-à-vis des fonctions Booléennes monotones.

Exemple 57. Les expressions m, $\mathrm{m}(x, x)$, et $x\,\mathrm{m}$ ne sont pas des termes de $\mathbf{M}$. Par contre, les expression x, 1, et $\mathrm{m}(x, y, \mathrm{m}(x, 1, 0))$ sont des termes de $\mathbf{M}$. ■

Étant donnée une formule ϕ de $\mathbf{M}$, on note $\mathrm{d}(\phi)$ sa *profondeur*, définie inductivement de la manière suivante :

1. pour toute variable ou constante a, $\mathrm{d}(a) = 0$;
2. pour toute formule $\phi = \mathrm{m}(\phi_1, \phi_2, \phi_3)$ de $\mathbf{M}$,

$$\mathrm{d}(\phi) = 1 + \max\{\mathrm{d}(\phi_1), \mathrm{d}(\phi_2), \mathrm{d}(\phi_3)\}.$$

La *taille* $|\phi|$ d'une formule médiane ϕ est le nombre de médianes qu'elle contient.

Exemple 58. Si $\phi = \mathrm{m}(\mathrm{m}(x, x, y), \mathrm{m}(x, y, z), v)$, alors $\mathrm{d}(\phi) = 2$ et $|\phi| = 3$. ■

Notons que représenter les formules par des arbres permet de mieux se représenter leur profondeur.

De même que dans la section précédente, deux termes médians ϕ et ψ sont *équivalents*, ce que l'on note $\phi \equiv \psi$, s'ils représentent la même fonction.

Exemple 59. Par exemple, les termes médians

$$\phi_1 = \mathrm{m}(\mathrm{m}(\mathrm{m}(x, u, v), \mathrm{m}(x, y, v), \mathrm{m}(y, u, v)), x, v)$$
$$\phi_2 = \mathrm{m}(\mathrm{m}(u, y, v), x, v)$$

sont équivalents. Par contre, les termes médians

$$\phi_3 = \mathrm{m}(\mathrm{m}(x, y, z), u, v)$$
$$\phi_4 = \mathrm{m}(x, \mathrm{m}(y, z, u), v)$$

ne sont pas équivalents. Ce dernier résultat exprime le fait que m n'est pas *associative* au sens de [Acz04, CM12, Pos40]. ■

5.1.2 Polynômes latticiels

Rappelons que les polynômes latticiels $f : L^n \to L$ sont exactement les solutions du *système de décomposition médiane* [Mar09b].

$$f(\mathbf{x}) = \mathrm{m}(f(\mathbf{x}_k^1), x_k, f(\mathbf{x}_k^0)) \tag{5.1}$$

pour tout $\mathbf{x} = (x_1, \ldots, x_n)$, $k \in [n]$, et où

$$\mathbf{x}_k^c = (x_1, \ldots, x_{k-1}, c, x_{k+1}, \ldots, x_n)$$

avec $c \in L$.

Exemple 60. Appliquons la formule de décomposition médiane au connecteur binaire $\wedge_L$. Dans l'expression $x \wedge_L y$ nous choisissons x pour pivot.

$$\begin{aligned} x \wedge_L y &= \mathrm{m}(1 \wedge_L y, x, 0 \wedge_L y) && \text{d'après l'équation 5.1 ;} \\ &= \mathrm{m}(y, x, 0) && \text{par définition de 0 et 1.} \end{aligned}$$

De manière similaire, on calcule que $x \vee_L y = \mathrm{m}(1, x, y)$. ∎

Une conséquence directe de l'équation 5.1 est que toute fonction polynomiale peut être représentée par un terme médian. En effet, l'application récursive de l'équation 5.1 sur chaque variable $x_1, \ldots, x_n$ d'une fonction polynomiale n-aire f produit une formule médiane qui représente f ; dans [CLMW11], les auteurs proposent un algorithme pour obtenir des représentations médianes de fonctions Booléennes, et que nous reproduisons ici (algorithme 1) pour le cas des fonctions Booléennes monotones. Nous discutons la conversion de fonctions Booléennes quelconques et reproduisons l'algorithme associé lorsque nous étendons nos résultats au cas non-monotone, dans la section 5.4.

Algorithm 1: MMNF (*Monotone Median Normal Form*) - forme normale médiane pour les fonctions Booléennes monotones (d'après [CLMW11])

Entrée: Une fonction Booléenne monotone f.
Sorties: Une formule médiane qui représente f.

si $n \geq 2$ **alors** ;
  $m_1 \leftarrow \mathrm{MMNF}(f(x_1, x_2, \ldots, x_{n-1}, 1))$;
  $m_0 \leftarrow \mathrm{MMNF}(f(x_1, x_2, \ldots, x_{n-1}, 0))$;
  retourner $\mathrm{m}(m_1, x_n, m_0)$;
sinon si $f = \mathbf{0}$ **alors**
  retourner 0 ;
sinon si $f = \mathbf{1}$ **alors**
  retourner 1 ;
sinon
  retourner x_1 ;

Toutefois, cette procédure ne produit pas forcément de formules optimales en termes de taille, ce qui motive notre étude, comme nous l'illustrons par l'exemple 61 ci-dessous.

Exemple 61. Considérons l'opérateur majorité à 5 entrées

$$\mathrm{m}_5 : \mathbf{x} \mapsto \mathrm{m}_5(x_1, x_2, x_3, x_4, x_5)$$

sur le domaine Booléen. En utilisant l'algorithme de décomposition médiane mentionné ci-dessus et en décomposant suivant les variables par ordre d'indices croissant (x_1, puis x_2, etc.), nous pouvons construire une représentation médiane de la fonction représentée par m_5. en connaissant ses valeurs sur tous les points de $\mathbb{B}^5$. La représentation obtenue est de taille $1+2+4+8+16 = 31$, et elle n'est pas optimale : voir la figure 5.2.

En effet, il en existe une plus petite représentation de taille 4, comme cela est démontré dans [MW62] de manière graphique et dans [BM92] de manière algébrique :

$$\mathrm{m}_5(x_1, x_2, x_3, x_4, x_5) = \mathrm{m}(\mathrm{m}(\mathrm{m}(x_2, x_3, x_4), x_4, x_5), \mathrm{m}(x_2, x_3, x_5), x_1).$$

$$
\begin{aligned}
&\mathrm{m}_5(x_1, x_2, x_3, x_4, x_5) = \\
&\mathrm{m}(\mathrm{m}_5(1, x_2, x_3, x_4, x_5), x_1, \mathrm{m}_5(0, x_2, x_3, x_4, x_5)) = \\
&\mathrm{m}(\mathrm{m}(\mathrm{m}_5(1, 1, x_3, x_4, x_5), x_2, \mathrm{m}_5(1, 0, x_3, x_4, x_5)), x_1, \mathrm{m}_5(0, 1, x_3, x_4, x_5), x_2, \mathrm{m}_5(0, 0, x_3, x_4, x_5))) = \\
&\cdots = \\
&\mathrm{m}(\mathrm{m}(\mathrm{m}(\mathrm{m}(\mathrm{m}_5(1,1,1,1,1), x_5, \mathrm{m}_5(1,1,1,1,0)), x_4, \mathrm{m}(\mathrm{m}_5(1,1,1,0,1), x_5, \mathrm{m}_5(1,1,1,0,0))), x_3, \\
&\mathrm{m}(\mathrm{m}(\mathrm{m}_5(1,1,0,1,1), x_5, \mathrm{m}_5(1,1,0,1,0)), x_4, \mathrm{m}(\mathrm{m}_5(1,1,0,0,1), x_5, \mathrm{m}_5(1,1,0,0,0))), x_2, \\
&\mathrm{m}(\mathrm{m}(\mathrm{m}_5(1,0,1,1,1), x_5, \mathrm{m}_5(1,0,1,1,0)), x_4, \mathrm{m}(\mathrm{m}_5(1,0,1,0,1), x_5, \mathrm{m}_5(1,0,1,0,0))), x_3, \\
&\mathrm{m}(\mathrm{m}(\mathrm{m}_5(1,0,0,1,1), x_5, \mathrm{m}_5(1,0,0,1,0)), x_4, \mathrm{m}(\mathrm{m}_5(1,0,0,0,1), x_5, \mathrm{m}_5(1,0,0,0,0))))), x_1, \\
&\mathrm{m}(\mathrm{m}(\mathrm{m}(\mathrm{m}_5(0,1,1,1,1), x_5, \mathrm{m}_5(0,1,1,1,0)), x_4, \mathrm{m}(\mathrm{m}_5(0,1,1,0,1), x_5, \mathrm{m}_5(0,1,1,0,0))), x_3, \\
&\mathrm{m}(\mathrm{m}(\mathrm{m}_5(0,1,0,1,1), x_5, \mathrm{m}_5(0,1,0,1,0)), x_4, \mathrm{m}(\mathrm{m}_5(0,1,0,0,1), x_5, \mathrm{m}_5(0,1,0,0,0))), x_2, \\
&\mathrm{m}(\mathrm{m}(\mathrm{m}_5(0,0,1,1,1), x_5, \mathrm{m}_5(0,0,1,1,0)), x_4, \mathrm{m}(\mathrm{m}_5(0,0,1,0,1), x_5, \mathrm{m}_5(0,0,1,0,0))), x_3, \\
&\mathrm{m}(\mathrm{m}(\mathrm{m}_5(0,0,0,1,1), x_5, \mathrm{m}_5(0,0,0,1,0)), x_4, \mathrm{m}(\mathrm{m}_5(0,0,0,0,1), x_5, \mathrm{m}_5(0,0,0,0,0))))) = \\
&\mathrm{m}(\mathrm{m}(\mathrm{m}(\mathrm{m}(1, x_5, 1), x_4, \mathrm{m}(1, x_5, 1)), x_3, \mathrm{m}(\mathrm{m}(1, x_5, 1), x_4, \\
&\mathrm{m}(1, x_5, 0))), x_2, \mathrm{m}(\mathrm{m}(\mathrm{m}(1, x_5, 1), x_4, \mathrm{m}(1, x_5, 0)), x_3, \\
&\mathrm{m}(\mathrm{m}(1, x_5, 0), x_4, \mathrm{m}(0, x_5, 0)))), x_1, \mathrm{m}(\mathrm{m}(\mathrm{m}(\mathrm{m}(1, x_5, 1), x_4, \\
&\mathrm{m}(1, x_5, 0)), x_3, \mathrm{m}(\mathrm{m}(1, x_5, 0), x_4, \mathrm{m}(0, x_5, 0))), x_2, \\
&\mathrm{m}(\mathrm{m}(\mathrm{m}(1, x_5, 0), x_4, \mathrm{m}(0, x_5, 0)), x_3, \mathrm{m}(\mathrm{m}(0, x_5, 0), x_4, \mathrm{m}(0, x_5, 0))))).
\end{aligned}
$$

FIGURE 5.2 – Application de l'algorithme 1 à la médiane à 5 entrées.

Cette représentation est d'ailleurs la plus petite, et unique modulo symétrie. Ce résultat peut être obtenu par la vérification exhaustive automatisée qu'aucune formule plus petite utilisant les mêmes variables ne représente m_5. ∎

Rappelons à présent une axiomatisation de la structure algébrique $\langle L, \mathrm{m}, 0, 1\rangle$, qui est l'ensemble L auquel on associe la fonction médiane ternaire ainsi que les fonctions d'arité 0, c'est-à-dire les constantes 0 et 1, par le système équationnel suivant. Cette axiomatisation a été donnée, par exemple, dans [Bir40], pour les algèbres Booléennes et pour les treillis distributifs en général.

Système 62.

$$
\begin{aligned}
&(M1)\ \mathrm{m}(x,y,z) = \mathrm{m}(x,z,y) = \mathrm{m}(z,x,y), && \text{(symétrie)}\\
&(M2)\ \mathrm{m}(x,x,y) = x, && \text{(majorité)}\\
&(M3)\ \mathrm{m}(\mathrm{m}(x,u,v),\mathrm{m}(y,u,v),z) = \mathrm{m}(\mathrm{m}(x,y,z),u,v), && \text{(distributivité)}\\
&(M4)\ \mathrm{m}(0,1,x) = x,
\end{aligned}
$$

pour tous $x, y, z, u, v \in L$.

Remarque 63. Une axiomatisation de l'algèbre Booléenne $\langle \{0,1\}, \wedge, \vee, \overline{\bullet}, 0, 1\rangle$ a été utilisée dans [AGDM14] pour la simplification de circuits Booléens ; nous avons décrit certains de leurs résultats et constructions dans le chapitre 4, section 4.4. Le Système 62 est une adaptation de ces résultats au cas des polynômes latticiels.

Afin de pouvoir manipuler librement des formules médianes, nous devons nous assurer que l'axiomatisation donnée dans le Système 62 est à la fois *correcte* et *complète*. La correction signifie que toute équation $s = t$ qui peut être dérivée du Système 62 est *valide*, c'est-à-dire que les formules s et t sont équivalentes. La complétude signifie que toute équation $s = t$ valide peut être dérivée à partir des axiomes du système. Ces propriétés de correction et de complétude sont importante pour la réécriture de termes ainsi que leur simplification. En effet, la correction assure que toute simplification effectuée par simplification des règles préserve l'équivalence logique, et la complétude assure qu'il est possible d'inférer tout terme médian de tout autre terme médian qui lui est équivalent en utilisant le Système 62.

Théorème 64. L'algèbre $\langle L, \mathrm{m}, \bot, \top\rangle$ munie des axiomes du Système 62 est correcte et complète.

Démonstration. La complétude est une conséquence immédiate du Théorème de Complétude de Birkhoff pour la logique équationnelle, décrit dans [Klo92, SB81]. Nous démontrons la correction de manière algébrique en dérivant les axiomes du système en utilisant les propriétés du treillis L ainsi que la définition algébrique de m donnée en Définition 56. Par exemple, la règle de symétrie $(M1)$ est une conséquence de la symétrie du meet $\wedge_L$ et du join $\vee_L$ de L. Soient $x, y, z, u, v \in L$. Par simplicité l'associativité de $\wedge_L$ et $\vee_L$ n'est pas indiquée quand elle est utilisée.

— Démontrons $(M1)$.

$$
\begin{aligned}
\mathrm{m}(x,y,z) &= (x \wedge_L y) \vee_L (y \wedge_L z) \vee_L (z \wedge_L x)\\
&= (z \wedge_L x) \vee_L (y \wedge_L z) \vee_L (x \wedge_L y) && \text{par symétrie de } \vee_L\\
&= (x \wedge_L z) \vee_L (z \wedge_L y) \vee_L (y \wedge_L x) && \text{par symétrie de } \wedge_L\\
&= \mathrm{m}(x,z,y).
\end{aligned}
$$

La seconde égalité se démontre de la même manière.

— Démontrons $(M2)$.

$$\begin{aligned}
\mathrm{m}(x,x,y) &= (x \wedge_L x) \vee_L (x \wedge_L y) \vee_L (y \wedge_L x) \\
&= x \vee_L (x \wedge_L y) \vee_L (y \wedge_L x) && \text{par idempotence de } \wedge_L \\
&= x \vee_L (x \wedge_L y) \vee_L (x \wedge_L y) && \text{par symétrie de } \wedge_L \\
&= x \vee_L (x \wedge_L y) && \text{par idempotence de } \vee_L.
\end{aligned}$$

— Démontrons $(M3)$.

$$\begin{aligned}
&\mathrm{m}(\mathrm{m}(x,u,v),\mathrm{m}(y,u,v),z) = \\
&(\mathrm{m}(x,u,v) \wedge_L \mathrm{m}(y,u,v)) \vee_L (\mathrm{m}(y,u,v) \wedge_L z) \vee_L (z \wedge_L \mathrm{m}(x,u,v)) \\
&= (((x \wedge_L u) \vee_L (u \wedge_L v) \vee_L (v \wedge_L x)) \wedge_L ((y \wedge_L u) \vee_L (u \wedge_L v) \vee_L (v \wedge_L y))) \\
&\vee_L (((y \wedge_L u) \vee_L (u \wedge_L v) \vee_L (v \wedge_L y)) \wedge_L z) \vee_L (z \wedge_L ((x \wedge_L u) \vee_L (u \wedge_L v) \\
&\vee_L (v \wedge_L x))) \\
&= (x \wedge_L y \wedge_L u) \vee_L (x \wedge_L y \wedge_L v) \vee_L (x \wedge_L u \wedge_L v \wedge_L y) \vee_L (u \wedge_L v \wedge_L y) \\
&\vee_L (u \wedge_L v) \vee_L (u \wedge_L v \wedge_L y) \vee_L (v \wedge_L x \wedge_L y \wedge_L u) \vee_L (u \wedge_L x \wedge_L v) \\
&\vee_L (x \wedge_L y \wedge_L v) \vee_L (y \wedge_L u \wedge_L z) \vee_L (u \wedge_L v \wedge_L z) \vee_L (v \wedge_L y \wedge_L z) \\
&\vee_L (x \wedge_L u \wedge_L z) \vee_L (u \wedge_L v \wedge_L z) \vee_L (v \wedge_L x \wedge_L z) \\
&= (x \wedge_L y \wedge_L u) \vee_L (x \wedge_L y \wedge_L v) \vee_L (u \wedge_L v) \vee_L (x \wedge_L y \wedge_L v) \vee_L (y \wedge_L u \wedge_L z) \\
&\vee_L (v \wedge_L y \wedge_L z) \vee_L (x \wedge_L u \wedge_L z) \vee_L (v \wedge_L x \wedge_L z) \qquad \text{par absorption} \\
&= (x \wedge_L y \wedge_L u) \vee_L (u \wedge_L v) \vee_L (x \wedge_L y \wedge_L v) \vee_L (y \wedge_L u \wedge_L z) \vee_L (v \wedge_L y \wedge_L z) \\
&\vee_L (x \wedge_L u \wedge_L z) \vee_L (v \wedge_L x \wedge_L z) \qquad \text{par idempotence} \\
&= (((x \wedge_L y) \vee_L (y \wedge_L z) \vee_L (z \wedge_L x)) \wedge_L u) \vee_L (u \wedge_L v) \vee_L (v \wedge_L ((x \wedge_L y) \\
&\vee_L (y \wedge_L z) \vee_L (z \wedge_L x))) \\
&= (\mathrm{m}(x,y,z) \wedge_L u) \vee_L (u \wedge_L v) \vee_L (v \wedge_L \mathrm{m}(x,y,z)) \\
&= \mathrm{m}(\mathrm{m}(x,y,z),u,v).
\end{aligned}$$

— Démontrons $(M4)$.

$$\begin{aligned}
\mathrm{m}(0,1,x) &= (0 \wedge_L 1) \vee_L (1 \wedge_L x) \vee_L (x \wedge_L 0) \\
&= 0 \vee_L x \vee_L 0 \\
&= 0 \vee_L x \\
&= x.
\end{aligned}$$

□

5.2 Formes normales médianes

Dans cette section, nous proposons une description structurelle des formules médianes et introduisons la notion de *forme normale médiane* (*MNF*, de l'anglais *Median Normal Form*), en adoptant le nom utilisé dans [CFL06, CLMW11] et par similarité avec les formes normales disjonctive, conjonctive, polynomiale, etc. Ces MNFs correspondent, comme nous allons le voir,

à des formules médianes qui sont « minimales » vis-à-vis de l'ordre lexicographique sur leurs descriptions structurelles.

Nous définissons un moyen de décrire de manière partielle la structure d'une formule médiane donnée, en comptant le nombre de médianes qui apparaissent à chaque niveau de la formule, vue comme un arbre. Cette description induit un bon ordre sur l'ensemble des formules médianes. Nous montrons par la suite que pour toute fonction polynomiale il existe un ensemble de plus petites représentations par rapport à ce bon ordre, que nous appelons les *formes normales médianes*, ou *MNF* (de l'anglais *Median Normal Forms*. De même que pour les représentations en forme normale conjonctive ou en forme normale disjonctive, cette représentation est unique modulo certaines propriétés telles que la symétrie ou l'associativité. En revanche, nous n'avons pas, pour l'instant, de description générale de la forme de ces représentations minimales. Ce problème constitue une intéressante piste de recherche future.

Définition 65. Soit ϕ une formule médiane de profondeur $d = \mathrm{d}(\phi)$. Rappelons que la profondeur d'une formule est définie récursivement par

$$\mathrm{d}(\mathrm{m}(\phi_1, \phi_2, \phi_3)) = 1 + \max_{i=1,2,3}\{\mathrm{d}(\phi_i)\}$$

et que la profondeur d'une variable ou d'une constance est 0.

Soit $n_0, \ldots, n_d$ la suite d'entiers naturels tels que pour tout $i \in \{0, 1, \ldots, d\}$, n_i est le nombre de médianes m aux profondeurs inférieures ou égales à i. La *représentation structurelle* de ϕ est le tuple

$$S_\phi = (n_d, \ldots, n_0).$$

Soit $\leq_S$ l'ordre sur les formules médianes défini par :

$$\phi_1 \leq_S \phi_2 \qquad \text{si} \qquad S_{\phi_1} \leq_{lex} S_{\phi_2}.$$

Exemple 66. La formule $\phi_1 = x$ a pour représentation structurelle

$$S_{\phi_1} = (0)$$

car elle est de profondeur 0 et ne fait intervenir aucune médiane. La formule $\phi_2 = \mathrm{m}(x, y, z)$ a pour représentation structurelle

$$S_{\phi_2} = (1, 0).$$

■

Fait 67. La suite $n_d, n_{d-1}, \ldots, n_0$ est une suite décroissante et $n_d = |\phi|$.

Remarque 68. L'ordre $\leq_S$ favorise en priorité la taille de la formule plutôt que sa profondeur. Par exemple, considérons les formules équivalentes suivantes :

$$\phi_1 = \mathrm{m}(x_1, x_2, \mathrm{m}(x_3, x_4, \mathrm{m}(x_5, x_6, x_7)))$$
$$\phi_2 = \mathrm{m}(\mathrm{m}(x_1, x_2, x_3), \mathrm{m}(x_1, x_2, x_4), \mathrm{m}(x_5, x_6, x_7)).$$

Ici, $|\phi_1| = 3 < 4 = |\phi_2|$, mais $\mathrm{d}(\phi_1) = 3 > 2 = \mathrm{d}(\phi_2)$. En examinant leurs représentations structurelles, nous avons :

$$S_{\phi_1} = (3, 2, 1, 0) \qquad \text{mais} \qquad S_{\phi_2} = (4, 3, 0);$$

donc $\phi_1 \leq_S \phi_2$.

Nous donnons à présent une définition d'une forme normale médiane comme une représentation médiane minimale.

Définition 69. Un terme médian ϕ est une *forme normale médiane* (MNF) si pour tout terme médian ϕ' tel que $\phi \equiv \phi'$, c'est-à-dire tel que ϕ et ϕ' sont équivalentes ou, plus explicitement, qu'elles représentent la même fonction, nous avons

$$\phi \leq_S \phi'.$$

Exemple 70. La formule

$$\phi = \mathrm{m}(\mathrm{m}(x, x, y), y, z)$$

n'est pas une forme normale médiane, puisque la formule

$$\phi' = \mathrm{m}(x, y, z)$$

y est équivalente, et que

$$S_{\phi'} = (1) \leq_{lex} (2, 1) = S_\phi.$$

L'équivalence des deux formules vient du fait que $\mathrm{m}(x, x, y) \equiv x$ d'après la règle $(M1)$ du Système 62. ■

Remarque 71. Comme nous l'avons défini, l'ordre structurel ne peut pas prendre en compte les symétries de la médiane dans les formules. Par exemple, les formules $\mathrm{m}(x, y, z)$ et $\mathrm{m}(x, z, y)$ ont le même tuple structurel $(1, 0)$, mais elles sont aussi équivalentes par application de $(M1)$, et sont toutes deux aussi des formes normales, puisqu'elles sont de taille minimale : aucune formule plus petite ne peut leur être équivalente. Une formule ne possède donc pas forcément une forme normale médiane unique, mais plutôt un *ensemble* de formes normales médianes. Par exemple, la formule ϕ de l'exemple 70 a pour ensemble de formes normales l'ensemble

$$\{\mathrm{m}(x, y, z), \mathrm{m}(x, z, y), \mathrm{m}(y, x, z), \mathrm{m}(y, z, x), \mathrm{m}(z, x, y), \mathrm{m}(z, y, x)\}.$$

5.3 Complexité de la simplification de formules

Dans cette section, nous adressons la question de la détermination de ces formes normales médianes. Pour cela, nous formalisons deux problèmes de décision qui expriment la tâche consistant à trouver des formules médianes dont les représentations structurelles sont minimales. Nous montrons qu'au plus ces deux problèmes sont modérément intractables, en ce sens qu'ils sont plus compliqués que des problèmes de P mais pas plus que des problèmes de Σ_2^{P}. et nous proposons un système de réécriture de termes pour obtenir des solutions approximatives à ces problèmes.

5.3.1 Formules structurellement plus petites

Bien que la définition des formes normales médianes soit exprimée de manière simple, une procédure qui convertirait une formule en entrée en une formule équivalente en forme normale médiane est sans doute intractable, c'est-à-dire non polynomiale. En effet, nous allons montrer que le fait même de vérifier si une formule est, ou non, une forme normale médiane semble être coûteux. Nous formalisons ce problème de décision dans la Définition 72 ci-dessous.

Définition 72. Considérons le problème de décision Monotone Smallmed :
Entrée : une formule médiane ϕ et une suite décroissante S ;

Sortie : RÉUSSITE s'il existe une formule ψ équivalente à ϕ et dont la représentation structurelle S_ψ est strictement plus petite que S. ÉCHEC si une telle formule n'existe pas.

Avant de démontrer que MONOTONE SMALLMED est intractable, nous majorons sa difficulté par Σ_2^{P}. Rappelons que la classe de complexité Σ_2^{P} est au second niveau de la hiérarchie polynomiale, entre NP et PSPACE [Sav98] ; voir la figure 3.11.

Théorème 73. MONOTONE SMALLMED appartient à la classe Σ_2^{P}.

Démonstration. Dans cette preuve nous utilisons une caractérisation adéquate de la classe Σ_2^{P} : celle-ci contient en effet les problèmes de décision

$$\{x \ : \ \exists c_1 \forall c_2 F(x, c_1, c_2)\}$$

dans lequel c_1 et c_2 sont des certificats de tailles polynomiales en la taille de x et dans lequel F est une fonction calculable en temps polynomial. Pour une description plus formelle de cette classe, nous nous référons au théorème 40. Considérons l'Algorithme 2, qui résout MONOTONE SMALLMED. Cet algorithme repose sur un test exhaustif de toutes les formules ψ plus petites, au sens structurel, qu'une formule ϕ donnée. L'assignation ς de l'Algorithme 2 fait référence à l'interprétation standard des variables et des symboles qui apparaissent dans la formule ϕ comme des variables Booléennes et des fonctions Booléennes. La taille du premier certificat ψ est polynomiale en la taille de l'entrée $|\phi|$ car sa représentation structurelle est majorée par celle de ϕ. La taille du second certificat ς est aussi polynomiale en la taille de l'entrée. Donc MONOTONE SMALLMED appartient à la classe Σ_2^{P}. □

Algorithm 2: Trouver une plus petite MNF équivalente.

Entrées: Une formule ϕ, une suite décroissante S.

Deviner de manière existentielle une formule ψ telle que $S(\psi) < S$;
  Deviner de manière universelle une assignation ς ;
    Vérifier que $\varsigma(\phi) = \varsigma(\psi)$;
    si oui, **alors retourner** RÉUSSITE;
si une telle formule n'existe pas, **alors retourner** ÉCHEC.

Notons que cet algorithme prend peu de place, comme illustré par la Figure 5.3.

Notons également que le Théorème 73 donne une borne supérieure asymptotique sur la complexité de MONOTONE SMALLMED ; toutefois, une borne inférieure similaire est encore en cours d'exploration.

Dans la Définition 72, la taille de la formule désirée est donnée en entrée par l'intermédiaire de la suite décroissante S. Si, maintenant, nous supposons une taille cible constante, c'est-à-dire si la suite décroissante est fixée au préalable, alors il est possible d'obtenir une meilleure borne de complexité.

Définition 74. Étant donnée une suite décroissante S fixée, considérons le problème de décision MONOTONE SMALLMED$_S$:

Entrée : une formule médiane ϕ ;

Sortie : RÉUSSITE s'il existe une formule ψ équivalente à ϕ et dont la représentation structurelle S_ψ est strictement plus petite que S. ÉCHEC si une telle formule n'existe pas.

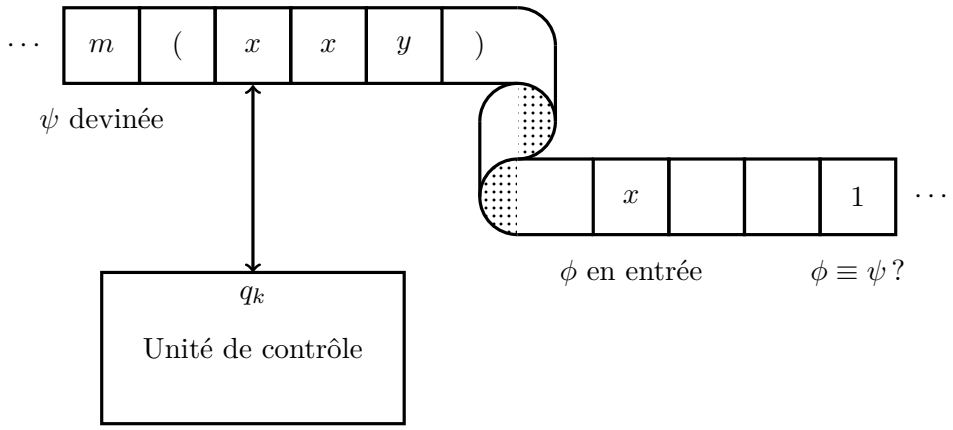


FIGURE 5.3 – Machine de Turing pour la recherche de plus petites formules.

Remarque 75. La définition 74 est indépendante de toute interprétation des formules médianes considérées, en ce sens que les objets manipulés sont syntactiques, c'est-à-dire des formules médianes et leurs structures. En fait, tous nos résultats peuvent être appliqués, par exemple, à la fois aux représentations des polynômes latticiels et des fonctions Booléennes monotones, puisque dans les deux domaines toute fonction peut être représentée par une formule médiane. Dans la Section 5.4 nous discutons de problèmes de décisions similaires mais, cette fois, liés à la représentation des fonctions Booléennes. Dans ce cas-ci, nous considérons un langage dans lequel les négations sont autorisées.

Théorème 76. Pour toute suite décroissante S, MONOTONE SMALLMED$_S$ est dans la classe coNP.

Démonstration. Soient s le premier élément de la suite S, n le nombre de variables qui apparaissent dans ϕ, et V_ϕ l'ensemble des variables qui apparaissent dans ϕ. Toute formule ψ qui a une représentation structurelle plus petite que celle de ϕ a au plus s médianes. Donc, une telle formule ψ ne peut pas contenir plus de $N = 2s + 1$ variables différentes, puisque la médiane est d'arité 3. Si ϕ est logiquement équivalente à une formule dont la représentation structurelle est plus petite que S, alors au plus N variables sont essentielles parmi celles qui apparaissent dans ϕ, en ce sens qu'elles ont un impact sur le comportement logique de ϕ : par exemple, dans l'expression $\mathrm{m}(x, x, y)$, la variable x est essentielle mais pas y car $\mathrm{m}(x, x, 1) \equiv \mathrm{m}(x, x, 0)$. Le Chapitre 3 contient une présentation plus formelle de la notion de variables essentielles.

Pour chaque sous-ensemble de variables $V \subset V_\phi$ de taille au plus N, il existe un nombre constant de formules ψ de représentation structurelle plus petite que S dont les variables sont prises dans V. L'algorithme suivant résout MONOTONE SMALLMED$_S$: pour chaque telle formule ψ, deviner de manière universelle une assignation ς pour les variables qui apparaissent dans ϕ et vérifier si $\varsigma(\phi) = \varsigma(\psi)$. Si tel est le cas, alors ψ et ϕ sont équivalentes et le résultat de l'algorithme est RÉUSSITE. Si aucune formule ne renvoie le résultat RÉUSSITE, alors nous pouvons arrêter l'algorithme avec le résultat ÉCHEC.

Le nombre de manières différentes de constituer l'ensemble V est borné par n^N, donc le nombre total d'essais universels est borné par $\mathcal{O}(n^N)$. Puisque N est constant, nous concluons que l'on peut déterminer si ϕ admet une formule équivalente de représentation structurelle plus petite que S en un nombre polynomial d'essais universels. Donc MONOTONE SMALLMED$_S$ est dans la classe co-NP. □

L'algorithme 2 ne retourne pas directement, à partir d'une formule en entrée, une MNF, mais peut toutefois être utilisé comme un sous-algorithme d'un autre algorithme qui, lui, retournerait une MNF. Soit ϕ une formule en entrée et S_ϕ sa représentation structurelle. Une recherche binaire nous permettrait d'identifier la plus petite représentation structurelle telle que l'algorithme 2 réussisse. La sortie du dernier appel à l'algorithme 2 est alors une MNF pour ϕ. Une recherche binaire effectue un nombre d'appels logarithmique en la taille du domaine ordonné de recherche. Dans notre cas, le domaine est l'ensemble des suites structurelles qui sont plus petites que S selon l'ordre lexicographique. Sa taille est au plus exponentielle en la taille de ϕ. Ainsi, la recherche binaire effectue un nombre d'appels à l'Algorithme 2 qui est polynomial en la taille de la formule en entrée.

5.3.2 Réécriture de termes

Orienter les règles du Système 62

Une implémentation naïve de l'Algorithme 2 consisterait en une recherche exhaustive de formules équivalentes à la formule d'entrée parmi toutes les formules structurellement plus petites. Une autre approche possible serait la recherche de formules structurellement plus petites parmi les formules équivalentes. Pour parcourir l'ensemble des formules équivalentes, nous utilisons les équations du Système 62 introduit ci-haut.

D'après le théorème 73, le système 62 est correct et complet. Autrement dit, pour toutes formules ϕ et ψ, celles-ci sont équivalentes si et seulement si il existe une réécriture depuis ϕ vers ψ par une séquence d'équations du système 62. Cette idée est implémentée dans l'algorithme 3.

Algorithm 3: Rapprocher une formule d'une MNF

Entrées: Une formule ϕ, une suite décroissante S.
Sorties: une formule ψ telle que $S_\psi < S$; l'algorithme diverge s'il n'en existe pas.

```
tant que φ ≥_S ψ faire
    Deviner de manière existentielle une réécriture par le Système 62 : ψ = ψ' ;
    Mettre à jour ψ ← ψ' ;
fintq
retourner ψ ;
```

Cet algorithme non-déterministe qui résout Monotone Smallmed utilise très peu de place : en effet, sa complexité en terme d'espace est linéaire en la taille de l'entrée. En fait, Monotone Smallmed est dans la classe NPSPACE, car l'algorithme 3 peut être implémenté par une machine de Turing prenant peu de place sur son ruban. Toutefois, celle-ci (et l'algorithme qu'elle implémente) ne s'arrête pas obligatoirement pour retourner un résultat.

D'après le théorème de Savitch (voir, par exemple, [Sav98]), nous savons que

$$\mathsf{NPSPACE} = \mathsf{PSPACE};$$

de plus, la classe Σ_2^{P} est contenue dans la classe PSPACE ; donc, le résultat du Théorème 73 est plus fort.

Nous ne savons pas si l'algorithme 3 termine toujours en temps polynomial, dans le meilleur cas où une formule structurellement plus petite que celle en entrée existe effectivement. En effet, la séquence de réécritures équationnelles nécessaires pour transformer une formule en une autre, plus petite, peut potentiellement être constituée d'un nombre exponentiel d'étapes. Par contre,

une preuve que l'algorithme 3 se termine toujours en temps polynomial constituerait une preuve que Monotone Smallmed est dans NP.

Une solution pour résoudre ce problème des explorations non-bornées lors de la recherche d'une dérivation est d'orienter les équations du système selon un ordre sur les formules : une formule est réécrite en une plus petite. En conséquence, l'application de n'importe quelle règle sur une formule, mise à part la règle de symétrie $(M1)$, la simplifiera selon l'ordre $\leq_S$: toute dérivation sera une simplification au sens large.

Considérons ainsi le système de réécriture de termes suivant, obtenu à partir du système 62 en orientant ses équations selon l'ordre structurel des formules qui apparaissent à gauche et à droite du symbole $\longrightarrow$.

Système 77.

$$
\begin{aligned}
&(R1)\ \mathrm{m}(x,y,z) = \mathrm{m}(x,z,y) = \mathrm{m}(z,x,y),\\
&(R2)\ \mathrm{m}(x,x,y) \longrightarrow x,\\
&(R3)\ \mathrm{m}(\mathrm{m}(x,u,v),\mathrm{m}(y,u,v),z) \longrightarrow \mathrm{m}(\mathrm{m}(x,y,z),u,v),\\
&(R4)\ \mathrm{m}(\bot,\top,x) \longrightarrow x,
\end{aligned}
$$

pour tous $x, y, z, u, v \in L$.

Nous adoptons la notation $(l \longrightarrow r)^{-1} := r \longrightarrow l$. Par exemple,

$$(R2)^{-1} = x \longrightarrow \mathrm{m}(x,x,y).$$

La règle de réécriture $(R1)$, qui n'est autre que la règle $(M1)$, est conservée telle quelle, sans orientation. De tels systèmes sont parfois appelés des *systèmes de réécriture modulo commutativité* [BN99]. On peut se trouver dans une situation similaire lorsque l'on manipule des systèmes qui mettent en jeu des opérations commutatives binaires telles que $\wedge_L$ ou $\vee_L$, auquel cas les règles de commutativité (de symétrie) ou d'associativité peuvent être conservées. Comme cela est expliqué dans [VM06], dans le contexte des termes médians, la commutativité $(M1)$ peut être orientée en définissant un ordre total sur les termes. Certains effets de cette orientation sont l'ordonnancement des termes grâce à l'application de ces nouvelles règles de commutativité orientées, ainsi que le blocage occasionnel de preuves de dérivations : dans ce cas, nous perdons ainsi la complétude du système.

Malgré ce désavantage, orienter les règles présente un certain intérêt car la recherche d'une dérivation entre formules est alors plus facile et les dérivations sont au plus de taille polynomiale. Un autre avantage conséquent est que le système est alors noethérien : toutes les dérivations sont de taille finie.

Lemme 78. Soient ϕ et ψ deux formules médianes. Si elle existe, la dérivation entre ϕ et ψ est polynomiale en la taille de ϕ.

Démonstration. Toutes les règles du système 77 exceptée la règle de symétrie $(R1)$ ont pour effet de retirer une médiane m de toute formule à laquelle elles sont appliquées. Si une telle dérivation existe, nécessairement, ϕ contient plus de médianes que ψ. La dérivation entre ϕ et ψ contient donc n étapes, avec n la différence entre le nombre de médianes dans ϕ et le nombre de médianes dans ψ. □

Par contre, il n'est pas toujours possible de réécrire toute formule en une formule équivalente en utilisant les règles du système 77, et encore moins en une forme normale médiane.

Exemple 79. Considérons la formule

$$\phi = \mathrm{m}(\mathrm{m}(\mathrm{m}(x, y, z), u, v), y, z).$$

ϕ a pour forme canonique

$$\phi' = \mathrm{m}(\mathrm{m}(x, u, v), y, z).$$

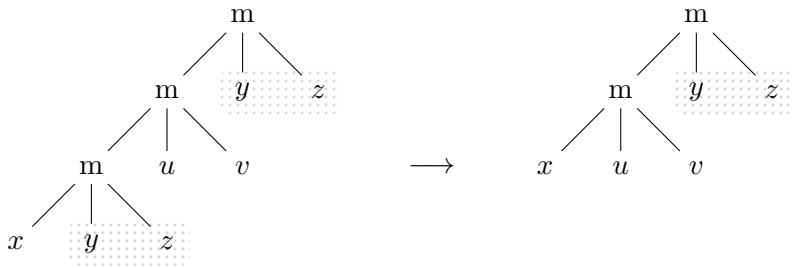
Il est possible de prouver que $\phi \equiv \phi'$ en dressant les tables de vérités de ϕ et ϕ'. Nous proposons dans cet exemple une dérivation explicite. Ensuite, nous démontrons que ϕ' est la plus petite formule équivalente à ϕ que l'on peut obtenir, modulo symétrie.

$$\begin{aligned}
\phi &= \mathrm{m}(\mathrm{m}(\mathrm{m}(x, y, z), u, v), y, z) \\
&\equiv \mathrm{m}(\mathrm{m}(x, \mathrm{m}(u, y, z), \mathrm{m}(v, y, z)), y, z) && \text{Règle } (R3)^{-1} \\
&\equiv \mathrm{m}(\mathrm{m}(\mathrm{m}(u, y, z), y, z), \mathrm{m}(\mathrm{m}(v, y, z), y, z), x) && \text{Règle } (R3)^{-1} \\
&\equiv \mathrm{m}(\mathrm{m}(u, y, z), \mathrm{m}(v, y, z), x) && \text{Règle } (R3) \\
&\equiv \mathrm{m}(\mathrm{m}(x, u, v), x, y) && \text{Règle } (R3) \\
&= \phi'.
\end{aligned}$$

Remarquons que certaines étapes de la dérivation nécessitent l'application de règles du Système 77 prises dans leur sens contraire. En fait, aucune règle de ce système prise dans son sens direct ne peut être directement appliquée à ϕ. Enfin, ϕ' ne peut être elle-même réécrite en une formule plus petite, même en inversant le sens des règles de système 77. En effet, il y a strictement plus que 3 variables qui sont essentielles dans ϕ' : il est donc impossible de trouver une formule médiane de taille 1 qui lui est équivalente. Donc, ϕ' est une forme normale médiane pour ϕ, qui ne peut pas être atteinte par le système 77. ■

L'exemple précédent illustre une importante source de complexité lors des dérivations qui font usage du système équationnel 62 : parfois, il est nécessaire d'augmenter temporairement la taille de la formule à transformer, afin de sortir d'un minimum local, de simplifier la formule encore plus, et d'atteindre ainsi une MNF. Cette augmentation est potentiellement non-bornée, comme nous l'indiquons dans la remarque 80.

Remarque 80. Nous pouvons généraliser la règle de l'exemple 79. Remarquons en effet la disposition particulière des variables dans cette règle, avec la répétition de la paire y, z dans la médiane à la racine et dans la médiane à la feuille de la formule vue comme un arbre.



Nous rappelons qu'un contexte est un terme qui contient une occurrence d'un symbole spécial $\diamond$ qui signifie un espace vide. Un contexte est généralement désigné par $C[\diamond]$. Si $t \in T_\tau(\Sigma)$ et que

t est substitué dans $\diamond$, alors le terme résultant est $C[t] \in T_\tau(\Sigma)$, et on dit que t est un *sous-terme* de $C[t]$. La règle généralisée est donc :

$$\mathrm{m}(C[\mathrm{m}(x,y,z)],y,z) \longrightarrow \mathrm{m}(C[x],y,z)$$

avec C un contexte quelconque. La preuve de cette règle repose sur une induction sur la profondeur du contexte C. Dans l'exemple 79, le contexte est donc

$$C[\diamond] = \mathrm{m}(\diamond,u,v).$$

De plus, la dérivation nécessite au moins k applications de $(R3)$, avec k la profondeur de C.

Bien que le système orienté ne soit ainsi plus complet, il est toujours correct.

Proposition 81. Le système 77 est correct mais pas complet.

Démonstration. La correction provient du fait que le système 62 est correct. L'incomplétude provient de l'exemple 79 : il est impossible de dériver la règle $\phi \longrightarrow \phi'$, c'est-à-dire la règle

$$\mathrm{m}(\mathrm{m}(\mathrm{m}(x,y,z),u,v),y,z) \longrightarrow \mathrm{m}(\mathrm{m}(x,u,v),y,z),$$

à partir des axiomes du système 77. □

Rajouter des règles au système de réécriture

Nous pourrions rajouter des règles au système 77, telle que la règle de l'exemple 79, pour obtenir le système 82.

Système 82.

$$\begin{aligned}
&(R1)\ \mathrm{m}(x,y,z) = \mathrm{m}(x,z,y) = \mathrm{m}(z,x,y),\\
&(R2)\ \mathrm{m}(x,x,y) \longrightarrow x,\\
&(R3)\ \mathrm{m}(\mathrm{m}(x,u,v),\mathrm{m}(y,u,v),) \longrightarrow \mathrm{m}(\mathrm{m}(x,y,z),u,v),\\
&(R4)\ \mathrm{m}(\bot,\top,x) \longrightarrow x,\\
&(R5)\ \mathrm{m}(\mathrm{m}(\mathrm{m}(x,y,z),u,v),y,z) \longrightarrow \mathrm{m}(\mathrm{m}(x,u,v),y,z),
\end{aligned}$$

pour tous $x,y,z,u,v \in L$.

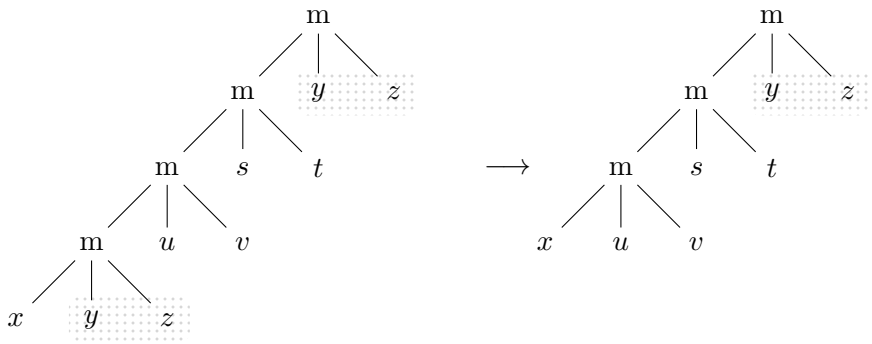
Puisque nous avons montré l'équivalence

$$\mathrm{m}(\mathrm{m}(\mathrm{m}(x,y,z),u,v),y,z) \equiv \mathrm{m}(\mathrm{m}(x,u,v),y,z),$$

le système 82 est toujours correct. Pourtant, il n'est toujours pas complet. Par exemple, la règle

$$\mathrm{m}(\mathrm{m}(\mathrm{m}(\mathrm{m}(x,y,z),u,v),s,t),y,z) \longrightarrow \mathrm{m}(\mathrm{m}(\mathrm{m}(x,u,v),s,t),y,z)$$

qui est un cas particulier de la règle généralisée $\mathrm{m}(C[\mathrm{m}(x,y,z)],y,z) \longrightarrow \mathrm{m}(C[x],y,z)$ et que l'on peut représenter, sous forme d'arbre, de la manière suivante :



ne peut être dérivée à partir des axiomes du système 82. En effet, on ne peut appliquer aucune règle de ce système à la partie gauche de la règle donnée ci-dessus, c'est-à-dire à

$$\mathrm{m}(\mathrm{m}(\mathrm{m}(\mathrm{m}(x,y,z),u,v),s,t),y,z),$$

et donc il n'est pas possible de réécrire cette formule en la formule

$$\mathrm{m}(\mathrm{m}(\mathrm{m}(x,u,v),s,t),y,z).$$

Par induction sur la structure des termes, nous avons la proposition suivante.

Proposition 83. Soit C un contexte. Alors, le système 77+(C), qui est le système 77 auquel nous avons rajouté la règle $\mathrm{m}(C[\mathrm{m}(x,y,z)],y,z) \longrightarrow \mathrm{m}(C[x],y,z)$, est correct mais non nécessairement complet.

En revanche, ajouter une famille infinie de règles pourrait rendre le système complet ; ce point est encore ouvert.

Conjecture 84. Soit l'ensemble de règles

$$C^* := \bigcup_{C \text{ un contexte}} \mathrm{m}(C[\mathrm{m}(x,y,z)],y,z) \longrightarrow \mathrm{m}(C[x],y,z).$$

Le système 77+(C^*) est correct et complet.

5.4 Extension au cas non-monotone

Dans cette section, nous étendons les résultats obtenus précédemment au cas non-monotone, ce qui revient à autoriser, dans nos formules, la négation. De ce fait, nous perdons le formalisme propre aux treillis, mais nous pouvons maintenant représenter toutes les fonctions Booléennes par des formules médianes. En effet, l'ensemble de connecteurs $\{\mathrm{m}, \overline{\bullet}\}$ est *complet* par rapport aux fonctions Booléennes, c'est-à-dire que toute fonction Booléenne a une représentation qui ne met en jeu que des connecteurs médians et la négation. Après une légère adaptation de la spécification équationnelle du système 62 pour inclure la négation, la plupart des définitions et des que nous obtenons sont similaires aux résultats déjà obtenus dans le cadre des fonctions Booléennes monotones et des polynômes latticiels. Nous rappelons ici la majeure partie du formalisme utilisé dans [CMPS19].

5.4.1 Spécification équationnelle

Nous ajoutons au système 62 deux axiomes afin de pouvoir manipuler les variables et les termes niés.

Système 85.

$$
\begin{aligned}
&(B1)\ \mathrm{m}(x,y,z) = \mathrm{m}(x,z,y) = \mathrm{m}(z,x,y),\\
&(B2)\ \mathrm{m}(x,x,y) = x,\\
&(B3)\ \mathrm{m}(\mathrm{m}(x,u,v),\mathrm{m}(y,u,v),) = \mathrm{m}(\mathrm{m}(x,y,z),u,v),\\
&(B4)\ \mathrm{m}(\overline{y},y,x) = x,\\
&(B\neg)\overline{\mathrm{m}(x,y,z)} = \mathrm{m}(\overline{x},\overline{y},\overline{z}),
\end{aligned}
$$

pour tous $x, y, z, u, v \in \{0, 1\}$.

Remarquons que l'on peut déduire la règle $(M4)$ du Système 62, c'est-à-dire la règle

$$\mathrm{m}(0, 1, x) = x,$$

à partir de la règle $(B4)$ du Système 85.

La règle de propagation $(B\neg)$, qui traduit la propriété d'*auto-dualité* de la médiane, nous permet, sans perdre de généralité, de ne considérer que des formules médianes dans lesquelles la négation n'apparaît qu'au niveau des variables : nous considérons donc les *littéraux*, c'est-à-dire les variables et les variables niées. En effet, étant donnée une formule ϕ, l'application répétée de la règle $(B\neg)$ au plus $|\phi|$ fois permet de propager toutes les négations qui apparaissent dans ϕ aux variables seules. Considérons la dérivation suivante en guise d'exemple :

$$
\begin{aligned}
\phi &= \overline{\mathrm{m}(\mathrm{m}(\overline{\mathrm{m}(x_1,x_2,x_3)},x_4,x_5),x_6,x_7)}\\
&\equiv \mathrm{m}(\mathrm{m}(\overline{\mathrm{m}(\overline{\mathrm{m}(x_1,x_2,x_3)},x_4,x_5)},\overline{x_6},\overline{x_7})\\
&\equiv \mathrm{m}(\mathrm{m}(\overline{\mathrm{m}(\overline{\mathrm{m}(x_1,x_2,x_3)},\overline{x_4},\overline{x_5})},\overline{x_6},\overline{x_7})\\
&\equiv \mathrm{m}(\mathrm{m}(\mathrm{m}(x_1,x_2,x_3),\overline{x_4},\overline{x_5}),\overline{x_6},\overline{x_7}).
\end{aligned}
$$

De même que pour les fonctions Booléennes monotones et les polynômes latticiels, nous avons besoin d'un système à la fois correct et complet pour réécrire toute formule médiane en toute autre qui lui est équivalente. En fait, le système 85 est correct et complet : ces propriétés viennent de la complétude et de la correction du système 62 pour les fonctions Booléennes monotones.

Théorème 86. L'algèbre $\langle \mathbb{B}, \mathrm{m}, \overline{\bullet}, 0, 1\rangle$ munie des axiomes du Système 85 est correcte et complète.

5.4.2 Obtenir une formule médiane dans le cas général

Dans cette section, nous discutons la conversion d'une fonction Booléenne quelconque, donc potentiellement non-monotone, en une formule médiane. Précédemment, nous avions reproduit l'algorithme 1 de [CLMW11] qui renvoie une formule médiane étant donnée la table de vérité d'une fonction Booléenne *monotone*. Nous examinons à présent le cas général.

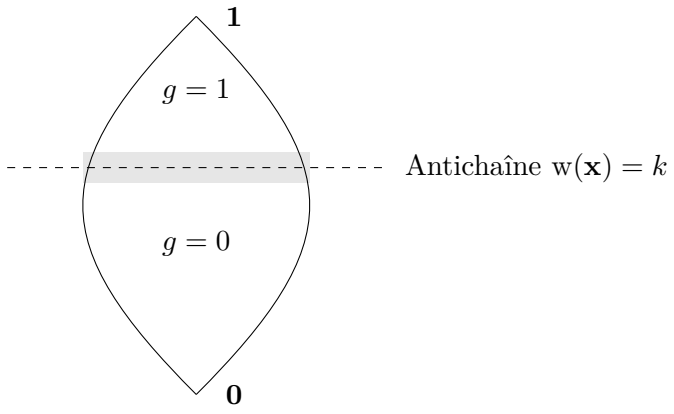


FIGURE 5.4 – Diagramme de Hasse d'une fonction à k-tranche.

Fonctions à tranche (Slice functions)

Dans cette section, nous adoptons les notations de [CLMW11, Juk12]. Une fonction Booléenne g est une *fonction à k-tranche* si $g(\mathbf{x}) = 0$ lorsque $\mathrm{w}(\mathbf{x}) < k$ et si $g(\mathbf{x}) = 1$ lorsque $\mathrm{w}(\mathbf{x}) > k$. Les fonctions à tranche sont monotones ; de plus, elle ne sont non-triviales que sur l'antichaîne de poids k. Dans la figure 5.4, nous avons représenté de manière schématique un diagramme de Hasse d'une telle fonction à tranche g.

Algorithme de conversion

Soit $f\colon \mathbb{B}^n \to \mathbb{B}$ une fonction Booléenne quelconque. Afin d'appliquer l'algorithme 1, nous construisons une fonction à $n-$*tranche* Booléenne monotone $g_f\colon \mathbb{B}^{2n} \to \mathbb{B}$ de telle sorte que l'on peut aisément retrouver f par une certaine substitution de variables pour des variables niées. Ainsi, nous pouvons appliquer l'algorithme 1 à g_f pour en obtenir une formule médiane, puis rétablir l'équivalence à f en appliquant la substitution mentionnée ci-dessus.

Soient $\mathbf{a} = (a_1, \ldots, a_{2n}) \in \mathbb{B}^{2n}$, $\mathbf{b} = (a_1, \ldots, a_n)$, $\mathbf{c} = (a_{n+1}, \ldots, a_{2n})$, et

$$g_f \colon = \begin{cases} 0 & \text{si } \mathrm{w}(\mathbf{a}) < n, \\ 1 & \text{si } \mathrm{w}(\mathbf{a}) > n, \\ f(\mathbf{b}) & \text{si } \mathbf{b} = \overline{\mathbf{c}}, \\ 0 & \text{sinon.} \end{cases} \tag{5.2}$$

Par construction, la fonction g_f est monotone, et pour tout $\mathbf{b} \in \mathbb{B}^n$,

$$f(\mathbf{b}) = g_f(\mathbf{b}, \overline{\mathbf{b}}),$$

et ainsi

$$f(x_1, \ldots, x_n) = g_f(x_1, \ldots, x_n, \overline{x_1}, \ldots, \overline{x_n}).$$

Exemple : conversion de la somme modulo 2

Soit $f(x, y) \colon = x \oplus y$. Puisque f n'est pas monotone, il nous faut utiliser l'algorithme 4. La fonction g_f construite comme nous l'avons détaillé ci-dessus est définie par la table de vérité 5.1. Nous donnons également un diagramme de Hasse de g_f dans la figure 5.5. La monotonie de

Algorithm 4: genMNF (*general Median Normal Form*) - forme normale médiane pour les fonctions Booléennes en général (d'après [CLMW11])

Entrée: Une fonction Booléenne f.
Sorties: Une formule médiane qui représente f.

si f est monotone **alors**
  retourner MMNF(f) ;
sinon
  Construire g_f comme décrit dans (5.2) ;
  $w \leftarrow$ MMNF(g_f) ;
  pour $i = 1$ à n **faire**
    Remplacer chaque occurrence de x_{n+i} dans w par $\overline{x_i}$;
retourner w ;

x_1	x_2	x_3	x_4	$g_f(x_1, x_2, x_3, x_4)$
0	0	0	0	0
		⋮		0
0	0	1	1	0
0	1	1	0	1
1	0	0	1	1
1	1	0	0	0
		⋮		1
1	1	1	1	1

TABLE 5.1 – Table de vérité de g_f pour $f = \oplus$.

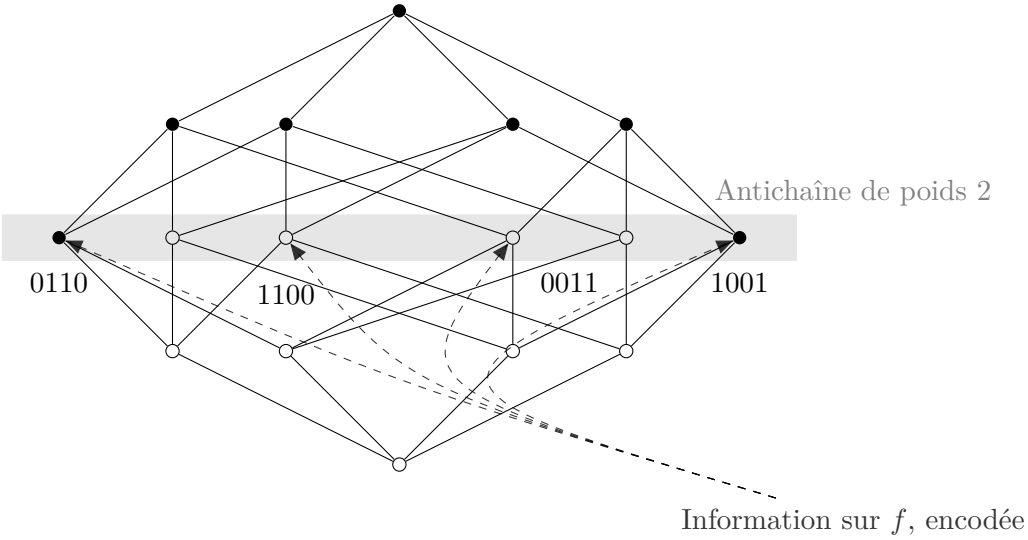


FIGURE 5.5 – Diagramme de Hasse pour la fonction g_f.

g_f y est immédiatement visible, ainsi que le fait que g_f est une fonction à 2-tranche. Toute l'information de f est encodée dans la partie grisée du diagramme.

Après application de l'algorithme 1 sur la fonction monotone g_f, nous en obtenons la représentation w suivante :

$$\begin{aligned} w =& \mathtt{m}(\mathtt{m}(\mathtt{m}(\mathtt{m}(0,x_1,0),x_2,\mathtt{m}(0,x_1,0)),x_3,\mathtt{m}(\mathtt{m}(0,x_1,0),x_2,\mathtt{m}(1,x_1,1))),x_4,\\ & \mathtt{m}(\mathtt{m}(\mathtt{m}(0,x_1,1),x_2,\mathtt{m}(0,x_1,1)),x_3,\mathtt{m}(\mathtt{m}(0,x_1,1),x_2,\mathtt{m}(1,x_1,1)))) \end{aligned}$$

puis, après simplification et remplacement de x_3 par $\overline{x_1}$ et de x_4 par $\overline{x_2}$:

$$\mathtt{m}(\mathtt{m}(0,\overline{x_1},x_2),\overline{x_2},\mathtt{m}(x_1,x_2,1))$$

qui est une formule médiane qui représente la somme $\oplus$. Dans [Juk12], Jukna mentionne une construction alternative de g_f, due à Lipton [Lip10], et discute de la complexité des *circuits* qui représentent ces fonctions.

5.4.3 MNFs Booléennes et complexité

Les MNFs dans le cas non-monotone

Dans cette section, nous adaptons les notions que nous avons introduites en section 5.2 dans le but de manipuler des représentations de fonctions Booléennes. Puisque la terminologie est exactement la même que dans le cas des fonctions Booléennes monotones, nous la conservons.

Définition 87. Nous désignons l'ensemble des *termes médians* (*formules médianes*, *expressions médianes*) par **M**. C'est l'ensemble de toutes les formules que l'on peut construire en utilisant des constantes, des variables, des variables niées et la médiane m. Étant donné un terme médian ϕ de profondeur d, soient $n_0,\ldots,n_d$ une suite d'entiers naturels tels que pour tout $i \in \{0\} \cup [d]$, n_i est le nombre de médianes dans ϕ à la profondeur i. La *représentation structurelle* de ϕ est le tuple

$$S_\phi = (n_d,\ldots,n_0).$$

De même que pour les fonctions monotones, nous pouvons comparer les termes médians grâce à un ordre $\leq_S$:

$$\phi_1 \leq_S \phi_2 \qquad \text{si} \qquad S_{\phi_1} \leq_{lex} S_{\phi_2}.$$

Puisque cet ordre est un bon ordre sur un ensemble fini de formules, il est possible, étant donné un terme médian, de considérer son ensemble de formes normales médianes.

Définition 88. Un terme médian ϕ est dit sous *forme normale médiane* (MNF) si pour tout terme médian ϕ' tel que $\phi \equiv \phi'$, on a

$$\phi \leq_S \phi'.$$

Étant donnés nos résultats précédents sur les fonctions Booléennes monotones, une question naturelle peut être posée. Soit f une fonction Booléenne monotone et ϕ une formule en MNF pour f. Le fait d'ajouter les négations dans notre langage nous permet-il de donner une représentation de f en forme normale médiane plus petite que ϕ ?

Directement, étant données deux représentations minimales ϕ et $\phi_\neg$ de la même fonction, dans lesquelles les négations les négations ne sont respectivement pas autorisées et le sont, alors

$$|\phi_\neg| \leq |\phi|$$

avec égalité si $\phi_\neg = \phi$.

Par contre, de manière surprenante, dans certains cas la comparaison peut être stricte. Autrement dit, certaines fonctions monotones peuvent avoir des représentations qui sont strictement plus petites lorsque l'on autorise la négation dans notre langage que lorsque l'on ne l'autorise pas.

Proposition 89. Il existe une fonction Booléenne monotone f telle que

$$|\phi_\neg| < |\phi|,$$

avec $\phi_\neg$ et ϕ les représentations minimales de f avec et sans négations respectivement.

Démonstration. Considérons les deux formules médianes

$$\begin{aligned}\phi &= \mathrm{m}(y, \mathrm{m}(u, v, t), \mathrm{m}(x, z, \mathrm{m}(u, v, t))),\\ \phi_\neg &= \mathrm{m}(x, \mathrm{m}(\overline{x}, y, z), \mathrm{m}(u, v, t)).\end{aligned}$$

Remarquons que $|\phi| = 4$ et $|\phi_\neg| = 3$. $\phi_\neg$ contient une variable niée, ce qui peut sembler surprenant puisqu'elle représente une fonction monotone : pourtant, nous avons effectivement $\phi_\neg \equiv \phi$, ce que l'on peut prouver en dressant les tables de vérité de ces deux formules, ou par une dérivation explicite que nous produisons ici.

$$\begin{aligned}\phi_\neg &= \mathrm{m}(x, \mathrm{m}(\overline{x}, y, z), \mathrm{m}(u, v, t))\\ &\equiv \mathrm{m}(y, \mathrm{m}(\mathrm{m}(u, v, t), x, \overline{x}), \mathrm{m}(x, z, \mathrm{m}(u, v, t))) && \text{Règle } (B3)^{-1}\\ &\equiv \mathrm{m}(y, \mathrm{m}(u, v, t), \mathrm{m}(x, z, \mathrm{m}(u, v, t))) && \text{Règle } (B_\neg)\\ &= \phi.\end{aligned}$$

Nous démontrons à présent que ϕ est la plus petite représentation sans négations et modulo permutations de variables équivalente à $\phi_\neg$. Puisque l'espace des formules plus petites que ϕ est fini, nous avons qu'il n'existe pas de formule ϕ' sans négation et de taille strictement inférieure à ϕ équivalente à ϕ grâce à une exploration exhaustive automatisée.

Nous donnons ici une preuve formelle de ce résultat. Supposons qu'il existe une formule ϕ' qui ne contient pas de négation et qui est strictement plus petite que ϕ. L'idée derrière la preuve (par l'absurde) est d'exhiber des tuples, parfois appelés *assignations* dans cette preuve, X telles que $\phi'(X) \neq \phi_{\neg}(X)$. Essentiellement, nous explorons toutes les possibilités pour ϕ' de manière plus méthodique qu'avec une recherche exhaustive.

Nécessairement, $|\phi'| = 3$ parce que toutes les variables sont essentielles dans ϕ. Son arité essentielle est donc 6, mais les formules médianes de taille 2 ont, au plus, une arité essentielle de 5. Sans perdre de généralité, ϕ' peut avoir deux structures différentes, séparés en cas (a) et cas (b) ci-dessous. De plus, ϕ' peut soit contenir deux variables qui sont répétées, ou une constantes et des variables non répétées. Dans le reste de la preuve, nous allons produire des assignations pour lesquelles ϕ' et $\phi_{\neg}$ prennent des valeurs différentes. Considérons les assignations suivantes :

— $X : x, y, u = 1$ et $v, t, z = 0$;
— $Y : u, y, z = 1$ et $v, t, x = 0$;
— $Z : v, u, t = 1$ et $x, y, z = 0$,

qui sont des points faux pour $\phi_{\neg}$, c'est-à-dire que

$$\phi_{\neg}(X) = \phi_{\neg}(Y) = \phi_{\neg}(Z) = 0.$$

1. Premièrement, supposons qu'il n'y ait pas de constantes dans ϕ'.
   (a) Supposons que ϕ' soit de la forme

   $$\phi' = \mathrm{m}(x_1, \mathrm{m}(x_2, x_3, x_4), \mathrm{m}(x_5, x_6, x_7)).$$

   Sans constantes dans ϕ, il y a une variable répétée. Si $x_2 = x_3$, ou si $x_2 = x_4$, alors ϕ' peut être simplifiée en $\mathrm{m}(x_1, x_2, \mathrm{m}(x_5, x_6, x_7))$ grâce à la règle de majorité $(B2)$, qui est d'arité essentielle au plus 5 et ne peut donc pas être équivalente à ϕ'. Un raisonnement similaire nous assure que, sans perdre de généralité puisque la médiane est symétrique, soit $x_1 = x_2$, soit $x_2 = x_5$.
      i. Supposons que $x_1 = x_2$.
         A. Si x est la variable répétée, alors selon l'assignation X nous avons $\phi'(X) = 1$.
         B. Si y est la variable répétée, ou, de manière équivalente, si z est la variable répétée, alors selon l'assignation X nous avons $\phi'(X) = 1$.
         C. Si u est la variable répétée, ou, de manière équivalente, v ou t, alors selon l'assignation X nous avons $\phi'(X) = 1$.
      ii. Supposons que $x_2 = x_5$.
         A. Si x est la variable répétée, alors selon l'assignation X nous avons $\phi'(X) = 1$.
         B. Si y est la variable répétée, ou, de manière équivalente, si z est la variable répétée, alors si $x = x_1$ alors $\phi'(X) = 1$; sinon, si $u = x_1$ alors $\phi'(X) = 1$; sinon, si $z = x_1$ alors $\phi_{\neg}(Y) = 0$ et $\phi'(Y) = 1$.
         C. Si u est la variable répétée, ou, de manière équivalente, si v ou t le sont, alors si $x = x_1$ ou $y = x_1$ alors $\phi'(X) = 1$; sinon, si $d = x_1$ alors $\phi_{\neg}(Z) = 0$ et $\phi'(Z) = 1$.
   (b) Supposons que ϕ' soit de la forme

   $$\mathrm{m}(\mathrm{m}(\mathrm{m}(x_1, x_2, x_3), x_4, x_5), x_6, x_7).$$

   Le raisonnement est similaire à celui ci-dessus. Les possibilités pour les variables répétées sont, sans perte de généralité, $x_1 = x_3$, $x_1 = x_5$, ou $x_3 = x_5$.

i. Supposons que $x_1 = x_3$.

A. Si x est répété ($x = x_1$), X est une assignation adéquate, qui produit une absurdité : $\phi'(X) \neq \phi_\neg(X)$.

B. Si y est répété, alors X est une assignation adéquate.

C. Si u est répété, alors X est une assignation adéquate.

ii. Supposons que $x_1 = x_5$.

A. Si x est répété, alors X est une assignation adéquate.

B. Si y est répété, alors si $x = x_3$ ou $u = x_3$, X est une assignation adéquate ; si $z = x_3$ alors Y est une assignation adéquate.

C. Si u est répété, alors si $x = x_3$ ou $y = x_3$, X est une assignation adéquate ; sinon, si $z = x_3$ alors Y est une assignation adéquate.

iii. Supposons que $x_3 = x_5$.

A. Si x est répété, alors X convient.

B. Si y est répété, alors si $x = x_7$ ou $u = x_7$ alors X convient ; sinon, si $z = x_7$ alors Y convient.

C. Si u est répété, alors X convient.

2. Secondement, supposons qu'il existe une constante dans ϕ', et qu'en particulier aucune variable ne soit répétée.

(a) Supposons que

$$\phi' = \mathrm{m}(x_1, \mathrm{m}(x_2, x_3, x_4), \mathrm{m}(x_5, x_6, x_7)),$$

telle que l'un des $x_1, \ldots, x_7$ soit une constante.

i. Si cette constante est 1, alors tous les tuples de poids au moins 3 sont des points vrais points de ϕ', mais il existe des faux points de $\phi_\neg$ de poids 3 : par exemple, X, Y, et Z.

ii. Supposons que cette constante soit 0.

A. Si $x_1 = 0$, alors $\phi' \equiv \mathrm{m}(x_2, x_3, x_4) \wedge_L \mathrm{m}(x_5, x_6, x_7)$ parce que $\mathrm{m}(a, 0, b) \equiv a \wedge_L b$. Tout tuple de poids 3 est un point faux pour cette fonction, mais il existe des points vrais de $\phi_\neg$ de poids 3.

B. Supposons que $x_2 = 0$ (le raisonnement est le même pour les autres cas). Si $x_1 = x$ alors l'assignation $x, y, u = 0$, $z, v, t = 1$ est un point vrai pour $\phi_\neg$ mais pas pour ϕ'. La même assignation convient pour les cas $x_1 = y, z$ et $x_1 = u, v, t$.

(b) Supposons que

$$\phi' = \mathrm{m}(\mathrm{m}(\mathrm{m}(x_1, x_2, x_3), x_4, x_5), x_6, x_7),$$

telle que l'un des $x_1, \ldots, x_7$ soit une constante.

i. Supposons que cette constante soit 1. Remarquons qu'il n'existe pas de tuple X' de poids 2 tel que $\phi_\neg(X') = 1$; toutefois, quelle que soit la position de la constante 1 dans ϕ', c'est-à-dire $x_7 = 1$ ou $x_5 = 1$ ou $x_3 = 1$, il est possible de trouver des assignations X' de poids 2 telles que $\phi'(X') = 1$. Par exemple, si $x_7 = 1$, l'assignation $x_4, x_5 = 1$ et $x_i = 0$, pour $i \neq 4, 5$ convient.

ii. Supposons que cette constante soit 0. Remarquons qu'il n'existe pas de tuple X' de poids 4 tel que $\phi_\neg(X') = 0$; toutefois, quelle que soit la position de la constante 0 dans ϕ', c'est-à-dire $x_7 = 0$ ou $x_5 = 0$ ou $x_3 = 0$, il est possible de trouver des assignations X' de poids 4 telles que $\phi'(X') = 0$.

Ainsi, il n'existe pas de formule médiane ϕ' strictement plus petite que ϕ et telle que $\phi' \equiv \phi$ et telle que ϕ' ne contient pas de variables niées. □

Ce résultat nous amène à explorer de nouveaux axes de recherche, tel que le problème de la recherche d'une description générale des formes normales médianes valable à la fois pour les fonctions monotones et les fonctions non-monotones, qui reste ouvert.

Remarque 90 (Circuits)**.** La taille des circuits Booléens dits *monotones*, c'est-à-dire construits sur la base monotone $\{\vee, \wedge\}$, a été étudiée, en particulier, par Razborov ([Raz85a, Raz85b]) qui établit une borne inférieure exponentielle pour la taille monotone de certains circuits, en réponse à la conjecture de Pratt ([Pra74]) sur la différence de complexité des circuits construits sur les bases $\{\vee, \wedge, \neg\}$ et $\{\vee, \wedge\}$: il arrive qu'ajouter la négation, même pour représenter des fonctions monotones, donne lieu à un gain de taille. Les études citées portent plus précisément sur des problèmes d'algèbre linéaire et de multiplication de matrices Booléennes (c'est-à-dire à coefficients Booléens) ; ainsi, le *permanent logique* est une valeur scalaire d'une matrice carrée $A = (a_{ij})$ de taille $n \times n$, tout comme le déterminant, et est défini par

$$Per(A) := \sum_{\pi \in \mathfrak{S}(n)} \prod_{i=1}^{n} a_{i\pi(i)}.$$

avec $\mathfrak{S}(n)$ l'ensemble des permutations $n \to n$. Remarquons la similarité avec le déterminant d'une matrice carrée, qui, là, fait intervenir la parité des permutations σ. C'est un problème #P-complet dans le cas Booléen ([Val79]), classe de complexité qui correspond aux problèmes de comptage de certificats acceptants (voir la section 3.5).

Problèmes de décision correspondants

Une autre question intéressante consiste à déterminer si les problèmes de décision dans le cas général, c'est-à-dire les problèmes Smallmed et Smallmed$_S$ qui correspondent respectivement aux problèmes Monotone Smallmed et Monotone Smallmed$_S$ dans le cas monotone, appartiennent aux mêmes classes de complexité.

Définition 91. Considérons le problème de décision Smallmed :

Entrée : une formule médiane ϕ (qui représente une fonction Booléenne) et une suite décroissante S ;

Sortie : Réussite s'il existe une formule ψ équivalente à ϕ et dont la représentation structurelle S_ψ est strictement plus petite que S. Échec si une telle formule n'existe pas.

Théorème 92. Smallmed est dans la classe Σ_2^{P}.

Démonstration. La preuve est la même que pour Monotone Smallmed ; voir le théorème 73. Remarquons que l'algorithme 2 résout aussi Smallmed. □

Remarque 93. La définition de Smallmed est indépendante des objets qui sont représentés, c'est-à-dire, dans ce cas, les fonctions Booléennes. Cela explique l'adaptation presque directe des notions et des résultats.

Décider si deux formules sont équivalentes en utilisant les règles du Système 85, c'est-à-dire en trouvant une dérivation explicite entre les deux formules, reste un problème difficile. Pour illustrer cela, remarquons que pour démontrer la règle de réécriture

$$\mathrm{m}(\mathrm{m}(\mathrm{m}(x, y, z), u, v), y, z) \longrightarrow \mathrm{m}(\mathrm{m}(x, u, v), x, y),$$

en réécrivant la partie gauche en la partie droite grâce aux axiomes du système 85, nous devons obligatoirement appliquer une règle qui augmente la taille de la formule ; voir l'exemple 79.

De même que pour les fonctions Booléennes monotones dans la section 5.3.2, nous pouvons considérer un système de réécriture de termes extrait du système 85 en en orientant les équations dans le but de simplifier la recherche de dérivations entre formules.

Système 94.

$$
\begin{aligned}
&(R1)\, \mathrm{m}(x,y,z) = \mathrm{m}(x,z,y) = \mathrm{m}(z,x,y),\\
&(R2)\, \mathrm{m}(x,x,y) \longrightarrow x,\\
&(R3)\, \mathrm{m}(\mathrm{m}(x,u,v),\mathrm{m}(y,u,v),) \longrightarrow \mathrm{m}(\mathrm{m}(x,y,z),u,v),\\
&(R4)\, \mathrm{m}(\overline{y},y,x) \longrightarrow x,\\
&(R\neg)\overline{\mathrm{m}(x,y,z)} \longrightarrow \mathrm{m}(\overline{x},\overline{y},\overline{z}),
\end{aligned}
$$

pour tous $x, y, z, u, v \in \{0, 1\}$.

Bien que le système 94 ne soit plus complet, il reste correct, tout comme le système 77.

5.5 Conclusion

Dans ce chapitre, nous nous sommes concentrés sur les règles de calculs qui régissent la réécriture de formules médianes, c'est-à-dire de formules construites en utilisant le connecteur `m`, des variables et des variables niées, et des constantes. Nous donnons une spécification équationnelle (système 62) correcte et complète qui nous permet de réécrire toute formule en toute autre qui lui est équivalente, ce qui peut être mis à profit pour, entre autres, la simplification de formules. Nous proposons une définition pour les formes normales médianes, qui sont des formules médianes minimales selon un certain ordre structurel sur les formules, par analogie avec les formes normales disjonctives, conjonctives, et polynomiales.

Nous discutons la réécriture de formules en orientant les règles de la spécification équationnelle, et proposons différents systèmes qui ont l'avantage de garantir un nombre de réécritures borné pour chaque réécriture, mais ont l'inconvénient de ne plus être complètes : certaines formules ne peuvent pas être simplifiées. Nous discutons également des systèmes infinis de règles de réécritures.

Nous étudions par la suite la complexité de problèmes de décisions tels que celui qui consiste à déterminer si une formule est minimale ou non. Nous démontrons que ce problème est dans Σ_2^{P}, c'est-à-dire au second niveau de la Hiérarchie Polynomiale : c'est une problème modérément intractable, en ce sens qu'il est sans doute plus difficile à résoudre qu'un problème de NP. Une piste de recherche possible est de préciser la difficulté de ce problème encore plus, en déterminant par exemple sa Σ_2^{P}-difficulté.

Nous étendons les résultats précédents au cas Booléen, en travaillant cette fois sur $\mathbb{B} = \{0, 1\}$ et en permettant le connecteur de négation $\neg$ dans nos formules. Malgré des difficultés identiques pour les problèmes de minimisation correspondants, nous obtenons un résultat surprenant, qui est que parfois, et pour représenter des fonctions monotones, permettre la négation dans nos formules médianes permet d'atteindre des représentations strictement plus petites.

Nous discutons des travaux futurs et des questions soulevées durant cette étude dans le chapitre 7.

Dans le chapitre suivant, nous continuons à travailler sur $\mathbb{B}$. Motivés par les résultats d'efficacité de la médiane (voir la figure 6.3), nous examinons d'autres systèmes de représentation

similaires aux formes normales disjonctives, conjonctives, polynomiales et polynomiales duales, et qui dépendent d'autres connecteurs tels que le NAND $\uparrow$, la médiane généralisée à $2n+1$ entrées $\mathtt{m}_{2n+1}$, ou encore le connecteur de Shannon $\mathtt{s}$, inspiré de la décomposition de Shannon des fonctions Booléennes :

$$f(x_1, \dots x_n) = (x_1 \wedge f(1, x_2, \dots, x_n)) \vee (\overline{x_1} \wedge f(0, x_2, \dots, x_n)).$$

6

Systèmes de Formes Normales

> La classe, c'est l'état permanent ; la forme, c'est l'état présent. La forme est la perfection de la classe.
> Étrange destinée que celle du mot forme. *Forma*, c'est d'abord le creuset où l'on coule la statue de bronze, c'est ensuite le galbe de la statue, puis c'en est la beauté. Le mot forme comporte un idéal de perfection.
>
> *Être chic*, Émile Moussat ([Mou37]).

Sommaire

Dans ce chapitre, nous nous concentrons sur la représentation des fonctions Booléennes par des *Systèmes de Formes Normales* (ou NFSs, de l'anglais $NormalFormSystem$) plus généraux que dans le Chapitre 5 dans lequel nous n'avons pris que la médiane et la négation comme connecteurs. De manière très générale, un NFS est vu comme un ensemble de termes construits à partir d'un ensemble donné de connecteurs, dont l'occurrence dans les termes est contrainte.

Cette contrainte est à rapprocher de celle que suivent les formes normales plus connues telles que la forme normale conjonctive ou disjonctive. Nous rappelons par exemple que dans une formule de logique propositionnelle mise sous forme conjonctive telle que

$$(x_1 \vee x_2) \wedge (x_2 \vee x_3) \wedge (x_3 \vee x_1),$$

les conjonctions de disjonctions sont autorisées, mais pas les disjonctions de conjonctions. De manière similaire, dans nos NFSs, l'occurrence des connecteurs est « figé » en ce sens que nous définissons au préalable un ordre d'apparition de ces connecteurs. Dans le cas de la forme normale conjonctive, cet ordre d'apparition est la conjonction $\wedge$, puis la disjonction $\vee$; pour la forme normale disjonctive, $\vee$ puis $\wedge$; pour la forme normale polynomiale, $\oplus$ puis $\wedge$; etc.

Étant donné un NFS fixé $\mathbf{A}$, la complexité d'une fonction Booléenne f dans $\mathbf{A}$ est le minimum des tailles des termes, dans $\mathbf{A}$, qui représentent f. Cela induit un préordre sur les NFSs : un NFS $\mathbf{A}$ est *polynomialement aussi efficace* qu'un NFS $\mathbf{B}$ s'il existe un polynôme P à coefficients entiers positifs tels que la complexité de toute fonction Booléenne f dans $\mathbf{A}$ est, au plus, la valeur de P prise en la complexité de f dans $\mathbf{B}$. Dans ce chapitre, nous étudions les NFSs monotones, c'est-à-dire les NFSs dont les connecteurs sont croissants ou décroissants en chacun de leurs arguments. Nous décrivons les NFSs monotones qui sont optimaux, c'est-à-dire qui sont minimaux selon l'ordre décrit ci-haut. Nous montrons que ces NFSs monotones minimaux sont tous équivalents. De plus, nous adressons des questions naturelles : l'optimalité dépend-elle de l'arité des connecteurs ? Dépend-elle du nombre de connecteurs utilisés ? Nous démontrons que les NFSs monotones optimaux sont exactement ceux qui ne mettent en jeu qu'un seul connecteur ou un connecteur ainsi que la négation. Finalement, nous montrons que l'optimalité ne dépend pas de l'arité des connecteurs.

Dans ce chapitre, nous indiquons les fonctions par des lettres et des symboles courbes, par exemple f, g, μ, u, etc. et les connecteurs par des lettres et des symboles d'imprimerie, par exemple $\mathtt{f}$, $\mathtt{g}$, $\mathtt{m}$, $\mathtt{u}$.

La plupart des résultats présentés dans ce chapitre ont été publiés dans [CLMP20].

6.1 Définitions et propriétés

Dans cette section, nous adaptons la notion de système de formes normales de [CFL06], que nous avons utilisée au Chapitre 5, et explicitons en particulier la structure des termes que de tels systèmes contiennent.

6.1.1 Définitions

Définition 95. Étant donnée une suite $\mathtt{a}_1, \ldots, \mathtt{a}_n$ de connecteurs distincts, nous disons qu'un terme est *stratifié relativement à* $\mathtt{a}_1, \ldots, \mathtt{a}_n$ si :

1. les symboles d'opération qui apparaissent dans t sont parmi $\mathtt{a}_1, \ldots, \mathtt{a}_n, 0, 1$; et
2. tout sous-terme de t de la forme $\mathtt{a}_j(t_1, \ldots, t_{\mathrm{ar}(\mathtt{a}_j)})$, pour $j \in \{1, \ldots, n\}$ n'a aucun sous-terme de la forme $\mathtt{a}_i(t_1, \ldots, t_{\mathrm{ar}(\mathtt{a}_i)})$, pour $i < j$.

Nous donnons dans la figure 6.1 une représentation schématique d'un terme stratifié relativement à $\mathtt{a}, \ldots, \mathtt{a}_n$, vu comme un arbre. Les ε_i, $i \in [N]$ pour un certain entier naturel N, désignent des variables, des variables niées, ou des constantes.

L'ensemble de tous les termes stratifiés relativement à $\mathtt{a}_1, \ldots, \mathtt{a}_n$ est noté $T(\mathtt{a}_1 \cdots \mathtt{a}_n)$.

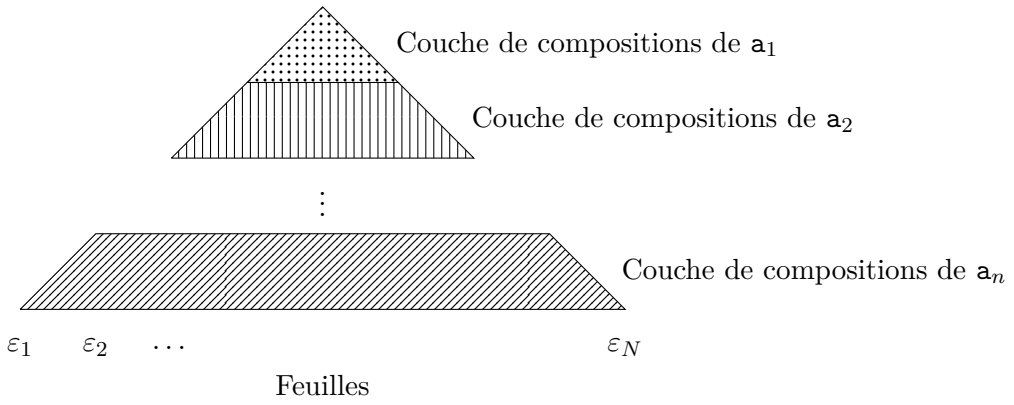


FIGURE 6.1 – Schéma d'un terme stratifié relativement à $\mathtt{a}_1, \ldots, \mathtt{a}_n$.

Exemple 96. Soit $\mathtt{a}$ un connecteur ternaire et $\mathtt{b}$ un connecteur binaire. Le terme

$$\mathtt{b}(\mathtt{a}(x_1, 0, x_3), x_4)$$

est un terme de $T(\mathtt{ba})$, de $T(\mathtt{ba}\neg)$, et de $T(\neg\mathtt{ba})$, mais n'est ni un terme de $T(\mathtt{a})$ ni un terme de $T(\mathtt{ab})$. Le terme

$$\mathtt{b}(\neg(\mathtt{a}(x_1, x_2, x_3)), 1)$$

est un terme de $T(\mathtt{b}\neg\mathtt{a})$ mais n'est pas un terme de $T(\mathtt{ba}\neg)$. ■

Remarque 97. Dans un terme $t \in T(\mathtt{a}_1\mathtt{a}_2\cdots\mathtt{a}_n\neg)$, la négation $\neg$ ne peut être appliquée qu'à des variables ou des constantes, ou des variables ou des constantes sur lesquelles porte déjà une négation. Par exemple, $\neg(\neg(x))$ et $\neg(0)$ peuvent être des sous-termes de t, mais pas le terme $\neg(\mathtt{a}_i(t_1, \ldots, t_{\mathrm{ar}(\mathtt{a}_i)}))$.

Nous rappelons que l'*interprétation* d'un terme ou d'un ensemble de termes T est notée $[T]$; voir le chapitre 3. Ainsi, étant donné un terme t, ici $[t]$ indique une fonction Booléenne ; étant donné un ensemble de termes $T = \{t_1, \ldots, t_n\}$, $[T]$ désigne l'ensemble de fonctions Booléennes

$$[T] = \{[t_1], \ldots, [t_n]\}.$$

Définition 98. Étant donnée une suite $\mathtt{a}_1, \ldots, \mathtt{a}_n$ de connecteurs distincts, l'ensemble de termes stratifiés $T(\mathtt{a}_1\cdots\mathtt{a}_n)$ est *redondant* s'il existe $i \in \{1, \ldots, n\}$ tels que

$$[T(\mathtt{a}_1\cdots\mathtt{a}_{i-1}\mathtt{a}_{i+1}\cdots\mathtt{a}_n)] = [T(\mathtt{a}_1\cdots\mathtt{a}_n)].$$

Sinon, nous disons qu'il est *non-redondant*.

La propriété de redondance exprime le fait que l'un ou plus des connecteurs n'apporte rien à l'ensemble de termes vis-à-vis de sa puissance expressive.

Exemple 99. Par exemple, l'ensemble $T(\uparrow\neg)$ est redondant parce que $\uparrow(x, x) \equiv \neg x$. Ainsi, $T(\uparrow) = \Omega$: la négation $\neg$ n'apporte rien en terme d'expressivité. Par contre, $T(\mathtt{m}\neg)$ n'est pas redondant car $[T(\mathtt{m}\neg)] = \Omega$ mais $[T(\mathtt{m})] = SM \neq \Omega$ et $[T(\neg)] = I^* \neq \Omega$. ■

Dans [CFL06], les auteurs ont observé que certaines factorisations du clone Ω produisent des systèmes de formes normales. Par exemple, la factorisation

$$\Omega = \mathcal{C}(\vee) \circ \mathcal{C}(\wedge) \circ \mathcal{C}(\neg)$$

exprime le fait que toute fonction Booléenne admet une représentation en Forme Normale Disjonctive (DNF, de l'anglais *disjunctive normal form*).

Nous adaptons quelque peu la notion de NFSs afin d'en expliciter les connecteurs.

Définition 100. Un *système de formes normales* (NFS, de l'anglais *normal form system*) est un ensemble non-redondant $T(\alpha_1 \cdots \alpha_n)$ de termes stratifiés tels que

$$[T(\mathtt{a}_1 \cdots \mathtt{a}_n)] = \Omega.$$

Si tous les $[\mathtt{a}_i]$ sont des fonctions pseudo-monotones, alors $T(\mathtt{a}_1 \cdots \mathtt{a}_n)$ est dit *monotone*.

Définition 101. Nous appelons les NFSs définis ci-dessous *NFSs primaux*.

- $\mathbf{M} := T(\mathtt{m}\neg)$;
- $\mathbf{W} := T(\mathtt{w}\neg)$;
- $\mathbf{D} := T(\vee \wedge \neg)$;
- $\mathbf{S} := T(\uparrow)$;
- $\mathbf{P} := T(\oplus\wedge)$;
- $\mathbf{M}_{2n+1} := T(\mathtt{m}_{2n+1}\neg)$;
- $\mathbf{U} := T(\mathtt{u}\neg)$;
- $\mathbf{C} := T(\wedge \vee \neg)$;
- $\mathbf{S}^{\mathrm{d}} := T(\downarrow)$;
- $\mathbf{P}^{\mathrm{d}} := T(\oplus\vee)$.

Le choix de ces fonctions est partiellement lié au fait que ce sont des générateurs de clones intéressants. Rappelons que $[\mathtt{m}]$ est la fonction médiane ternaire, $\mathtt{m}_{2n+1}$ la fonction médiane $2n+1$-aire, $[\mathtt{w}] = (x_1 \wedge x_2) \vee x_3$ est le générateur d'arité minimale du clone $M_{\mathrm{c}}W_\infty$ des fonctions monotones, 1-séparantes, et qui préservent les constantes, $[\mathtt{u}] = (x_1 \vee x_2) \wedge x_3$ est le générateur d'arité minimale du clone $M_{\mathrm{c}}U_\infty$ des fonctions monotones, 0-séparantes, et qui préservent les constantes, $[\uparrow] = \neg(x_1 \wedge x_2)$ est un générateur d'arité minimale du clone Ω de toutes les fonctions Booléennes, et $[\downarrow] = \neg(x_1 \vee x_2)$ est un autre générateur du clone Ω.

Les NFSs $\mathbf{M}$, $\mathbf{C}$, $\mathbf{D}$, $\mathbf{P}$ et $\mathbf{P}^{\mathrm{d}}$ correspondent respectivement aux formes normales médiane, conjonctive, disjonctive, polynomiale et polynomiale duale. Remarquons que tous les NFSs primaux sont monotones, sauf $\mathbf{P}$ et $\mathbf{P}^{\mathrm{d}}$, car $\oplus$ n'est pas pseudo-monotone.

6.1.2 Quelques propriétés des systèmes de formes normales

L'interprétation de n'importe quel terme de $T(\mathtt{a}_1 \cdots \mathtt{a}_n)$ peut être exprimée comme une composition ordonnée de fonctions de $\mathcal{C}([\mathtt{a}_1]), \ldots, \mathcal{C}([\mathtt{a}_n])$ et I, respectivement. On y voit là une correspondance entre d'une part la structure d'algèbre de termes et la composition de connecteurs, et d'autre part la structure de clone et la composition de fonctions.

Fait 102. $[T(\mathtt{a}_1 \cdots \mathtt{a}_n)] = \mathcal{C}([\mathtt{a}_1]) \circ \cdots \circ \mathcal{C}([\mathtt{a}_n]) \circ I$.

Par exemple,

$$\begin{aligned}[\mathbf{M}] &= [T(\mathtt{m}\neg)] \\ &= \mathcal{C}([\mathtt{m}]) \circ \mathcal{C}([\neg]) \circ I \\ &= \mathcal{C}(\mu) \circ \mathcal{C}(\neg) \circ I.\end{aligned}$$

La table de composition de clones de [CFL06] que nous avons reproduite dans le chapitre 3 (table 3.3) révèle le fait suivant.

Fait 103. Pour tout clone $\mathcal{C}$, la composition $\mathcal{C} \circ I$ est aussi un clone : c'est le clone généré par l'ensemble $\mathcal{C} \cup I$. De plus, $\mathcal{C} \circ I = I \circ \mathcal{C} \circ I$. Par conséquent, pour tous clones $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_n$,

$$\mathcal{C}_1 \circ \mathcal{C}_2 \circ \ldots \mathcal{C}_n \circ I = (\mathcal{C}_1 \circ I) \circ (\mathcal{C}_2 \circ I) \circ \cdots \circ (\mathcal{C}_n \circ I).$$

Lemme 104. Soient $\mathtt{a}_1, \ldots, \mathtt{a}_n$ et $\mathtt{b}_1, \ldots, \mathtt{b}_n$ des connecteurs tels que $[T(\mathtt{a}_i)] = [T(\mathtt{b}_i)]$ pour tout i, c'est-à-dire tels que $\mathcal{C}([\mathtt{a}_i]) \circ I = \mathcal{C}([\mathtt{b}_i]) \circ I$, pour tous $i \in \{1, \ldots, n\}$. Alors,

$$[T(\mathtt{a}_1 \cdots \mathtt{a}_n)] = [T(\mathtt{b}_1 \cdots \mathtt{b}_n)].$$

Démonstration. D'après les Faits 102 et 103,

$$\begin{aligned}
[T(\mathtt{a}_1 \cdots \mathtt{a}_n)] &= \mathcal{C}([\mathtt{a}_1]) \circ \cdots \circ \mathcal{C}([\mathtt{a}_n]) \circ I \\
&= (\mathcal{C}([\mathtt{a}_1]) \circ I) \circ \cdots \circ (\mathcal{C}([\mathtt{a}_n]) \circ I) \\
&= (\mathcal{C}([\mathtt{b}_1]) \circ I) \circ \cdots \circ (\mathcal{C}([\mathtt{b}_n]) \circ I) \\
&= \mathcal{C}([\mathtt{b}_1]) \circ \cdots \circ \mathcal{C}([\mathtt{b}_n]) \circ I \\
&= [T(\mathtt{b}_1 \cdots \mathtt{b}_n)].
\end{aligned}$$

□

La table de composition de clones nous permet également d'obtenir une caractérisation des fonctions quasi-Sheffer qui sont, comme définies dans la définition 11, des fonctions f telles que $\mathcal{C}(f) \circ \Omega(1) = \Omega$.

Proposition 105. Une fonction f est quasi-Sheffer si et seulement si $f \notin V \cup L \cup \Lambda$.

Démonstration. Le résultat peut être lu directement dans la table de composition de clones de [CFL06], que nous avons reproduite dans la table 3.3. Étant donné que la composition de classes de fonctions est monotone vis-à-vis de l'inclusion des ensembles, et que les clones SM, $M_c U_\infty$, et $M_c W_\infty$ vérifient

$$SM \circ \Omega(1) = M_c U_\infty \circ \Omega(1) = M_c W_\infty \circ \Omega(1) = \Omega,$$

si une fonction n'est pas quasi-Sheffer, alors $\mathcal{C}(f) \subset V \cup L \cup \Lambda$. Réciproquement, si une fonction $f \in V \cup L \cup \Lambda$, alors $\mathcal{C}(f) \circ \Omega(1) \neq \Omega$. □

Exemple 106. Le clone SM des fonctions auto-duales et monotones est précomplet, car il contient la fonction majorité ternaire μ. En revanche, le clone Λ de toutes les conjonctions n'est pas précomplet. ■

La séparation entre les clones précomplets et les clones qui ne le sont pas est indiquée dans la figure 3.6.

Si au moins deux connecteurs ainsi que la négation sont permis pour construire un NFS, alors, par la propriété de non-redondance, ces connecteurs sont nécessairement dans le clone des conjonctions Λ, des disjonctions V, ou des fonctions linéaires L.

Lemme 107. Si $T(\mathtt{a}_1 \cdots \mathtt{a}_n \neg)$, avec $n > 1$, est un NFS, alors chacun des $[\mathtt{a}_i]$ est dans $V \cup L \cup \Lambda$.

Démonstration. Supposons qu'il existe un i tel que $[\mathtt{a}_i]$ ne soit pas dans $V \cup L \cup \Lambda$. Par la Proposition 105, $[\mathtt{a}_i]$ est quasi-Sheffer, c'est-à-dire que $\mathcal{C}([\mathtt{a}_i]) \circ \Omega(1) = \Omega$. Donc, $[T(\mathtt{a}_i \neg)] = \mathcal{C}([\mathtt{a}_i]) \circ \mathcal{C}([\neg]) \circ I = \mathcal{C}([\mathtt{a}_i]) \circ \Omega(1) = \Omega$, donc $T(\mathtt{a}_1 \cdots \mathtt{a}_n \neg)$ est redondant et n'est donc pas un NFS. □

De plus, par la nécessité de non-redondance encore, il ne peut y avoir plus de 2 tels connecteurs dans un NFS.

Proposition 108. Si $[\mathtt{a}_1], \ldots, [\mathtt{a}_n] \in \Omega \setminus \Omega(1)$ et si $T(\mathtt{a}_1 \cdots \mathtt{a}_n\neg)$ est un NFS, alors $n \leq 2$.

Démonstration. Supposons au contraire que $n \geq 3$. Nous allons parvenir à une contradiction. Par le Lemme 107, $[\mathtt{a}_1], \ldots, [\mathtt{a}_n]$ sont tous dans $V \cup L \cup \Lambda$. Pour $i \in \{1, \ldots, n\}$, soit

$$[\mathtt{b}_i] := \begin{cases} \vee & \text{si } [\mathtt{a}_i] \in V \setminus \Omega(1), \\ \wedge & \text{si } [\mathtt{a}_i] \in \Lambda \setminus \Omega(1), \\ \oplus & \text{si } [\mathtt{a}_i] \in L \setminus \Omega(1). \end{cases}$$

Alors

$$\mathcal{C}([\mathtt{a}_i]) \circ I = \mathcal{C}([\mathtt{b}_i]) \circ I$$

pour tout $i \in \{1, \ldots, n\}$. D'après le Lemme 104,

$$[T(\mathtt{a}_{i_1} \cdots \mathtt{a}_{i_\ell}\neg)] = [T(\mathtt{b}_{i_1} \cdots \mathtt{b}_{i_\ell}\neg)]$$

pour tout $i_1, \ldots, i_\ell \in \{1, \ldots, n\}$.

Si $\mathtt{b}_i = \mathtt{b}_{i+1}$ pour un certain $i \in \{1, \ldots, n-1\}$, alors

$$\begin{aligned} [T(\mathtt{a}_1 \cdots \mathtt{a}_n\neg)] &= [T(\mathtt{b}_1 \cdots \mathtt{b}_n\neg)] \\ &= [T(\mathtt{b}_1 \cdots \mathtt{b}_i\mathtt{b}_{i+2} \cdots \mathtt{b}_n\neg)] \\ &= [T(\mathtt{a}_1 \cdots \mathtt{a}_i\mathtt{a}_{i+2} \cdots \mathtt{a}_n\neg)]; \end{aligned}$$

donc $T(\mathtt{a}_1 \cdots \mathtt{a}_n\neg)$ est redondant, donc ce n'est pas un NFS, ce qui est contradictoire.

Supposons maintenant que $\mathtt{b}_i \neq \mathtt{b}_{i+1}$ pour tout $i \in \{1, \ldots, n-1\}$. Alors, il existe des indices i, j avec $i < j$ tels que

$$(\mathtt{b}_i, \mathtt{b}_j) \in \{(\vee, \wedge), (\wedge, \vee), (\oplus, \wedge), (\oplus, \vee)\}.$$

Puisque $T(\vee \wedge \neg)$, $T(\wedge \vee \neg)$, $T(\oplus\wedge)$ et $T(\oplus\vee)$ sont des NFSs primaux (voir la Définition 101), il s'ensuit que $T(\mathtt{b}_1 \cdots \mathtt{b}_n\neg)$ et, par conséquent, que $T(\mathtt{a}_1 \cdots \mathtt{a}_n\neg)$ sont redondants et ne sont donc pas des NFSs, ce qui est là encore contradictoire. □

Dans [Wer42], Wernick montre qu'il n'y a pas d'ensemble non-redondant complet au sens des fonctions Booléennes qui contient plus de trois connecteurs logiques binaires. La Proposition 108 étend ce résultat à des connecteurs d'arité arbitraire, dans le cas de termes stratifiés dans lesquels les négations n'apparaissent qu'au niveau des feuilles.

Corollaire 109. Si $T(\mathtt{a}_1 \cdots \mathtt{a}_n\neg)$ est un NFS, alors soit $n = 2$ et tous les $[\mathtt{a}_i]$ sont dans $V \cup \Lambda$, soit $n = 1$ et $[\mathtt{a}_1]$ est quasi-Sheffer.

Ce corollaire motive la définition suivante.

Définition 110. Un NFS est *Sheffer* (respectivement *quasi-Sheffer*) s'il est de la forme $T(\mathtt{a})$ (respectivement $T(\mathtt{a}\neg)$).

Pour démontrer aisément qu'un NFS est Sheffer (respectivement quasi-Sheffer), il suffit de vérifier que $\mathtt{a}$ peut exprimer $\wedge, \vee, \neg$ (respectivement $\wedge, \vee$), en utilisant des variables et, potentiellement, des constantes. Dans la table 6.1, nous donnons quelques exemples de telles vérifications.

NFS	Connecteur	Équivalences
$\mathbf{M} = T(\mathtt{m}\neg)$	$\mathtt{m}$	$\mathtt{m}(x,1,y) \equiv x \vee y$
		$\mathtt{m}(x,0,y) \equiv x \wedge y$
$\mathbf{U} = T(\mathtt{u}\neg)$	$\mathtt{u}$	$\mathtt{u}(x,y,1) \equiv (x \vee y) \wedge 1 \equiv x \vee y$
		$\mathtt{u}(x,0,y) \equiv (x \vee 0) \wedge y \equiv x \wedge y$
$\mathbf{S} = T(\uparrow)$	$\uparrow$	$\uparrow(x,1) \equiv \neg(x \wedge 1) \equiv \neg x$
		$\uparrow(\uparrow(x,1),\uparrow(y,1)) \equiv \neg(\neg x \wedge \neg y) \equiv x \vee y$
		$\uparrow(\uparrow(x,y),1) \equiv \neg(\uparrow(x,y)) \equiv x \wedge y$

TABLE 6.1 – Exprimer $\wedge, \vee, \neg$ avec d'autres connecteurs.

6.2 Efficacité des systèmes de formes normales

Un préordre sur les NFSs a été introduit par [CFL06] : deux NFSs **A** et **B** sont en relation selon ce préordre si **A** est polynomialement au moins aussi efficace que **B**. Dans cette section, nous étendons ce préordre pour comparer des ensembles de termes arbitraires, et non nécessairement des NFSs. Nous proposons des réductions pour convertir des termes d'un NFS à un autre, que nous appliquons à la comparaisons de NFSs. Nous étendons en particulier les résultats de [CFL06] à tous les NFSs primaux.

6.2.1 Définitions

Étant donné un terme $t \in T_\tau(X)$ et $b \in \Omega \cup X$, c'est-à-dire un symbole de fonction ou de variable b, nous notons $|t|_b$ le nombre d'occurrences du symbole b dans t. La *taille* d'un terme t, que nous notons $|t|$, est le nombre d'occurrences de tous les connecteurs, différents de 0, 1, et $\neg$, dans t :

$$|t| = \sum_{\alpha \in \Omega \setminus \{0,1,\neg\}} |t|_\alpha.$$

Par exemple, $|x \wedge (\neg y \vee 1)| = |x \wedge (\neg y \vee 1)|_\wedge + |x \wedge (\neg y \vee 1)|_\vee = 1 + 1 = 2$.

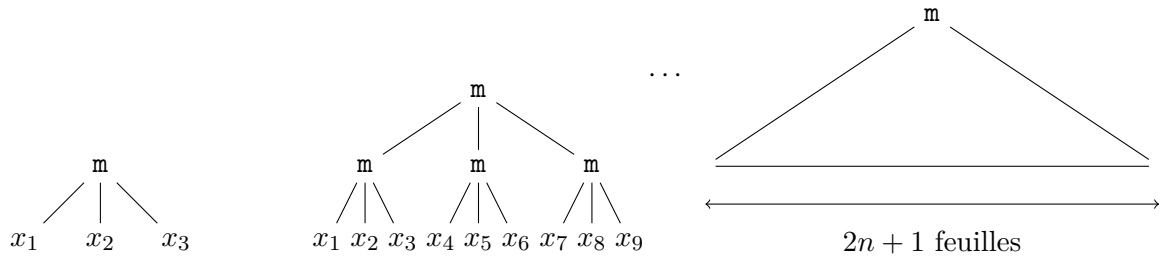
Remarque 111. Notre définition pour la taille d'un terme peut paraître inhabituelle, car nous avons choisi de ne compter ni les constantes, ni les négations, ni les variables. Toutefois, remarquons que le nombre de variables ou de constantes qui apparaissent dans un terme est linéaire en le nombre de connecteurs (d'arité non-nulle). De plus, le nombre de négations dans la plus petite représentation d'une certaine fonction dans un certain NFS est borné par le nombre de variables. Les différences polynomiales en la taille des termes ne sont pas significatives dans l'analyse de l'efficacité des NFSs qui suit. Par conséquent, compter ou pas les constantes, les négations, et les variables n'a pas d'effet sur nos résultats : nous les omettons donc afin de simplifier les calculs. Dans la figure 6.2, nous illustrons la variation (affine) de taille dans un terme médian de taille maximale, selon que l'on compte les feuilles ou non.

Définition 112. Étant donné un ensemble de termes T et une fonction Booléenne $f \in [T]$, la *complexité de f relativement à T*, que nous notons $C_T(f)$, est définie par :

$$C_T(f) = \min\{|t| \colon t \in T,\ [t] = f\}.$$

Exemple 113. Examinons la fonction majorité μ. Le terme

$$\mathtt{m}(x,y,z),$$



Taille : 1	Taille : 4	Taille : n
Taille avec variables : 4	Taille avec variables : 13	Taille avec variables : $3n+1$

FIGURE 6.2 – Variation de taille d'un terme médian.

de taille 1, est le plus petit terme de $T(\mathtt{m}\neg)$ qui représente μ. Par contre, le terme

$$(x \wedge y) \vee (y \wedge z) \vee (z \wedge x)$$

est le plus petit terme de $T(\vee \wedge \neg)$ qui représente μ. Donc $C_{T(\mathtt{m}\neg)}(\mu) = 1$ mais $C_{T(\vee\wedge\neg)}(\mu) = 5$.■

Nous remarquons que pour un ensemble de termes quelconque T, la complexité C_T est une *fonction partielle* sur Ω : toutes les fonctions de Ω n'ont pas forcément de représentation dans T. Par contre, la complexité $C_{\mathbf{A}}$ pour un NFS donné $\mathbf{A}$ est une *fonction totale*, car $[\mathbf{A}] = \Omega$ par définition des NFSs.

Nous généralisons à présent la notion d'efficacité de [CFL06] pour comparer des ensembles de termes qui ne sont pas nécessairement des NFSs.

Définition 114. Soient T, S deux ensembles de termes tels que $[S] \subseteq [T]$.
- T est *polynomialement au moins aussi efficace que* S, noté $T \preceq S$, s'il existe un polynôme $P \in \mathbb{N}[X]$ tel que $\forall f \in [S],\ C_T(f) \leq P(C_S(f))$.
- T and S sont *incomparables*, noté $T \parallel S$, si $T \not\preceq S$ et $S \not\preceq T$.
- T est *polynomialement plus efficace que* S, noté $T \prec S$, si $T \preceq S$ et $S \not\preceq T$.
- T et S sont *équivalents*, noté $T \sim S$, si $T \preceq S$ et $S \preceq T$.

Nous utilisons parfois le symbole $\preceq$ dans l'autre sens : $S \succeq T$ si $T \preceq S$.

Ainsi défini, $\preceq$ est un préordre qui n'est pas total, et $\sim$ est une relation d'équivalence sur l'ensemble des parties de l'ensemble de tous les termes $\mathcal{P}(T)$.

Théorème 115. ([CFL06, Theorème 5])
Pour chaque paire de NFSs $\mathbf{A}, \mathbf{B} \in \{\mathbf{C}, \mathbf{D}, \mathbf{P}, \mathbf{P}^{\mathrm{d}}\}$, si $\mathbf{A} \neq \mathbf{B}$, alors $\mathbf{A} \parallel \mathbf{B}$. De plus,

$$\mathbf{M} \prec \mathbf{C}, \mathbf{D}, \mathbf{P}, \mathbf{P}^{\mathrm{d}}.$$

Nous illustrons le théorème 115 par la figure 6.3.

Après ces préliminaires et définitions, nous pouvons reformuler notre étude : nous sommes intéressés par les *NFSs monotones minimaux*, c'est-à-dire, les NFSs monotones qui sont des minimums pour le préordre $\preceq$. Comme nous allons le voir, ces NFSs minimaux existent, et sont, de plus, tous équivalents, ce qui motive la définition d'optimalité suivante.

Définition 116. Un NFS monotone $\mathbf{A}$ est *optimal* si $\mathbf{A}$ est minimal et s'il n'existe pas de NFS minimal $\mathbf{B}$ incomparable à $\mathbf{A}$, ou, de manière équivalente, si pour tout NFS monotone $\mathbf{B}$ nous avons $\mathbf{A} \preceq \mathbf{B}$.

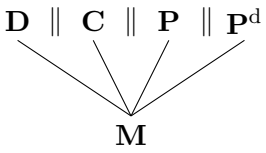


FIGURE 6.3 – Illustration du théorème 115 : semi-treillis de NFSs ordonnés par $\preceq$.

6.2.2 Réductions linéaires et quasi-linéaires

Comme nous allons le voir dans les Théorèmes 135 et 137, les NFSs monotones optimaux sont de la forme $T(\mathtt{a})$ ou $T(\mathtt{a}\neg)$. Ainsi, dans le reste de cette section ainsi que dans la Section 6.3, nous nous concentrons sur les NFSs de ces formes.

Dans cette section, nous définissons des relations entre des ensembles de termes. Ces relations sont de différentes natures, que nous décrivons, en fonction de la manière selon laquelle il est possible de convertir un terme d'un ensemble à un autre.

Exemple 117. En guise d'illustration, considérons l'équivalence

$$\mathtt{u}(x, y, z) \equiv \mathtt{m}(\mathtt{m}(x, 1, y), 0, z)$$

qui nous permet de convertir des termes de $\mathbf{U}$ en des termes de $\mathbf{M}$. Grâce à cette équivalence, les termes peuvent être convertis avec une augmentation de taille qui est, au plus, affine : chaque connecteur $\mathtt{u}$ est remplacé par exactement deux connecteurs $\mathtt{m}$, et, de plus, aucune variable n'est repétée.

Par exemple, convertissons le terme de $\mathbf{U}$

$$\mathtt{u}(\mathtt{u}(x, 0, y), 0, z)$$

qui représente la conjonction ternaire $x \wedge y \wedge z$.

$$\begin{aligned}\mathtt{u}(\mathtt{u}(x, 0, y), 0, z) &\equiv \mathtt{u}(\mathtt{m}(\mathtt{m}(x, 1, 0), 0, y), 0, z)\\ &\equiv \mathtt{m}(\mathtt{m}(\mathtt{m}(\mathtt{m}(x, 1, 0), 0, y), 1, 0), 0, z).\end{aligned}$$

Le terme de $\mathbf{M}$ obtenu est de taille 4, alors que le terme de $\mathbf{U}$ de départ est de taille 2.

De manière générale, si $t^{\min}_{\mathbf{U}}$ est une représentation minimale dans $\mathbf{U}$ d'une fonction Booléenne $f \in \Omega$, alors $C_{\mathbf{U}}(f) = |t^{\min}_{\mathbf{U}}|$. Si $t_{\mathbf{M}}$ est le résultat de la conversion directe du terme $t^{\min}_{\mathbf{U}}$ en utilisant l'équivalence ci-dessus, alors $C_{\mathbf{M}}(f) \leq |t_{\mathbf{M}}|$, et puisque $|t_{\mathbf{M}}| = 2|t^{\min}_{\mathbf{U}}|$, nous obtenons $C_{\mathbf{M}}(f) \leq 2C_{\mathbf{U}}(f)$; et ainsi, par définition, $\mathbf{M}$ est polynomialement au moins aussi efficace que $\mathbf{U}$ ou, en utilisant la notation du préordre, $\mathbf{M} \preceq \mathbf{U}$. ∎

Remarque 118. Cette conversion directe ne garantit pas l'obtention d'une formule minimale. Dans l'exemple 117 précédent, la formule médiane obtenue peut être simplifiée encore plus :

$$\begin{aligned}\mathtt{m}(\mathtt{m}(\mathtt{m}(\mathtt{m}(x, 1, 0), 0, y), 1, 0), 0, z) &\equiv \mathtt{m}(\mathtt{m}(\mathtt{m}(x, 0, y), 1, 0), 0, z)\\ &\equiv \mathtt{m}(\mathtt{m}(x, 0, y), 0, z)\end{aligned}$$

qui est une MNF pour la conjonction ternaire.

Type de réduction	Symbole	Exemple
Linéaire	$A \sqsupseteq B$	$\mathtt{u}(x,y,z) \equiv \mathtt{m}(\mathtt{m}(x,1,y),0,z)$
Universelle quasi-linéaire	$A \sqsupseteq_\forall B$	$\mathtt{m}(x,y,z) \equiv \mathtt{u}(\mathtt{u}(x,0,y),\mathtt{u}(x,y,z),1)$
		$\mathtt{m}(x,y,z) \equiv \mathtt{u}(\mathtt{u}(y,0,z),\mathtt{u}(y,z,x),1)$
		$\mathtt{m}(x,y,z) \equiv \mathtt{u}(\mathtt{u}(z,0,x),\mathtt{u}(z,x,y),1)$
Existentielle quasi-linéaire	$A \sqsupseteq_\exists B$	$\mathtt{s}(x,y,z) \equiv \mathtt{m}(\mathtt{m}(x,0,y),1,\mathtt{m}(\overline{x},0,z))$

TABLE 6.2 – Illustrations de réductions.

Nous mettrons à profit ce type de relations pour établir l'équivalence entre les NFSs monotones optimaux. Dans le meilleur des cas, les connecteurs d'un NFS peuvent être remplacés par un terme linéaire de l'autre NFS. Alors, une réécriture directe de ces termes par un connecteur, c'est-à-dire en remplaçant directement chaque connecteur dans la formule de départ par le terme linéaire mentionné ci-dessus, fournit une conversion efficace. Comme nous allons le voir, il est possible de procéder à des conversions efficaces sous des conditions moins contraignantes.

Définition 119 (Réductions)**.** Considérons deux ensembles de termes $A = T(\mathtt{a}\neg)$ (ou $T(\mathtt{a})$) et $B = T(\mathtt{b}\neg)$ (ou $T(\mathtt{b})$) tels que $[A] \subseteq [B]$ ou, autrement dit, tels que toute fonction qui a une représentation dans A possède aussi une représentation dans B. Nous disons que :

— il existe une *réduction linéaire* de A vers B, ou que A est *linéairement réductible à* B, noté $A \sqsupseteq B$, s'il existe un terme linéaire $t \in T(\mathtt{b})$ tel que

$$\mathtt{a}(x_1,\ldots,x_{\mathrm{ar}(\mathtt{a})}) \equiv t;$$

— il existe une *réduction universelle quasi-linéaire* de A vers B, ou que A est *universellement réductible à* B, noté $A \sqsupseteq_\forall B$, si pour tout $j \in \{1,\ldots,\mathrm{ar}(\mathtt{a})\}$, il existe $t_j \in T(\mathtt{b})$ tel que

$$\mathtt{a}(x_1,\ldots,x_{\mathrm{ar}(\mathtt{a})}) \equiv t_j$$

et que $|t_j|_{x_j} = 1$;

— il existe une *réduction existentielle quasi-linéaire* de A vers B, ou que A est *existentiellement réductible à* B, noté $A \sqsupseteq_\exists B$, s'il existe $t \in T(\mathtt{b})$ tel que

$$\mathtt{a}(x_1,\ldots,x_{\mathrm{ar}(\mathtt{a})}) \equiv t$$

et que $|t|_{x_j} = 1$ pour un certain $j \in \{1,\ldots,\mathrm{ar}(\mathtt{a})\}$.

Nous illustrons ces réductions dans la table 6.2.

Remarque 120. La notion de réductibilité linéaire peut être rattachée à la notion de « fonctions Booléennes à lecture unique » (*read-once Boolean functions* en anglais), qui sont des fonctions qui peuvent être représentées par un terme linéaire [CH11, CL12].

Proposition 121. Pour tous ensembles de termes A et B, $A \sqsupseteq B$ implique $A \sqsupseteq_\forall B$, et $A \sqsupseteq_\forall B$ implique $A \sqsupseteq_\exists B$; autrement dit,

$$\sqsupseteq \subset \sqsupseteq_\forall \subset \sqsupseteq_\exists.$$

Démonstration. Le fait que les inclusions

$$\sqsupseteq \subseteq \sqsupseteq_\forall \subseteq \sqsupseteq_\exists$$

soient vraies est une conséquence directe de leur définition. Nous montrons ici que ces inclusions sont strictes.

Afin de démontrer la stricte inclusion de $\sqsupseteq$ dans $\sqsupseteq_\forall$, nous considérons les NFSs $\mathbf{M}$ et $\mathbf{U}$. Nous déduisons $\mathbf{M} \sqsupseteq_\exists \mathbf{U}$ de l'équivalence

$$\mathtt{m}(x, y, z) \equiv \mathtt{u}(\mathtt{u}(x, 0, y), \mathtt{u}(x, y, z), 1),$$

et ainsi $\mathbf{M} \sqsupseteq_\forall \mathbf{U}$ car μ est symétrique. Par contre, nous n'avons pas que $\mathbf{M} \sqsupseteq \mathbf{U}$, comme l'argument suivant le montre.

Supposons au contraire qu'il existe un terme ternaire linéaire $t \in \mathbf{U}$ équivalent à $\mathtt{m}(x_1, x_2, x_3)$, et supposons que t a la plus petite taille possible parmi les termes qui vérifient ces conditions. Par minimalité, aucun sous-terme de t de la forme $\mathtt{u}(t_1, t_2, t_3)$ ne satisfait $t_1 = 1$ ou $t_2 = 1$ ou $t_3 = 0$, Car pour tous termes s, s', nous avons l'équivalence $\mathtt{u}(1, s, s') \equiv \mathtt{u}(s, 1, s') \equiv s'$ ainsi que $\mathtt{u}(s, s', 0) \equiv 0$; ainsi, nous pourrions obtenir un plus petit terme équivalent à t en remplaçant le sous-terme $\mathtt{u}(t_1, t_2, t_3)$ par t_3 ou 0 en conséquence. Un argument similaire montre qu'aucun sous-terme t de la forme $\mathtt{u}(t_1, t_2, t_3)$ ne satisfait $t_1 = t_2 = 0$, ou $t_1 = 0$ et $t_3 = 1$, ou $t_2 = 0$ et $t_3 = 1$, car pour tout terme s, nous avons les équivalences $\mathtt{u}(0, 0, s) \equiv 0$, $\mathtt{u}(0, s, 1) \equiv \mathtt{u}(s, 0, 1) \equiv s$. Par conséquent, dans chaque sous-terme de la forme $\mathtt{u}(t_1, t_2, t_3)$ au plus un terme parmi t_1, t_2, t_3 est une constante.

Il n'existe pas deux sous-termes $\mathtt{u}(t_1, t_2, t_3)$ et $\mathtt{u}(s_1, s_2, s_3)$ de t tels que deux des termes t_1, t_2, t_3 sont des variables et que deux des termes s_1, s_2, s_3 sont des variables, car alors une variable apparaîtrait deux fois dans t, qui, nous le rappellons, est ternaire ; cela contradirait la linéarité de t.

Par conséquent, il n'existe pas de sous-terme de la forme $\mathtt{u}(t_1, t_2, t_3)$ avec

$$t_{i_1} = \mathtt{u}(t_{i_1,1}, t_{i_1,2}, t_{i_1,3}),$$

et

$$t_{i_2} = \mathtt{u}(t_{i_2,1}, t_{i_2,2}, t_{i_2,3}),$$

pour $i_1 \neq i_2$. Sinon, t_{i_1} et t_{i_2} contiendraient des sous-termes de la forme $\mathtt{u}(s_1, s_2, s_3)$ dans lequels les s_i sont des constantes ou des variables ; comme nous l'avons vu ci-dessus, ces sous-termes ont nécessairement deux variables chacun, ce qui contredit là encore la linéarité de t.

Aucun terme de $\mathbf{U}$ qui contient au plus une occurrence de $\mathtt{u}$ ne peut être équivalent à $\mathtt{m}(x_1, x_2, x_3)$. La seule possibilité restante est que $t = \mathtt{u}(t_1, t_2, t_3)$ dans lequel un des sous-termes t_1, t_2, t_3 est une constante, un autre une variable, et le dernier de la forme $\mathtt{u}(s_1, s_2, s_3)$ dans lequel un des termes s_1, s_2, s_3 est une constante et les deux autres des variables, afin que les trois variables apparaissent et que les conditions sur les constantes que nous avons établies plus haut soit satisfaites.

Pour ces termes, nous avons les équivalences suivantes, avec $\{i, j, k\} = \{1, 2, 3\}$:

$$\begin{aligned}
&\mathtt{u}(\mathtt{u}(x_i, x_j, 1), x_k, 1) \equiv \mathtt{u}(x_i, \mathtt{u}(x_j, x_k, 1), 1) \equiv x_i \vee x_j \vee x_k,\\
&\mathtt{u}(\mathtt{u}(x_i, x_j, 1), 0, x_k) \equiv \mathtt{u}(0, \mathtt{u}(x_i, x_j, 1), x_k) \equiv (x_i \vee x_j) \wedge x_k \equiv \mathtt{u}(x_i, x_j, x_k),\\
&\mathtt{u}(\mathtt{u}(x_i, 0, x_j), x_k, 1) \equiv \mathtt{u}(\mathtt{u}(0, x_i, x_j), x_k, 1) \equiv (x_i \wedge x_j) \vee x_k \equiv \mathtt{w}(x_i, x_j, x_k),\\
&\mathtt{u}(\mathtt{u}(x_i, 0, x_j), 0, x_k) \equiv \mathtt{u}(\mathtt{u}(0, x_i, x_j), 0, x_k) \equiv \mathtt{u}(0, \mathtt{u}(x_i, 0, x_j), x_k)\\
&\qquad\qquad \equiv \mathtt{u}(0, \mathtt{u}(0, x_i, x_j), x_k) \equiv x_i \wedge x_j \wedge x_k.
\end{aligned}$$

Aucun de ces termes n'est équivalent à $\mathtt{m}(x_1, x_2, x_3)$. Nous avons atteint une contradiction, et nous concluons donc que $\mathbf{M} \not\sqsupseteq \mathbf{U}$.

À présent, montrons que $\sqsupseteq_\forall$ est strictement inclus dans $\sqsupseteq_\exists$. Considérons le NFS $T(\mathtt{s})$ dans lequel

$$\mathtt{s} := (x_1 \wedge x_2) \ \vee \ (\neg x_1 \wedge x_3).$$

Fonction	Terme équivalent à $\neg x_1$
$x_1 \oplus x_2$	$x_1 \oplus 1$
$\uparrow(x_1, x_2)$	$\uparrow(x_1, 1)$
$\downarrow(x_1, x_2)$	$\uparrow(x_1, 0)$
$\mathbf{s}(x_1, x_2, x_3)$	$\mathbf{s}(x_1, 0, 1)$

TABLE 6.3 – Simulation de la négation de manière linéaire.

D'après l'équivalence

$$\mathtt{s}(x_1, x_2, x_3) \equiv \mathtt{m}(\mathtt{m}(x_1, x_2, 0), \mathtt{m}(\neg x_1, x_3, 0), 1),$$

nous avons $T(\mathtt{s}) \sqsupseteq_\exists \mathbf{M}$. En revanche, d'après le Corollaire 152, $T(\mathtt{s}) \not\sqsupseteq_\forall \mathbf{M}$. □

Nous montrons à présent qu'il est possible de « simuler » la négation de manière efficace à l'aide d'une fonction non monotone en choisissant des constantes adéquates.

Lemme 122. Si $[\mathtt{a}] \notin M$, alors il existe un terme unaire linéaire $t \in T(\mathtt{a})$ tel que $|t|_\mathtt{a} = 1$ et $t \equiv \neg x_1$.

Démonstration. Supposons que $\mathtt{a}$ soit n-aire. Puisque $\mathtt{a}$ n'est pas monotone, il existe deux tuples $(a_1, \ldots, a_n), (b_1, \ldots, b_n) \in \mathbb{B}^n$ tels que

$$(a_1, \ldots, a_n) \leq (b_1, \ldots, b_n) \quad \text{et} \quad [\mathtt{a}](a_1, \ldots, a_n) > [\mathtt{a}](b_1, \ldots, b_n).$$

Considérons la séquence de tuples $\mathbf{d}_0, \mathbf{d}_1, \ldots, \mathbf{d}_n$, dans laquelle

$$\begin{aligned}
\mathbf{d}_0 &:= (a_1, \ldots, a_n), \\
\mathbf{d}_1 &:= (b_1, a_2, \ldots, a_n), \\
&\vdots \\
\mathbf{d}_i &:= (b_1, \ldots, b_i, a_{i+1}, \ldots, a_n), \\
&\vdots \\
\mathbf{d}_n &:= (b_1, \ldots, b_n).
\end{aligned}$$

Pour tout $j \in \{1, \ldots, n\}$, nous avons $\mathbf{d}_{j-1} \leq \mathbf{d}_j$, et $\mathbf{d}_{j-1}$ et $\mathbf{d}_j$ diffèrent potentiellement seulement sur la composante d'indice j.

Puisque $[\mathtt{a}](\mathbf{d}_0) > [\mathtt{a}](\mathbf{d}_n)$, alors il existe un indice $i \in \{1, \ldots, n\}$ tel que $[\mathtt{a}](\mathbf{d}_{i-1}) > [\mathtt{a}](\mathbf{d}_i)$. Donc $\mathtt{a}(b_1, \ldots, b_{i-1}, x_1, a_{i+1}, \ldots, a_n) \equiv \neg x_1$. □

Nous donnons, dans la table 6.3, quelques exemples de simulations efficaces de la négation $\neg x_1$ par des fonctions Booléennes non monotones, en ce sens que les termes obtenus sont linéaires en x_1.

6.2.3 Propriétés des réductions (quasi-)linéaires

Dans cette section, nous montrons que ces réductions impliquent le préordre $\succeq$, ce qui nous permettra par la suite d'ordonner des ensembles de termes et des NFSs suivant $\succeq$.

Proposition 123. Pour tous ensembles de termes A et B, $A \sqsupseteq B$ implique $A \succeq B$; autrement dit, $\sqsupseteq \subseteq \succeq$.

Démonstration. Étant donnés des ensembles de termes $A = T(\mathtt{a}\neg)$ (ou $T(\mathtt{a})$) et $B = T(\mathtt{b}\neg)$) (ou $(T(\mathtt{b}))$ tels que $A \sqsupseteq B$, il existe un terme linéaire $t \in T(\mathtt{b})$ tel que $\mathtt{a}(x_1, \ldots, x_{\mathrm{ar}\,(\mathtt{a})}) \equiv t$. Puis, nous pouvons convertir tout terme s de A en un terme équivalent de B en remplaçant chaque occurrence de $\mathtt{a}$ par t ; plus précisément, en remplaçant chaque sous-terme de la forme

$$\mathtt{a}(s_1, \ldots, s_{\mathrm{ar}\,(\mathtt{a})})$$

par

$$t\{s_1/x_1, \ldots, s_{\mathrm{ar}\,(\mathtt{a})}/x_{\mathrm{ar}\,(\mathtt{a})}\}.$$

Puisque t est linéaire, la taille du terme de B obtenu est $|t| \cdot |s|$, c'est-à-dire un multiple constant de $|s|$. □

Exemple 124. Nous avons la propriété $\mathbf{M} \sqsupseteq \mathbf{M}_5$ car $\mathtt{m}(x, y, z) \equiv \mathtt{m}_5(0, 1, x, y, z)$. Par exemple, grâce à cette équivalence, nous pouvons convertir le terme

$$t_1 := \mathtt{m}(\mathtt{m}(x, y, z), u, v)$$

en

$$t_2 := \mathtt{m}_5(0, 1, \mathtt{m}_5(0, 1, x, y, z), u, v).$$

De plus, $|t_1| = |t_2|$. D'après la Proposition 123, il en résulte que $\mathbf{M} \succeq \mathbf{M}_5$. ■

Proposition 125. Si $A = T(\mathtt{a}\neg)$ (ou $T(\mathtt{a})$) et $B = T(\mathtt{b}\neg)$ (ou $T(\mathtt{b})$) sont deux ensembles de termes tels que $A \sqsupseteq_\forall B$ et $t_1, \ldots, t_{\mathrm{ar}(\mathtt{a})}$ sont des termes qui satisfont les conditions d'une réduction universelle quasi-linéaire de A vers B, alors pour tout $f \in [A]$, nous avons

$$C_B(f) \leq nk(C_A(f))^q + 1$$

où $n := \mathrm{ar}\,(\mathtt{b})$, $k := \max_i\{|t_i|_\mathtt{b}\}$, et $q := \max_{i,j}\{|t_i|_{x_j}\}$. Par conséquent,

$$\sqsupseteq_\forall \subseteq \succeq.$$

Démonstration. Soient A et B deux ensembles de termes qui sont tels que $A \sqsupseteq_\forall B$. Autrement dit, par définition,

$$\forall i, \exists t_i \in T(\mathtt{b}),\ \alpha(x_1, \ldots, x_{\mathrm{ar}\,(\mathtt{a})}) \equiv t_i \quad \text{et} \quad |t_i|_{x_i} = 1.$$

Afin de démontrer que $A \succeq B$, nous donnons un procédé récursif efficace pour convertir un terme de A en un terme de B qui lui est équivalent. Puis, nous démontrons que la taille du terme une fois converti est polynomiale en la taille du terme original de A.

Il nous faut distinguer les cas où $B = T(\mathtt{b}\neg)$ et où $B = T(\mathtt{b})$. Tout d'abord, nous considérons le cas où $B = T(\mathtt{b}\neg)$. Soit s un terme de A. Nous rappelons qu'étant donnée une suite de n entiers $(r_i)_{i=1}^n$, $\mathrm{argmax}_i(r_i)$ est le plus petit entier j tel que pour tout $i \in \{1, \ldots, n\}$, $r_j \geq r_i$. Nous désignons par

$$\textsc{conv}_{A\to B}(s)$$

le terme de B, équivalent à s, défini de manière inductive comme suit.

- Si s est une variable ou une constante, alors $\textsc{conv}_{A\to B}(s) := s$;
- si $s = \neg t$, alors $\textsc{conv}_{A\to B}(s) := s$;
- si $s = \mathtt{a}(s_1, \ldots, s_{\mathrm{ar}\,(\mathtt{a})})$, alors

$$\textsc{conv}_{A\to B}(s) := t_\ell\{\textsc{conv}_{A\to B}(s_1)/x_1, \ldots, \textsc{conv}_{A\to B}(s_{\mathrm{ar}\,(\mathtt{a})})/x_{\mathrm{ar}\,(\mathtt{a})}\},$$

avec $\ell := \mathrm{argmax}_i(|\textsc{conv}_{A\to B}(s_i)|)$.

L'idée sous-jacente à ce procédé de conversion récursive est de repérer le sous-terme de taille maximale parmi ceux qui ont déjà été convertis afin de ne pas le répéter. Comme nous allons le voir, cette précaution est suffisante pour assurer une conversion efficace, c'est-à-dire sans explosion exponentielle de la taille du terme converti. Le fait que $\textsc{conv}_{A\to B}(s) \equiv s$ est assuré par la stabilité des interprétations par substitution.

Soit $k := \max_i\{|t_i|_{\mathtt{b}}\}$ et $q := \max_{i,j}\{|t_i|_{x_j}\}$. Soit un terme s de A qui représente une fonction Booléenne f. Nous allons montrer, par induction sur la structure des termes de A, que $|\textsc{conv}_{A\to B}(s)| \leq k|s|^q$.

— Supposons que s soit un littéral ou une constante. Alors, $|\textsc{conv}_{A\to B}(s)| = 0 = |s| = k|s|^q$.

— Supposons que

$$s = \alpha(s_1, s_2, \ldots, s_{\mathrm{ar}\,(\mathtt{a})})$$

tel que $s_i \in T(\mathtt{a})$ pour tout i. Nous rappelons que ℓ est défini par

$$\ell = \mathrm{argmax}_i(|\textsc{conv}_{A\to B}(s_i)|).$$

Par conséquent,

$$\begin{aligned}
|\textsc{conv}_{A\to B}(s)| = |t_\ell\{\textsc{conv}_{A\to B}(s_i)/x_i\}| &= |t_\ell|_{\mathtt{b}} + \sum_{j=1}^{\mathrm{ar}\,(\mathtt{a})} |t_\ell|_{x_j}|\textsc{conv}_{A\to B}(s_j)| \\
&\leq k + q \sum_{j=1,\ j\neq\ell}^{\mathrm{ar}\,(\mathtt{a})} |\textsc{conv}_{A\to B}(s_j)| + |\textsc{conv}_{A\to B}(s_\ell)| \\
&\leq k + |\textsc{conv}_{A\to B}(s_1)| + q \sum_{j=2}^{\mathrm{ar}\,(\mathtt{a})} |\textsc{conv}_{A\to B}(s_j)|.
\end{aligned}$$

L'avant-dernière inégalité provient du fait que $|t_\ell|_{x_\ell} = 1$, $|t_i|_{x_j} \leq q$, et $|t_\ell|_{\mathtt{b}} \leq k$. La dernière inégalité provient du fait que

$$|\textsc{conv}_{A\to B}(s_1)| \leq |\textsc{conv}_{A\to B}(s_\ell)|$$

par définition de ℓ, et donc que

$$q|\textsc{conv}_{A\to B}(s_1)| + |\textsc{conv}_{A\to B}(s_\ell)| \leq |\textsc{conv}_{A\to B}(s_1)| + q|\textsc{conv}_{A\to B}(s_\ell)|.$$

Supposons à présent sans perte de généralité que

$$|s_1| \geq |s_2| \geq \cdots \geq |s_{n-1}| \geq |s_{\mathrm{ar}\,(\mathtt{a})}|.$$

Alors,

$$\begin{aligned}
|\textsc{conv}_{A\to B}(s)| &\leq k(1 + |s_1|^q + q \sum_{i=2}^{\mathrm{ar}\,(\mathtt{a})} |s_i|^q) \quad \text{par hypothèse d'induction} \\
&\leq k(1 + |s_1| + |s_2| + \cdots + |s_{\mathrm{ar}\,(\mathtt{a})}|)^q = k|s|^q.
\end{aligned}$$

La dernière inégalité provient du fait que $|s_{i+1}|^q \leq |s_{i+1}|^{q-1}|s_i|$.

Soit $f \in [A] \subseteq [B]$ et soit s un terme de taille minimale dans A qui représente f. Alors,

$$C_A(f) = |s|.$$

De plus,

$$C_B(f) \leq |\textsc{conv}_{A\to B}(s)|.$$

Puisque

$$|\textsc{conv}_{A\to B}(s)| \leq k|s|^q,$$

nous avons donc :

$$C_B(f) \leq |\textsc{conv}_{A\to B}(s)| \leq k|s|^q = k(C_A(f))^q.$$

Par conséquent, $A \succeq B$.

Nous considérons à présent le cas où $B = T(\mathtt{b})$. Si $A = T(\mathtt{a})$, alors la conversion de A vers B décrite ci-haut fonctionne sans plus de modification, et nous avons la même borne supérieure polynomiale pour la taille du terme converti. Si $A = T(\mathtt{a}\neg)$, alors nous avons besoin de gérer les négations qui peuvent apparaître dans un terme $s \in T(\mathtt{a}\neg)$. Dans le cas où $B = T(\mathtt{b})$, $[\mathtt{b}]$ n'est nécessairement pas monotone. Donc, d'après le Lemme 122, il existe un terme unaire linéaire $t \in T(\mathtt{b})$ tel que $|t|_{\mathtt{b}} = 1$ et $t \equiv \neg x_1$. Soit

$$s' := \textsc{conv}_{A\to T(\mathtt{b}\neg)}(s)$$

la conversion de s en un terme équivalent de $T(\mathtt{b}\neg)$ en suivant le procédé décrit ci-haut, et soit s'' le terme obtenu à partir de s' en y remplaçant tous les sous-termes de la forme $\neg u$ par $t\{u/x_1\}$. Alors, $s'' \in T(\mathtt{b})$ et $s'' \equiv s$. Si s est un terme de taille minimale dans A qui représente une fonction f, alors il n'y a aucune négation itérée dans s, et le nombre de négations dans s' est au plus de $(n-1)|s'| + 1$, où $n := \operatorname{ar}(\mathtt{b})$. Puisque chaque négation de s' est remplacée par un terme (t) qui ne contient qu'une seule occurence de $\mathtt{b}$, alors

$$C_B(f) \leq |s''| \leq |s'| + (n-1)|s'| + 1 = n|s'| + 1 \leq nk|s|^q + 1 = nk(C_A(f))^q + 1.$$

Ainsi, dans ce cas aussi, $A \succeq B$. □

Exemple 126. Examinons le connecteur $\mathtt{m}$. Nous avons les équivalences suivantes :

$$\begin{aligned}
\mathtt{m}(x,y,z) &\equiv (y \uparrow z) \uparrow (x \uparrow ((y \uparrow 1) \uparrow (z \uparrow 1))),\\
\mathtt{m}(x,y,z) &\equiv (x \uparrow z) \uparrow (y \uparrow ((x \uparrow 1) \uparrow (z \uparrow 1))),\\
\mathtt{m}(x,y,z) &\equiv (y \uparrow x) \uparrow (z \uparrow ((y \uparrow 1) \uparrow (x \uparrow 1))).
\end{aligned}$$

Puisque chaque équivalence est linéaire en une variable (respectivement x, y, et z), alors $\mathbf{M} \sqsupseteq_\forall \mathbf{S}$. D'après la proposition 125, nous déduisons que $\mathbf{M} \succeq \mathbf{S}$. ■

Si un des connecteurs est une fonction symétrique, nous pouvons également traiter le cas des réductions quasi-linéaires existentielles.

Proposition 127. Soient deux ensembles de termes stratifiés $A = T(\mathtt{a}\neg)$ et $B = T(\mathtt{b}\neg)$. Si $A \sqsupseteq_\exists B$ et si $[\mathtt{a}]$ est une fonction symétrique, alors $A \sqsupseteq_\forall B$; par conséquent, $A \succeq B$.

Démonstration. La symétrie de $[\mathtt{a}]$ nous permet de produire une réduction quasi-linéaire universelle à partir de la réduction $A \sqsupseteq_\exists B$. Il suffit ensuite d'appliquer la proposition 125. □

Exemple 128. L'algorithme de conversion utilisé dans la preuve de la proposition 123 peut être implémenté et effectivement utilisé pour convertir des formules. Nous illustrons ici la propriété $\mathbf{M}_5 \sqsupseteq \mathbf{M}$, que nous démontrons formellement ci-après dans la proposition 133.

Toutes les formules médianes de $\mathbf{M}_5$ de taille inférieure à 13 sont générées, puis converties suivant l'algorithme décrit précédemment en une formule de $\mathbf{M}$, grâce à l'équivalence

$$\mathtt{m}_5(x_1, x_2, x_3, x_4, x_5) \equiv \mathtt{m}(\mathtt{m}(\mathtt{m}(x_2, x_3, x_4), x_4, x_5), \mathtt{m}(x_2, x_3, x_5), x_1)$$

dont nous donnons une démonstration dans l'exemple 61.

Dans la figure 6.4, nous avons représenté par des carrés rouges, pour chaque taille de formule de $\mathbf{M}_5$, la taille des formules de $\mathbf{M}$ après conversion. En reprenant les notations de la proposition 123,

$$\begin{aligned}
\mathtt{a} &= \mathtt{m}_5, \\
\mathtt{b} &= \mathtt{m}, \\
n &= \mathrm{ar}\,(\mathtt{b}) = 3, \\
k &= \max_i\{|t_i|_{\mathtt{b}}\} = 4, \\
q &= \max_{i,j}\{|t_i|_{x_j}\} = 2,
\end{aligned}$$

où les t_i sont les termes de $\mathbf{M}$ obtenus en permutant les variables de la partie droite de l'équivalence ci-dessus. Nous avons donc également représenté par des ronds bleus le polynôme

$$X \mapsto nkX^q + 1 = 12X^2 + 1$$

qui illustre la majoration polynomiale de la taille des formules converties.

Notons que ces formules converties ne sont pas nécessairement minimales, et que la majoration obtenue n'est pas optimale. ∎

6.2.4 Équivalences entre les NFSs primaux Sheffer et quasi-Sheffer

Pour illustrer et appliquer la réductibilité linéaire, nous montrons que tous les NFSs primaux, qu'ils soient Sheffer ou quasi-Sheffer, sont tous équivalents à $\mathbf{M}$, et que, par conséquent, ils sont strictement plus efficaces que les autres NFSs primaux qui ne sont ni Sheffer ni quasi-Sheffer. Pour cela, nous adaptons le système de décomposition médiane de [Mar09b], que nous avions reproduit déjà dans l'équation 5.1, aux termes (et non seulement aux fonctions).

La plupart des preuves reposent sur la production de termes qui permettent des conversions efficaces.

Proposition 129. Les NFSs primaux $\mathbf{U}$, $\mathbf{W}$, and $\mathbf{M}$ sont équivalents deux-à-deux, c'est-à-dire que $\mathbf{U} \sim \mathbf{W} \sim \mathbf{M}$.

Démonstration. Considérons les équivalences

$$\mathtt{u}(x, y, z) \equiv \mathtt{m}(\mathtt{m}(x, 1, y), 0, z)$$

et

$$\mathtt{m}(x, y, z) \equiv \mathtt{u}(\mathtt{u}(x, 0, y), \mathtt{u}(x, y, z), 1).$$

Nous avons que

$$|\mathtt{m}(\mathtt{m}(x, 1, y), 0, z)|_w = 1$$

pour tout $w \in \{x, y, z\}$ et que

$$|\mathtt{u}(\mathtt{u}(x, 0, y), \mathtt{u}(x, y, z), 1)|_z = 1.$$

Par conséquent, $\mathbf{U} \sqsupseteq \mathbf{M}$ and $\mathbf{M} \sqsupseteq_\exists \mathbf{U}$. Les Propositions 123 et 125 ainsi que la symétrie de $[\mathtt{m}]$ impliquent que $\mathbf{U} \sim \mathbf{M}$. Un raisonnement dual permet de prouver que $\mathbf{W} \sim \mathbf{M}$. □

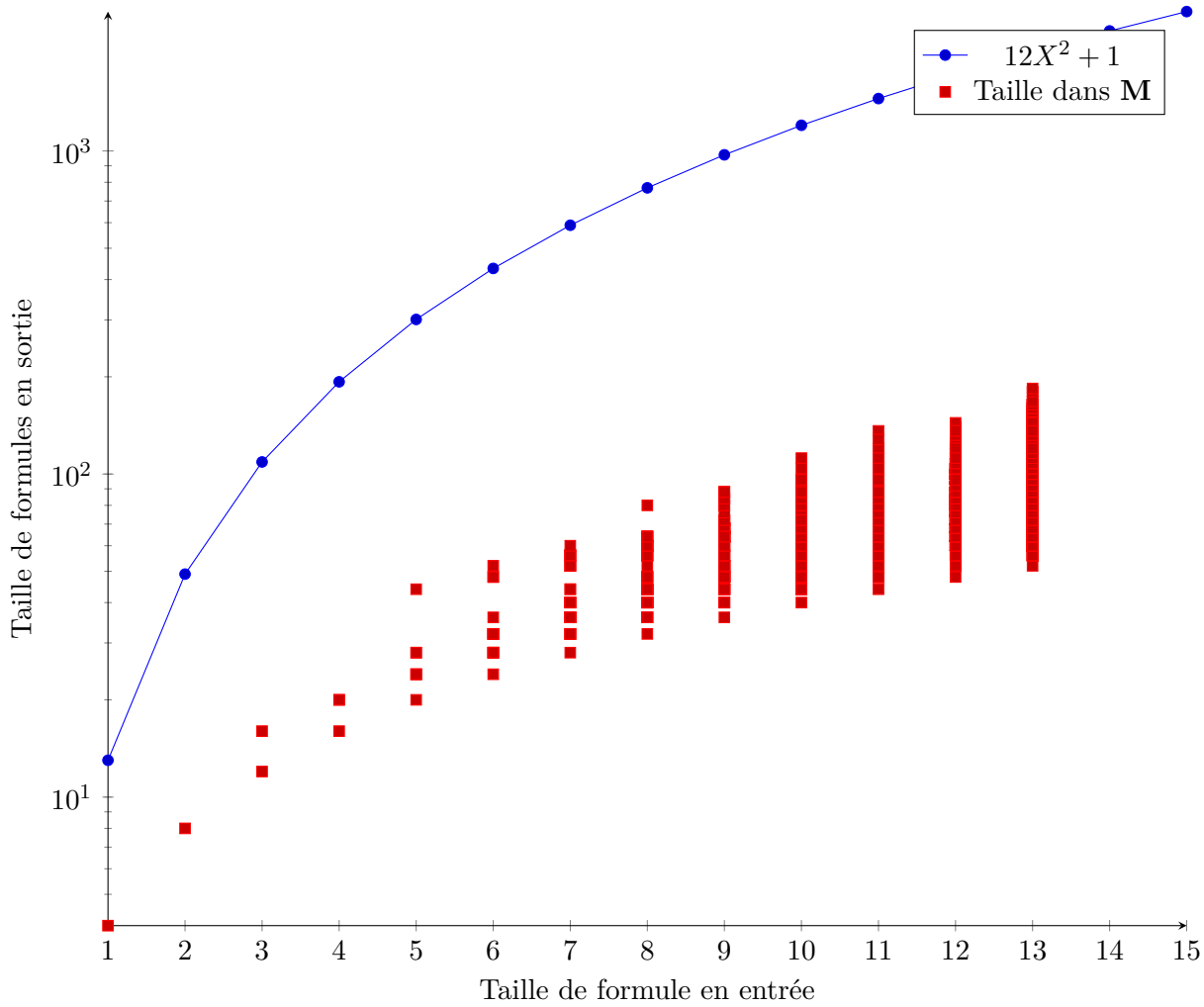


FIGURE 6.4 – Comparaison de la taille de termes de $\mathbf{M}$ et de $\mathbf{M}_5$.

Proposition 130. Les NFSs primaux $\mathbf{S}$, $\mathbf{S}^{\mathrm{d}}$, and $\mathbf{M}$ sont équivalents deux-à-deux, c'est-à-dire que $\mathbf{S} \sim \mathbf{S}^{\mathrm{d}} \sim \mathbf{M}$.

Démonstration. Considérons les équivalences

$$x \uparrow y \equiv \mathtt{m}(\neg x, 1, \neg y)$$

et

$$\mathtt{m}(x, y, z) \equiv (y \uparrow z) \uparrow (x \uparrow ((y \uparrow 1) \uparrow (z \uparrow 1)))$$

(voir l'Exemple 126). Remarquons que

$$|\mathtt{m}(\neg x, 1, \neg y)|_x = |\mathtt{m}(\neg x, 1, \neg y)|_y = 1,$$

et que

$$|(y \uparrow z) \uparrow (x \uparrow ((y \uparrow 1) \uparrow (z \uparrow 1)))|_x = 1.$$

Par conséquent, $\mathbf{S} \sqsupseteq_\exists \mathbf{M}$ et $\mathbf{M} \sqsupseteq_\exists \mathbf{M}$. Remarquons également que μ et $\uparrow$ sont toutes deux des fonctions symétriques. D'après la Proposition 125 il suit que $\mathbf{M} \sim \mathbf{S}$. Un raisonnement dual permet de prouver que $\mathbf{S}^{\mathrm{d}} \sim \mathbf{M}$. □

Proposition 131 (Système de décomposition médiane [Mar09b, Théorème 17])**.** Soit f une fonction Booléenne monotone, représentée par le terme $\mathtt{f}$. Alors, pour tout $k \in \{1, \ldots, \mathrm{ar}(\mathtt{f})\}$,

$$f(x_1, \ldots, x_{\mathrm{ar}(f)}) = \mu(f(x_1, \ldots, x_{\mathrm{ar}(f)})\{0/x_k\}, x_k, f(x_1, \ldots, x_{\mathrm{ar}(f)})\{1/x_k\}).$$

Le terme de droite de l'équivalence ci-dessus est la *décomposition médiane* de f suivant la *variable pivot* x_k.

Les sous-termes $\mathtt{f}(x_1, \ldots, x_{\mathrm{ar}(\mathtt{f})})\{c/x_k\}$, pour $c \in \mathbb{B}$, qui apparaissent dans la décomposition médiane induisent des fonctions monotones, si f l'est au départ. Lorsque nous appliquons le système de décomposition médiane de manière récursive sur les sous-termes $\mathtt{f}(x_1, \ldots, x_{\mathrm{ar}(\mathtt{f})})\{c/x_k\}$, pour $c \in \mathbb{B}$, en choisissant une nouvelle variable pivot à chaque étape, nous produisons un terme de $\mathbf{M}$ qui représente f. Remarquons que la première variable choisie apparaît seulement une fois dans ce terme.

Cette méthode de construction d'une représentation médiane d'une fonction Booléenne monotone a été présentée sous la forme d'un algorithme dans [CLMW11], que nous avons reproduit dans le chapitre précédent : algorithme 1. Cet algorithme fut ensuite adapté à des fonctions Booléennes arbitraires grâce à l'utilisation de l'encodage d'une fonction non-monotone en une fonction monotone ayant deux fois plus de variables.

Exemple 132. La fonction $f := [(x \wedge y) \wedge z]$ est monotone. D'après la décomposition médiane appliquée avec x comme variable pivot, nous avons l'équivalence

$$\mathtt{f}(x, y, z) \equiv \mathtt{m}(\mathtt{f}(0, y, z), x, \mathtt{f}(1, y, z)).$$

Après avoir décomposé les sous-termes en $\mathtt{f}$ restants $\mathtt{f}(0, y, z)$ et $\mathtt{f}(1, y, z)$ suivant les variables y et z, nous obtenons l'équivalence de conversion suivante :

$$\mathtt{f}(x, y, z) \equiv \mathtt{m}(\mathtt{m}(\mathtt{m}(0, z, 0), y, \mathtt{m}(0, z, 0)), x, \mathtt{m}(\mathtt{m}(0, z, 0), y, \mathtt{m}(0, z, 1)))$$

dans laquelle x n'apparaît qu'une seule fois.

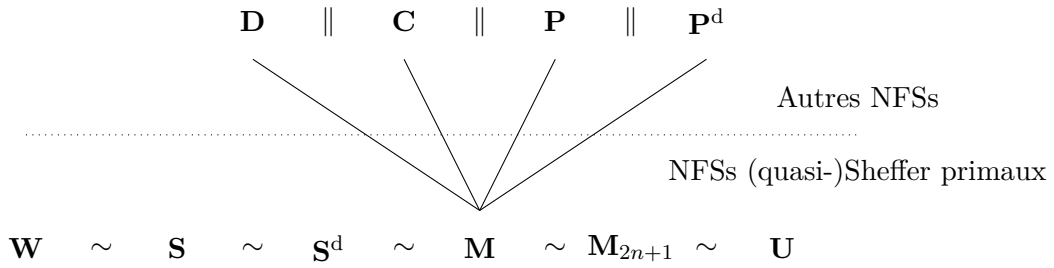


FIGURE 6.5 – Semi-treillis des NFSs primaux ordonnés par $\preceq$.

Remarquons d'ailleurs que la partie droite de cette équivalence peut être simplifiée encore plus, grâce au Système 77, en l'équivalence

$$\mathtt{f}(x, y, z) \equiv \mathtt{m}(0, x, \mathtt{m}(0, y, z)).$$

■

Proposition 133. Pour tout $n \geq 1$, les NFSs primaux $\mathbf{M}_{2n+1}$ and $\mathbf{M}$ sont équivalents ; autrement dit, $\mathbf{M}_{2n+1} \sim \mathbf{M}$.

Démonstration. D'après le système de décomposition médiane ainsi que le fait que μ_{2n+1} est une fonction symétrique et monotone, d'après la Proposition 125, nous concluons que $\mathbf{M} \preceq \mathbf{M}_{2n+1}$. D'après la Proposition 125 à nouveau ainsi que l'équivalence $\mathtt{m}(x, y, z) \equiv \mathtt{m}_{2n+1}(z, x^n, y^n)$, nous concluons que $\mathbf{M}_{2n+1} \preceq \mathbf{M}$. □

Les Propositions 129, 130, et 133, ainsi que le Théorème 115, nous donnent la classification des NFSs primaux de la Figure 6.5. Cette classification peut être améliorée de plusieurs manières :

— les connecteurs considérés sont d'arité minimale. Que dire des NFSs générés par des connecteurs d'arité quelconque ?
— Les connecteurs considérés sont interprétés comme des générateurs de clones particuliers. Que dire des autres générateurs ?
— Existe-t-il d'autres étages à cette classification ?

Dans la section suivante, nous répondons à ces questions par les théorèmes 135 et 137 et la proposition 148, sauf la dernière, qui constitue un travail en cours.

6.3 Optimalité pour les NFSs monotones

Dans cette section, nous montrons que les NFSs monotones optimaux sont exactement ceux qui reposent sur l'utilisation d'un seul connecteur ou d'un seul connecteur et de la négation. Pour ce faire, nous montrons, tout d'abord, que le NFS médian $\mathbf{M}$ est optimal parmi les NFSs monotones (Théorème 135). Puis, en utilisant des réductions entre NFSs adéquates, nous montrons, cas par cas, que chaque NFS monotone qui repose sur l'utilisation d'un seul connecteur ou d'un seul connecteur et de la négation est au moins aussi efficace que le NFS médian $\mathbf{M}$.

6.3.1 Optimalité pour la forme normale médiane

Dans cette section, nous étendons les résultats de la section précédentes en montrant que $\mathbf{M}$ est *optimal* parmi les NFSs monotones.

Proposition 134. Pour tout [a] pseudo-monotone, nous avons $\mathbf{M} \preceq T(\mathtt{a})$.

Démonstration. Puisque [a] est pseudo-monotone, alors il existe une fonction monotone [g] et des littéraux $l_i \in \{x_i, \neg(x_i)\}$, $1 \leq i \leq n$ tels que $\mathtt{a}(x_1, \ldots, x_n) \equiv \mathtt{g}(l_1, \ldots, l_n)$. En appliquant le système de décomposition médiane sur g, en choisissant la première variable pivot de manières différentes, nous pouvons produire n terms $t_i \in T(\mathtt{m})$ tels que $t_i \equiv \mathtt{g}(x_1, \ldots, x_n)$ et $|t_i|_{x_i} = 1$. Définissons $t'_i := t_i\{l_1/x_1, \ldots, l_n/x_n\}$; alors, $t'_i \in T(\mathtt{m}\neg)$, $t'_i \equiv \mathtt{a}(x_1, \ldots, x_n)$ et $|t'_i|_{x_i} = 1$. En utilisant la propriété d'auto-dualité de la médiane μ pour propager les négations aux variables, nous obtenons que $T(\mathtt{a}) \sqsupseteq_\forall T(\mathtt{m}\neg) = \mathbf{M}$. Par la Proposition 125, $\mathbf{M} \preceq T(\mathtt{a})$. □

Théorème 135. **M** est optimale parmi les NFSs monotones.

Démonstration. Tout d'abord, nous montrons, par récurrence sur le nombre de connecteurs n, que, pour tout ensemble de termes non-redondant $T(\mathtt{a}_1 \cdots \mathtt{a}_n)$ dans lequel chaque $[\mathtt{a}_i]$ est une fonction pseudo-monotone, nous avons l'inégalité

$$\mathbf{M} \preceq T(\mathtt{a}_1 \cdots \mathtt{a}_n).$$

Si $n = 1$, alors l'inégalité est vraie d'après la Proposition 134.

Supposons à présent que l'hypothèse de récurrence soit vraie pour tout entier naturel plus petit que n. Nous montrons que l'inégalité est vraie pour n. Soit $T_i := T(\mathtt{a}_1 \cdots \mathtt{a}_i)$, pour $i \leq n$.

Étant donnée une fonction f, considérons un terme $t \in T_n$ tel que $C_{T_n}(f) = |t|$ et tel que $[t] = f$ (t représente f). Le terme t peut être représenté de la manière suivante :

$$t'\{t_1/x_1, \ldots, t_j/x_j\}$$

dans laquelle t' est un terme de T_{n-1} d'arité j, pour un certain entier j, et $t_i \in T(\mathtt{a}_n)$ pour tout $i \leq j$.

Puisque t est minimal, nous avons que $C_{T_{n-1}}([t']) = |t'|$ et $C_{T(\mathtt{a}_n)}([t_i]) = |t_i|$. Sinon, cela contredirait le fait que t est un terme de T_n de taille minimale qui représente f. De plus,

$$C_{T_n}([t]) = C_{T_{n-1}}([t']) + \Sigma_{i=1}^{j} C_{T(\mathtt{a}_n)}([t_i])$$

par non-redondance.

D'après l'hypothèse de récurrence, $\mathbf{M} \preceq T_{n-1}$. Par définition, il existe donc un polynôme P tel que

$$C_{\mathbf{M}}([t']) \leq P(C_{T_{n-1}}([t'])).$$

Puisque la taille du terme minimal dans **M** équivalent à t' est borné par $P(C_{T_{n-1}}([t']))$, nous savons alors qu'il ne contient pas plus de $3P(C_{T_{n-1}}([t']))$ feuilles, car m est un connecteur ternaire. Par conséquent, $j \leq 3P(C_{T_{n-1}}([t']))$.

D'après la Proposition 134, il existe un polynôme Q tel que pour tout $\forall i,\ 1 \leq i \leq j$,

$$C_{\mathbf{M}}([t_i]) \leq Q(C_{T(\mathtt{a}_n)}([t_i])).$$

Par conséquent,

$$\begin{aligned} C_{\mathbf{M}}([t]) &\leq C_{\mathbf{M}}([t']) + \Sigma_{i=1}^{j} C_{\mathbf{M}}([t_i]) \\ &\leq P(C_{T_{n-1}}([t'])) + 3P(C_{T_{n-1}}([t'])) \max_i Q(C_{T(\alpha_n)}([t_i])) \\ &\leq P(C_{T_n}([t])) + 3P(C_{T_n}([t]))Q(C_{T_n}([t])) \\ &= R(C_{T_n}([t])) \end{aligned}$$

avec $R := P + 3P \cdot Q$. La dernière inégalité provient du fait que $C_{T_{n-1}}([t']) \leq C_{T_n}([t])$, que $C_{T(\mathtt{a}_n)}([t_i]) \leq C_{T_n}([t])$ pour tout i, et que P et Q sont des polynômes à coefficients positifs, et donc que les fonctions polynomiales induites par P et Q sont monotones croissantes.

Nous avons montré que $\mathbf{M} \preceq T(\mathtt{a}_1 \cdots \mathtt{a}_n)$ pour tout ensemble de termes non-redondant $T(\mathtt{a}_1 \cdots \mathtt{a}_n)$ avec les connecteurs pseudo-monotones $\mathtt{a}_1, \ldots, \mathtt{a}_n$. Par conséquent, nous obtenons ce résultat pour tout NFS monotone. □

6.3.2 Optimalité des NFSs monotones Sheffer et quasi-Sheffer

Dans cette section, nous généralisons les résultats obtenus dans la Section 6.2.4, qui traitait des seuls NFSs primaux, en montrant que tout NFSs monotone Sheffer ou quasi-Sheffer est optimal.

Lemme 136. Pour tout NFS $T(\mathtt{a})$, $T(\mathtt{a}) \sim T(\mathtt{a}\neg)$.

Démonstration. Considérons un NFS $T(\mathtt{a})$.

Un terme de $T(\mathtt{a})$ est aussi un terme de $T(\mathtt{a}\neg)$: donc $T(\mathtt{a}\neg) \preceq T(\mathtt{a})$.

D'après le Fait 102, $[T(\mathtt{a})] = \mathcal{C}([\mathtt{a}]) \circ I$. La fonction $[\mathtt{a}]$ est nécessairement non-monotone et non-constante, car si $[\mathtt{a}] \in M$, alors

$$[T(\mathtt{a})] = \mathcal{C}([\mathtt{a}]) \circ I \subseteq M \circ I = M \subsetneq \Omega$$

et $T(\mathtt{a})$ n'est pas un NFS. Donc, il existe une suite de constantes $c_1, \ldots, c_{\mathrm{ar}(\mathtt{a})-1}$ et une permutation π telles que

$$\mathtt{a}(\pi(c_1, \ldots, c_{\mathrm{ar}(\mathtt{a})-1}, x)) \equiv \neg x.$$

Ceci met en évidence l'existence d'une réduction $T(\mathtt{a}\neg) \sqsupseteq T(\mathtt{a})$. □

Théorème 137. Tous les NFSs monotones Sheffer et quasi-Sheffer sont optimaux.

Démonstration. D'après le Lemme 136, il suffit de considérer des ensembles de termes de la forme $T(\mathtt{a}\neg)$ tels que $[T(\mathtt{a}\neg)] = \Omega$ et de prouver, pour chacun, que $T(\mathtt{a}\neg) \sim \mathbf{M}$. D'après le Théorème 135, nous savons déjà que $\mathbf{M} \preceq T(\mathtt{a}\neg)$. Il nous reste à démontrer que $T(\mathtt{a}\neg) \preceq \mathbf{M}$. Cette inégalité dépend directement de la nature de la fonction $[\mathtt{a}]$. D'après le Fait 102, nous devons considérer les fonctions $[\mathtt{a}]$ qui satisfont

$$\Omega = [T(\mathtt{a}\neg)] = \mathcal{C}([\mathtt{a}]) \circ \mathcal{C}(\neg) \circ I = \mathcal{C}([\mathtt{a}]) \circ \Omega(1).$$

Les clones $\mathcal{C}$ qui satisfont l'égalité $\mathcal{C} \circ \Omega(1) = \Omega$ peuvent tous être lus dans la table de composition de clones de [CFL06], que nous avons reproduite dans la table 3.3. Ce sont les clones suivants : Ω, T_0, T_1, T_c, M, M_0, M_1, M_c, S, S_c, SM, et, pour $k = 2, \ldots, \infty$, U_k, MU_k, $T_\mathrm{c}U_k$, $M_\mathrm{c}U_k$, W_k, MW_k, $T_\mathrm{c}W_k$, $M_\mathrm{c}W_k$. Ainsi, nous devons considérer des fonctions $[\mathtt{a}]$ qui sont des générateurs d'un de ces clones listés ci-dessus.

Notons que les clones M, M_0, M_1, MU_k, MW_k ne sont pas générés par une fonction unique : par conséquent, nous n'avons pas besoin de les considérer. En particulier, certains de ces clones nécessitent l'utilisation de fonctions constantes, tels que M_0 et M_1, qui sont déjà, de toutes manières, autorisées dans nos formules grâce à $\Omega(1)$, qui les contient.

Dans la section suivante, nous allons établir, pour chaque clone considéré $\mathcal{C}$, une proposition de la forme

« *Si* $\mathcal{C}([\mathtt{a}]) = \mathcal{C}$, *alors* $T(\mathtt{a}\neg) \preceq \mathbf{M}$ ».

Plus précisément, les propositions et les ensembles de clones sur lesquels elles portent sont les suivants :

- Proposition 140 pour SM, le clone des fonctions auto-duales et monotones,
- Proposition 142 pour $M_{\mathrm{c}}U_k$, $M_{\mathrm{c}}W_k$, pour $k = 2, \ldots, \infty$, les clones des fonctions monotones et séparantes de rang k,
- Proposition 144 pour M_{c}, le clone des fonctions monotones préservant les constantes,
- Proposition 145 pour U_k, $T_{\mathrm{c}}U_k$, W_k, $T_{\mathrm{c}}W_k$, pour $k = 2, \ldots, \infty$, les clones des fonctions séparantes,
- Proposition 147 pour Ω, T_0, T_1, T_{c}, S, S_{c}, les clones restants.

Nous donnons une illustration de ces différents ensembles de clones dans la Figure 6.6.

Le théorème suit de toutes ses propositions. □

6.3.3 Preuve d'optimalité

Dans cette sous-section, nous utiliserons les notations suivantes. Étant donné un tuple $\mathbf{x}$ et une permutation π, $\pi(\mathbf{x})$ indique le tuple obtenu en permutant les coordonnées de $\mathbf{x}$ selon π. Étant donné $b \in \mathbb{B}$ et un entier $k > 0$, soit b^k une notation succinte pour $b, \ldots, b$ avec k occurrences de b.

Le clone SM

Lemme 138. Soit $f \in SM$ une fonction d'arité $n \geq 2$ qui ne soit pas une projection, et soit $\mathbf{x}$ un point vrai minimal de f. Alors, il existe un autre point vrai $\mathbf{y}$ tel que $\delta(\mathbf{x}, \mathbf{y}) = n - 1$. De plus, la seule coordonnée commune de $\mathbf{x}$ et de $\mathbf{y}$ a pour valeur 1.

Démonstration. Soit $\mathbf{x}$ un point vrai minimal pour la fonction $f : \mathbb{B}^n \to \mathbb{B}$ de SM.

- Si $\mathrm{w}(\mathbf{x}) = 0$, alors f est constante (égale à 1) par monotonicité ; ce cas ne peut pas avoir lieu, car SM ne contient aucune fonction constante, qui ne sont pas auto-duales.
- Si $\mathrm{w}(\mathbf{x}) = 1$, alors f est une projection par monotonicité et auto-dualité.

Supposons donc que $\mathrm{w}(\mathbf{x}) \geq 2$, et, sans perte de généralité, que

$$\mathbf{x} = (1, 1^k, 0^l)$$

avec $k > 0$ et $k + l + 1 = n$. Le tuple

$$\mathbf{z} := (0, 1^k, 0^l)$$

est un point faux parce que $\mathbf{x}$ est un point vrai minimal. Ainsi, le tuple

$$\mathbf{y} := \overline{\mathbf{z}} = (1, 0^k, 1^l)$$

est un point vrai par auto-dualité. De plus, $\delta(\mathbf{x}, \mathbf{y}) = k + l = n - 1$, et la coordonnée commune de $\mathbf{x}$ et de $\mathbf{y}$ est 1. □

Nous illustrons la construction d'un tel point $\mathbf{y}$ dans l'exemple 139 ci-dessous.

Exemple 139. Considérons la fonction $f \in SM$ définie par

$$f := \mu(\mu(x, y, z), x, t).$$

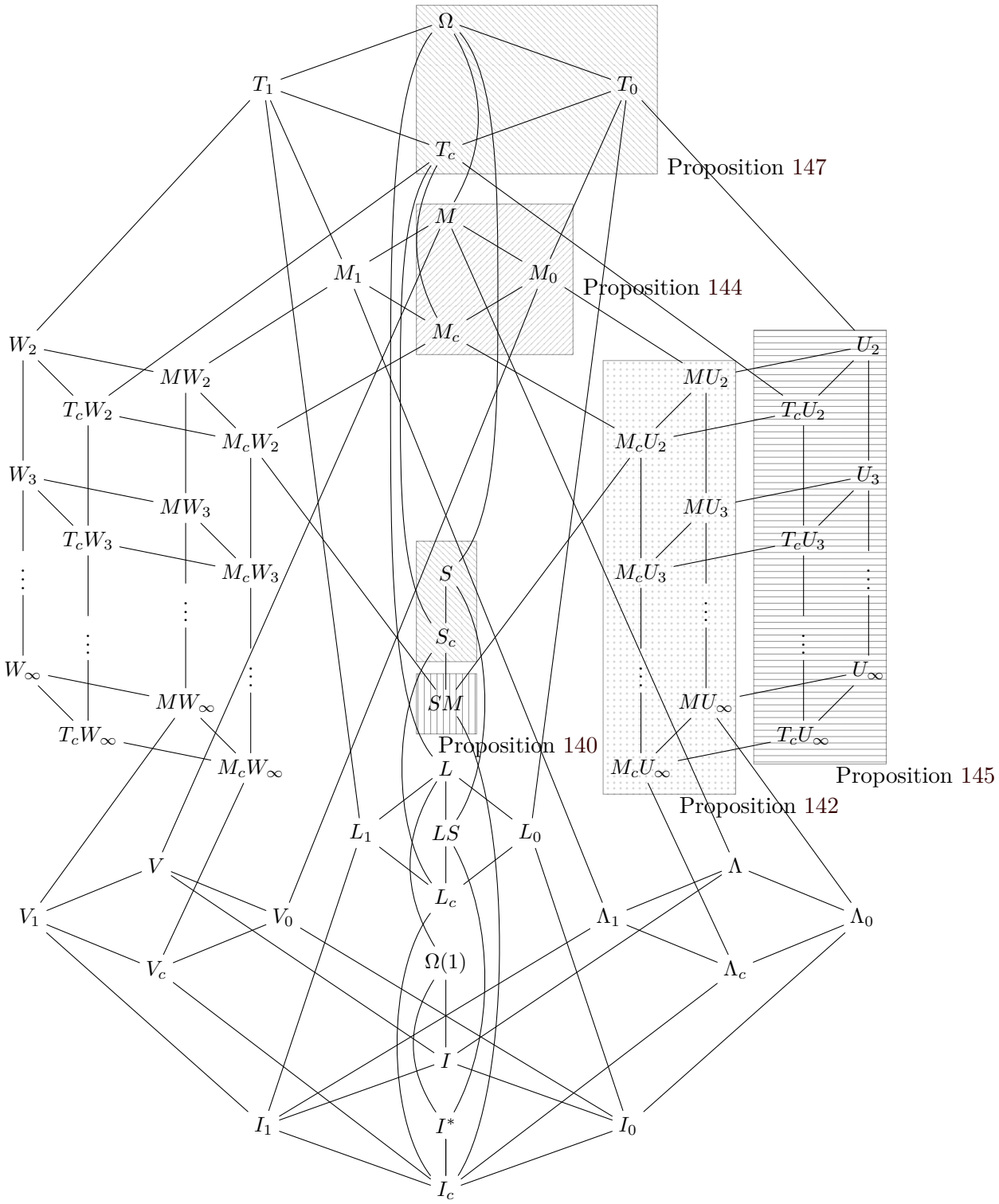


FIGURE 6.6 – Décomposition de la preuve en fonction des clones considérés, représentée sur le treillis de Post. Le théorème est démontré pour la partie gauche de manière duale.

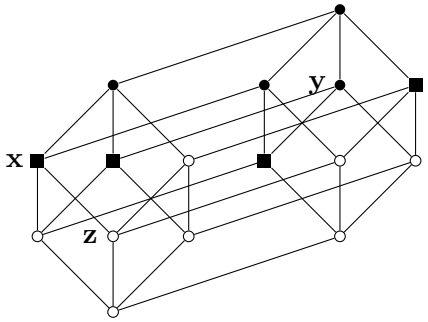


FIGURE 6.7 – Un diagramme de Hasse pour la fonction $\mu(\mu(x,y,z),x,t)$. Les points vrais minimaux sont indiqués par des carrés pleins.

f est monotone et auto-duale. Nous la représentons par le diagramme de Hasse de la figure 6.7. Nous pouvons directement lire, sur ce diagramme, les points vrais minimaux de f, qui sont indiqués par des carrés pleins, ■. Ce sont les points

$$\{(1,1,0,0),(1,0,1,0),(1,0,0,1),(0,1,1,1)\}.$$

Tous ces points sont de poids supérieur à 2. Posons $\mathbf{x} = (1,1,0,0)$. Par la construction précédente, nous avons $\mathbf{z} = (0,1,0,0)$, et $\mathbf{y} = \overline{\mathbf{z}} = (1,0,1,1)$ qui est un point vrai pour f différent de $\mathbf{x}$, à distance $\delta(\mathbf{x},\mathbf{y}) = 3$ de $\mathbf{x}$, et dont la seule coordonnée commune avec $\mathbf{x}$ est 1. ■

Proposition 140. Si $\mathtt{a}$ est un connecteur tel que $\mathcal{C}([\mathtt{a}]) = SM$, alors $T(\mathtt{a}\neg) \preceq \mathbf{M}$.

Démonstration. Notons $a := [\mathtt{a}]$. Remarquons qu'une fonction auto-duale et monotone est soit une projection, soit possède au moins trois arguments essentiels. Puisque a génère le clone SM, elle ne peut être une projection : par conséquent, $\mathrm{ar}(a) \geq 3$. Puisque a et monotone et auto-duale, nous pouvons séparer ses points vrais et ses points faux en deux ensembles de même cardinalité (par auto-dualité) tels qu'aucun point vrai de a n'est couvert par un point faux de a (par monotonicité). Nous illustrons cette propriété dans la figure 6.8, qui est une représentation schématique d'une séparation entre les ensembles de points vrais et faux d'une fonction monotone et auto-duale. Le treillis est représenté par un diagramme de Hasse schématisé.

D'après le Lemme 138, il existe deux points vrais $\mathbf{x},\mathbf{y}$ tel que $\mathbf{x}$ est minimal, à distance $\mathrm{ar}(a)-1$; ils ont exactement une seule coordonnée commune, égale à 1. Donc il existe une permutation π telle que $\pi(\mathbf{x}) = (1,1^k,0^l)$ et $\pi(\mathbf{y}) = (1,0^k,1^l)$, pour certains entiers k,l tels que $1+k+l = \mathrm{ar}(a)$.

Soit a' la fonction ternaire

$$a' := a(\pi^{-1}(x_1,x_2^k,x_3^l)).$$

Puisque a' est obtenue à partir de a en composant celle-ci avec des projections, alors $a' \in SM$.

Nous allons montrer que $a' = \mu$ en en déterminant certains points. Nous avons

$$a'(1,1,0) = a(\mathbf{x}) = 1$$

et

$$a'(1,0,1) = a(\mathbf{y}) = 1.$$

Puisque $\mathbf{x}$ est un point vrai minimal de a et que

$$\pi^{-1}(1,0,0) < \mathbf{x},$$

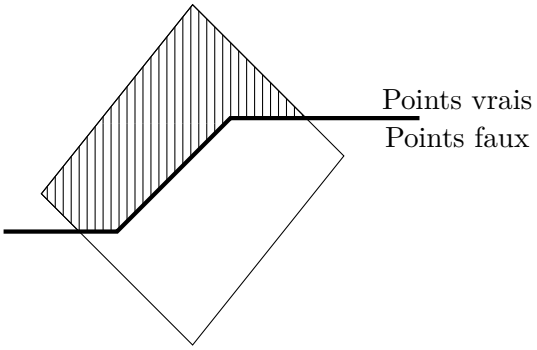


FIGURE 6.8 – Séparation des points vrais et faux pour une fonction monotone et auto-duale.

nous avons aussi

$$a'(1,0,0) = a(\pi^{-1}(1,0,0)) = 0.$$

De l'auto-dualité de a' nous obtenons

$$a'(0,0,1) = a'(0,1,0) = 0 \text{ et } a'(0,1,1) = 1,$$

et de la monotonicité de a' nous obtenons

$$a'(0,0,0) = 0 \text{ et } a'(1,1,1) = 1.$$

Donc $a' = \mu$. Rappelons que la médiane μ est symétrique, ce qui nous donne la propriété de symétrie « partielle » suivante pour $\mathtt{a}$:

$$\mathtt{a}(x_1, x_2^k, x_3^l) \equiv \mathtt{a}(x_2, x_1^k, x_3^l) \equiv \mathtt{a}(x_3, x_1^k, x_2^l).$$

Cela signifie que nous avons $\mathbf{M} \sqsupseteq_\forall T(\mathtt{a}\neg)$ avec, suivant les notations de l'énoncé de la définition des réductions, $t_1 = \mathtt{a}(x_1, x_2^k, x_3^l)$, $t_2 = \mathtt{a}(x_2, x_1^k, x_3^l)$, et $t_3 = \mathtt{a}(x_3, x_1^k, x_2^l)$. Ainsi, d'après la Proposition 125 nous concluons que $T(\mathtt{a}\neg) \preceq \mathbf{M}$. □

Les clones M_cU_k et M_cW_k

Lemme 141. Tout générateur d'un clone $\mathcal{C}$ qui vérifie $\Lambda \subsetneq \mathcal{C} \subseteq M$, possède au moins deux points vrais minimaux.

Démonstration. Les fonctions monotones qui ont moins de deux points vrais minimaux sont les constantes, les projections, ou les conjonctions de variables, et ne peuvent donc pas générer $\mathcal{C}$. □

Nous rappelons que u and w sont les générateurs d'arité minimale de M_cU_∞ et de M_cW_∞, respectivement.

Proposition 142. Si $\mathtt{a}$ est un connecteur tel que $\mathcal{C}([\mathtt{a}]) = M_cU_k$ ou tel que $\mathcal{C}([\mathtt{a}]) = M_cW_k$ pour un certain $k \in \{2, \ldots, \infty\}$, alors $T(\mathtt{a}\neg) \preceq \mathbf{M}$.

Démonstration. Notons $a := [\mathtt{a}]$. Nous nous concentrons sur le cas $\mathcal{C}(a) = M_cU_k$ pour un certain $k \in \{2, \ldots, \infty\}$. Le cas dual, pour M_cW_k, peut être démontré de manière similaire.

Tout d'abord, rappelons que, d'après la Proposition 129,

$$\mathbf{U} \sim \mathbf{W} \sim \mathbf{M}.$$

Par conséquent, il nous suffit de montrer que $T(\mathtt{a}\neg) \preceq \mathbf{U}$ ou que $T(\mathtt{a}\neg) \preceq \mathbf{W}$ ou que $T(\mathtt{a}\neg) \preceq \mathbf{M}$.

D'après le Lemme 141, il existe deux points vrais minimaux et incomparables $\mathbf{x}, \mathbf{y}$ pour a. Puisque $a \in U_2$, c'est-à-dire qu'elle est 1-séparatrice de rang 2, par définition ces deux points ont une coordonnée en commun, qui a la valeur 1. À une permutation près, nous supposons sans perte de généralité que

$$\mathbf{x} = (1^k, 1^l, 1, 0^m, 0^n)$$

et que

$$\mathbf{y} = (0^k, 1^l, 1, 1^m, 0^n)$$

pour certains entiers k, l, m, n tels que $k > 0$, $m > 0$, $l \geq 0$, $n \geq 0$ et $1 + k + l + m + n = \mathrm{ar}(a)$.

Soit α' la fonction ternaire définie par

$$a' := [a(x_1, 1^{k+l-1}, x_3, x_2^m, 0^n)].$$

Notons que a' est monotone, car elle a été obtenue à partir de la fonction monotone a par identification de variables et substitution de variables par des constantes. Autrement dit,

$$a' \in \mathcal{C}(a) \circ I \subseteq M \circ I = M.$$

Nous allons montrer que $a' \in \{u, \mu, w\}$. À partir des informations que nous avons pour l'instant, nous savons que la fonction a' satisfait les relations suivantes (voir le premier diagramme de Hasse de la Figure 6.9) :

— $a'(1,0,1) = a(\mathbf{x}) = 1$. Par monotonicité, $a'(1,1,1) = 1$. De plus, puisque $\mathbf{x}$ est un point vrai minimal de a, alors $(1,0,1)$ est aussi un point vrai minimal de a' ; donc, $a'(0,0,1) = a'(1,0,0) = a'(0,0,0) = 0$.

— $a'(0,1,1) = a(\mathbf{y}')$, avec $\mathbf{y}' = (0, 1^{k-1}, 1^l, 1, 1^m, 0^n)$. Puisque $\mathbf{y}' > \mathbf{y}$ et puisque $\mathbf{y}$ est un point vrai (minimal) de a, alors $a'(0,1,1) = a(\mathbf{y}') = 1$.

La valeur de a' sur les points $(0,1,0)$ et $(1,1,0)$ reste indéterminée, mais la monotonicité de a' implique que $a'(0,1,0) \leq a'(1,1,0)$. Ceci nous restreint à trois possibilités (voir les diagrammes de Hasse de la Figure 6.9) :

— $a'(0,1,0) = a'(1,1,0) = 0$, auquel cas $a' = u$;

— $a'(0,1,0) = 0$, $a'(1,1,0) = 1$, auquel cas $a' = \mu$;

— $a'(0,1,0) = a'(1,1,0) = 1$, auquel cas $a' = w$.

Considérons les conséquences de ces trois différentes possibilités pour a'.

1. Si $a' = u$, alors nous avons l'équivalence

   $$\mathtt{u}(x_1, x_2, x_3) \equiv \mathtt{a}(x_1, 1^{k+l-1}, x_3, x_2^m, 0^n).$$

   Nous rappelons que

   $$\mathtt{u}(x_1, x_2, x_3) \equiv (x_1 \vee x_2) \wedge x_3$$

   par définition, ce qui montre, par symétrie de $\vee$, que u est invariante par la transposition de ses deux premiers arguments, c'est-à-dire que

   $$\mathtt{u}(x_1, x_2, x_3) \equiv \mathtt{u}(x_2, x_1, x_3),$$

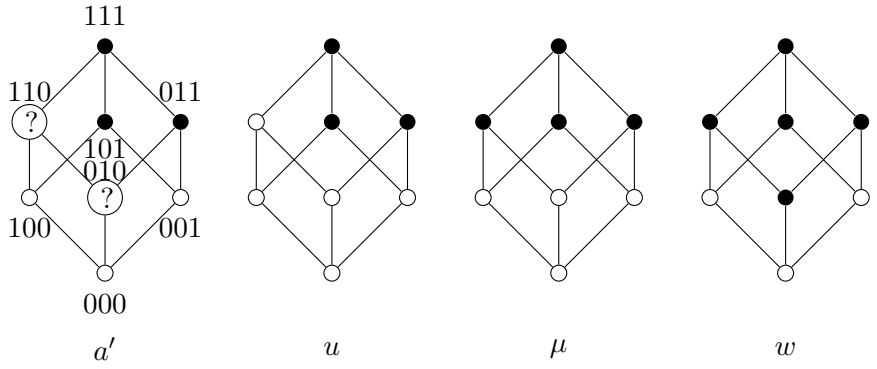


FIGURE 6.9 – Diagrammes de Hasse des différentes possibilités pour a'.

ce qui implique l'équivalence

$$\mathtt{u}(x_1, x_2, x_3) \equiv \mathtt{a}(x_2, 1^{k+l-1}, x_3, x_1^m, 0^n).$$

Nous avons donc obtenu, pour chaque variable $x_i, i \in \{1, 2, 3\}$, un terme de $T(a^\neg)$ équivalent à $\mathtt{u}(x_1, x_2, x_3)$ et qui ne contient qu'une seule occurrence de x_i. Par conséquent, $\mathbf{U} \sqsupseteq_\forall T(\mathtt{a}^\neg)$. Donc, d'après la Proposition 125 nous avons bien $T(\mathtt{a}^\neg) \preceq \mathbf{U}$.

2. Si $a' = \mu$, alors
$$\mathtt{m}(x_1, x_2, x_3) \equiv \mathtt{a}(x_1, 1^{k+l-1}, x_3, x_2^m, 0^n).$$
Puisque μ est une fonction symétrique, alors $T(\mathtt{m}^\neg) \sqsupseteq_\forall T(\mathtt{a}^\neg)$, d'après la Proposition 127, et donc $T(\mathtt{a}^\neg) \preceq \mathbf{M}$.
3. Si $a' = w$, alors en suivant le même raisonnement que dans le cas 1 (nous rappelons que $\mathtt{w}(x_1, x_2, x_3) \equiv (x_1 \wedge x_2) \vee x_3$ par définition) nous obtenons $T(\mathtt{a}^\neg) \preceq \mathbf{W}$. □

Le clone M_c

Lemme 143. Soit un connecteur $\mathtt{a}$ tel que $\mathcal{C}([\mathtt{a}]) = M_c$. Alors, il existe un terme linéaire binaire $t \in T(\mathtt{a})$ tel que $t \equiv x_1 \wedge x_2$.

Démonstration. Soit $\mathtt{a}$ un connecteur n-aire tel que $\mathcal{C}([\mathtt{a}]) = M_c$. Notons $a := [\mathtt{a}]$. Soit $\mathbf{a}$ un point vrai minimal de a. Nécessairement, $\mathrm{w}(\mathbf{a}) \geq 2$, car sinon a serait soit une projection ou une constante, et donc pas un générateur de M_c. Donc, il existe des indices $p, q \in \{1, \ldots, n\}$ avec $p \neq q$ tels que $a_p = a_q = 1$. Pour $i \in \{1, \ldots, n\}$, soit

$$t_i := \begin{cases} a_i & \text{si } i \notin \{p, q\}, \\ x_1 & \text{si } i = p, \\ x_2 & \text{si } i = q. \end{cases}$$

Alors, $\mathtt{a}\{t_1/x_1, \ldots, t_n/x_n\}$ est un terme linéaire binaire de $T(\mathtt{a})$ qui est équivalent à $x_1 \wedge x_2$. □

Proposition 144. Si $\mathtt{a}$ est un connecteur tel que $\mathcal{C}([\mathtt{a}]) = M_c$, alors $T(\mathtt{a}^\neg) \preceq \mathbf{M}$.

Démonstration. Soit a un générateur n-aire du clone M_c tel que $a := [\mathtt{a}]$ pour un certain connecteur $\mathtt{a}$. Considérons la fonction a' d'arité $n + 1$ définie par

$$a' := a(x_1, \ldots, x_n) \wedge x_{n+1}.$$

Notons que a' est monotone et préserve les constantes, et que pour tout point vrai $\mathbf{a}$ de a', $a_{n+1} = 1$; donc $a' \in M_{\mathrm{c}}U_\infty$. Puisqu'elle est un générateur de M_{c}, la fonction a n'est pas un membre de Λ ; par conséquent, nécessairement $a' \notin \Lambda$. Donc, a' est aussi un générateur de $M_{\mathrm{c}}U_\infty$, et d'après la proposition 142, $T(\mathtt{a}'\neg) \preceq \mathbf{M}$.

Soit t le terme binaire linéaire de $T(\mathtt{a})$ équivalent à $x_1 \wedge x_2$ obtenu par application du lemme 143, et soit

$$t' := t\{\mathtt{a}(x_1, \ldots, x_n)/x_1, x_{n+1}/x_2\}.$$

Alors, t' est un terme linéaire de $T(\mathtt{a})$, et, de plus, $\mathtt{a}'(x_1, \ldots, x_{n+1}) \equiv t'$. Par conséquent, $T(\mathtt{a}'\neg) \sqsupseteq T(\mathtt{a}\neg)$, grâce au terme t'. D'après la proposition 123, $T(\mathtt{a}\neg) \preceq T(\mathtt{a}'\neg)$. Par transitivité de la relation $\preceq$, nous concluons que $T(\mathtt{a}\neg) \preceq \mathbf{M}$. □

Les clones U_k, $T_{\mathrm{c}}U_k$, W_k, et $T_{\mathrm{c}}W_k$

Proposition 145. Si $[\mathtt{a}]$ est une fonction pseudo-monotone telle que

$$\mathcal{C}([\mathtt{a}]) \in \{U_k, T_{\mathrm{c}}U_k, W_k, T_{\mathrm{c}}W_k\}$$

pour un certain $k \in \{2, \ldots, \infty\}$, alors $T(\mathtt{a}\neg) \preceq \mathbf{M}$.

Démonstration. Soit a une fonction pseudo-monotone n-aire telle que $a := [\mathtt{a}]$ pour un certain connecteur $\mathtt{a}$. Supposons tout d'abord que

$$\mathcal{C}(a) \in \{U_k, T_{\mathrm{c}}U_k\}.$$

Puisque a n'est pas monotone, c'est-à-dire que $a \notin M$, le lemme 122 fournit un terme unaire linéaire $u \in T(\mathtt{a})$ avec $|u|_{\mathtt{a}} = 1$ tel que $u \equiv \neg x_1$: en d'autres termes, il est aisé de simuler la négation avec $\mathtt{a}$ de manière efficace.

Nous rappelons, d'après le fait 6, que a est pseudo-monotone si et seulement si il existe une fonction monotone $[g]\colon \mathbb{B}^n \to \mathbb{B}$ ainsi qu'un sous-ensemble $S \subseteq \{1, \ldots, n\}$ tel que pour tout $x_1, \ldots, x_n \in \mathbb{B}$,

$$\mathtt{a}(x_1, \ldots, x_n) = \mathtt{g}(l_1, \ldots, l_n),$$

où $l_i = x_i$ si $i \in S$ et $l_i = \neg(x_i)$ si $i \notin S$.

Alors, g est de la forme

$$\mathtt{a}(l'_1, \ldots, l'_n),$$

avec $l'_j \in \{x_j, \neg x_j\}$, pour tout $j \in \{1, \ldots, n\}$. Par conséquent, il existe un terme $t \in T(\mathtt{a})$ qui satisfait $|t|_{x_i} = 1$, pour tout i tel que $1 \leq i \leq n$, et $g = [t]$.

Notons que $g \notin V$. En effet, supposons qu'au contraire $g \in V$. Alors, a est soit une disjonction de variables niées ou une disjonction de variables niées et de variables non-niées. En choisissant bien $i_1, \ldots, i_n \in \{1, 2\}$, nous obtenons soit

$$\mathtt{a}(x_{i_1}, \ldots, x_{i_n}) \equiv \neg x_1 \vee \neg x_2 \equiv x_1 \uparrow x_2$$

soit

$$\mathtt{a}(x_{i_1}, \ldots, x_{i_n}) \equiv \neg x_1 \vee x_2.$$

Puisque $\mathcal{C}(\uparrow) = \Omega$ et que $\mathcal{C}([\neg x_1 \vee x_2]) = W_\infty$, alors $W_\infty \subseteq \mathcal{C}(a)$, ce qui contredit notre supposition initiale.

Si g a au moins deux points vrais minimaux, alors il existe $i, j \in \{1, \ldots, n\}$, $i < j$, et des constantes $c_1, \ldots, c_n$ telles que

$$x \vee y \equiv \mathtt{g}(c_1, \ldots, c_{i-1}, x, c_{i+1}, \ldots, c_{j-1}, y, c_{j+1}, \ldots, c_n).$$

Si g a un unique point vrai minimal, alors c'est une conjonction de variables. En utilisant les lois de De Morgan ainsi que la représentation donnée plus haut de $\neg$ et de g en des termes en $\mathtt{a}$, nous obtenons que les deux fonctions $\wedge$ et $\vee$ peuvent être obtenues comme les fonctions de termes de termes binaires $t \in T(\mathtt{a})$ tels que chaque variable n'apparaît qu'une seule fois dans t.

À présent, soit

$$h := g(x_1, \ldots, x_n) \vee x_{n+1}].$$

Puisque $g \notin V$, alors h est un générateur de $M_{\mathrm{c}}W_\infty$. De plus, par construction, $T(\mathtt{h}\neg)$ est linéairement réductible à $T(a\neg)$. D'après la Proposition 142, $T(\mathtt{a}\neg) \preceq \mathbf{M}$.

Les cas suivants, lorsque $\mathcal{C}(a) \in \{W_k, T_{\mathrm{c}}W_k\}$, peuvent être traités de manière similaire. □

Les clones Ω, T_0, T_1, T_{c}, S, et S_{c}

Lemme 146. Pour toute fonction $[\mathtt{a}] \notin M \cup L$, il existe un terme binaire linéaire $t \in T(\mathtt{a})$ tel que $t \equiv x_1 \wedge x_2$.

Démonstration. Soit un connecteur $\mathtt{a}$ tel que $[\mathtt{a}] \notin M \cup L$. Soit $P_{\mathtt{a}}$ une représentation de $[\mathtt{a}]$ sous forme normale polynomiale de taille minimale. Alors, $P_{\mathtt{a}}$ est de la forme

$$C_1 \oplus C_2 \oplus \ldots \oplus C_p,$$

où chaque sous-terme C_i est soit une variable, une constante, ou une conjonction de variables. Nécessairement, il existe au moins un C_i qui est une conjonction d'au moins deux variables, car sinon $[\mathtt{a}]$ serait une fonction linéaire. Soit C le sous-terme de taille minimale parmi les C_i ; C est une conjonction de variables. Sans perte de généralité, nous pouvons supposer que

$$C = x_1 \wedge x_2 \wedge \ldots \wedge x_k.$$

À présent, soit $\mathtt{a}'$ le terme obtenu à partir de $\mathtt{a}(x_1, \ldots, x_n)$ en substituant chaque occurrence de x_i par 1, pour $3 \leq i \leq k$, et en substituant chaque occurrence de x_j par 0, pour $j > k$. Le terme $\mathtt{a}'$ qui résulte de cette construction est linéaire et est équivalent à l'un des termes suivants :

$$\begin{aligned}
&x_1 \wedge x_2, \\
&(x_1 \wedge x_2) \oplus 1 && \equiv \neg x_1 \vee \neg x_2, \\
&(x_1 \wedge x_2) \oplus x_1 && \equiv x_1 \wedge \neg x_2, \\
&(x_1 \wedge x_2) \oplus (x_1 \oplus 1) && \equiv \neg x_1 \vee x_2, \\
&(x_1 \wedge x_2) \oplus x_2 && \equiv \neg x_1 \wedge x_2, \\
&(x_1 \wedge x_2) \oplus (x_2 \oplus 1) && \equiv x_1 \vee \neg x_2, \\
&(x_1 \wedge x_2) \oplus (x_1 \oplus x_2) && \equiv x_1 \vee x_2, \\
&(x_1 \wedge x_2) \oplus ((x_1 \oplus x_2) \oplus 1) && \equiv \neg x_1 \wedge \neg x_2.
\end{aligned}$$

Nous pouvons donc aisément obtenir la conjonction $x_1 \wedge x_2$ à partir de $\mathtt{a}'$ en niant des variables ou en niant le terme $\mathtt{a}'$ tout entier. Puisque $[\mathtt{a}]$ n'est pas monotone, le Lemme 122 fournit un terme linéaire $t \in T(\mathtt{a})$ qui représente la négation Booléenne. En utilisant à la fois $\mathtt{a}'$ and t, nous pouvons à présent construire un terme linéaire de $T(\mathtt{a})$ équivalent à $x_1 \wedge x_2$. □

Proposition 147. Si $[\mathtt{a}]$ est une fonction pseudo-monotone telle que $S_{\mathrm{c}} \subseteq \mathcal{C}([\mathtt{a}])$, alors $T(\mathtt{a}\neg) \preceq \mathbf{M}$.

Démonstration. Soit $[\mathtt{a}]$ une fonction pseudo-monotone n-aire telle que $S_{\mathrm{c}} \subseteq \mathcal{C}([\mathtt{a}])$, et définissons

$$a' := [a(x_1, \ldots, x_n) \wedge x_{n+1}].$$

La fonction a' est 1-séparante : en effet, x_{n+1} prend nécessairement la valeur 1 à chaque point vrai. Donc, $a' \in U_\infty$. Puisque la fonction a est un générateur d'un clone qui contient le clone S_{c}, elle n'est pas une fonction de $M \cup L$; par conséquent, $a' \notin M$. Donc, a' est soit un générateur de U_∞, soit un générateur de $T_{\mathrm{c}}U_\infty$. Il existe une réduction linéaire de $T(\mathtt{a}'\neg)$ vers $T(\mathtt{a}\neg)$, comme en témoigne le terme linéaire

$$t\{\mathtt{a}(x_1, \ldots, x_n)/x_1, x_{n+1}/x_2\} \in T(\mathtt{a}),$$

dans lequel $t \in T(\mathtt{a})$ est le terme binaire linéaire qui représente $\wedge$ fourni par le lemme 146 (souvenons-nous que $a \notin M \cup L$). Ainsi, $T(\mathtt{a}\neg) \preceq T(\mathtt{a}'\neg)$ par la proposition 123, ce qui nous permet d'atteindre le résultat désiré, car a' est pseudo-monotone, et $T(\mathtt{a}'\neg) \preceq \mathbf{M}$ par la proposition 145. □

6.4 Autres NFSs

Dans cette section, nous considérons les NFSs restants, c'est-à-dire les NFSs dont les connecteurs sont dans $V \cup L \cup \Lambda$ ainsi que ceux qui sont générés par une fonction qui n'est pas pseudo-monotone.

6.4.1 DNF, CNF, PNF généralisées

D'après la Proposition 108, nous pouvons nous concentrer seulement sur les NFSs qui ont au plus 3 connecteurs. Dans [CFL06] il a été démontré que

$$\mathbf{M} \prec \mathbf{C}, \mathbf{D}, \mathbf{P}, \mathbf{P}^{\mathrm{d}}.$$

Toutefois, les connecteurs qui y avaient été considérés étaient ceux d'arité minimale, c'est-à-dire la disjonction binaire $\vee$, la conjonction binaire $\wedge$, et la somme ternaire $\oplus_3$. En fait, ces résultats sont aussi vrais pour des connecteurs d'arité arbitraires. Étant donné un ensemble de termes de la forme $T = T(\mathtt{ab}\neg)$, nous vérifions, en utilisant la table de composition de clones de [CFL06], que les seules possibilités pour que T soit un NFS sont $[\mathtt{a}] \in \Lambda$, $[\mathtt{b}] \in V$, ou $[\mathtt{a}] \in V$, $[\mathtt{b}] \in \Lambda$. De manière similaire, pour qu'un ensemble de termes $T(\mathtt{ab})$ soit un NFS, les seules possibilités sont $[\mathtt{a}] \in L$, $[\mathtt{b}] \in \Lambda$, ou $[\mathtt{a}] \in L$, $[\mathtt{a}] \in V$.

La proposition suivante montre que tout NFS qui se base sur deux connecteurs, avec ou sans la négation, est équivalent à la forme normale conjonctive, disjonctive, polynomiale ou polynomiale duale.

Proposition 148. Pour tous connecteurs $\mathtt{a}$, $\mathtt{b}$, $\mathtt{c}$ tels que $\mathcal{C}([\mathtt{a}]) \circ I = L$, $\mathcal{C}([\mathtt{b}]) = \Lambda_{\mathrm{c}}$, et $\mathcal{C}([\mathtt{c}]) = V_{\mathrm{c}}$, nous avons $T(\mathtt{ab}) \sim \mathbf{P}$, $T(\mathtt{ac}) \sim \mathbf{P}^{\mathrm{d}}$, $T(\mathtt{bc}\neg) \sim \mathbf{C}$, $T(\mathtt{cb}\neg) \sim \mathbf{D}$.

Démonstration. Sans perte de généralité, nous pouvons supposer que $\mathtt{a}$, $\mathtt{b}$, et $\mathtt{c}$ ont pour arités respectives ℓ, m, et n, et que tous leurs arguments sont essentiels. Alors, $\ell \geq 2$, $m \geq 2$, $n \geq 2$, et

$$\begin{aligned}
&\mathtt{a}(x_1, \ldots, x_\ell) \equiv x_1 \oplus x_2 \oplus \ldots \oplus x_\ell \oplus c =: t_{\mathtt{a}} \in T(\oplus) \quad \text{pour un certain } c \in \mathbb{B},\\
&\mathtt{b}(x_1, \ldots, x_m) \equiv x_1 \wedge x_2 \wedge \ldots \wedge x_m =: t_{\mathtt{b}} \in T(\wedge),\\
&\mathtt{c}(x_1, \ldots, x_n) \equiv x_1 \vee x_2 \vee \ldots \vee x_n =: t_{\mathtt{c}} \in T(\vee).
\end{aligned}$$

De plus,

$$x_1 \wedge x_2 \equiv \mathtt{b}(x_1, x_2, 1, \ldots, 1) =: t_\wedge \in T(\mathtt{b}),$$
$$x_1 \vee x_2 \equiv \mathtt{b}(x_1, x_2, 0, \ldots, 0) =: t_\vee \in T(\mathtt{c}).$$

Afin de représenter $\oplus$ comme un terme $t_\oplus \in T(\mathtt{a})$, nous devons considérer différents cas :

$$t_\oplus := \begin{cases} \mathtt{a}(x_1, x_2, 0, \ldots, 0, c), & \text{si } \ell \geq 3, \\ \mathtt{a}(x_1, x_2), & \text{si } \ell = 2 \text{ et } c = 0, \\ \mathtt{a}(\mathtt{a}(x_1, x_2), 0), & \text{si } \ell = 2 \text{ et } c = 1. \end{cases}$$

Dans chaque cas, $x_1 \oplus x_2 \equiv t_\oplus$.

Les termes $t_\mathtt{a}$, $t_\mathtt{b}$, $t_\mathtt{c}$, $t_\oplus$, $t_\wedge$, et $t_\vee$ sont tous linéaires ; nous pouvons à présent faire des transformations directes et sans changement de complexité entre $T(\mathtt{ab})$ et $\mathbf{P}$, entre $T(\mathtt{ac})$ et $\mathbf{P}^\mathrm{d}$, entre $T(\mathtt{bc}\neg)$ et $\mathbf{C}$, et entre $T(\mathtt{cb}\neg)$ et $\mathbf{D}$.

Par exemple, étant donné un terme $t \in T(\mathtt{ab})$, nous obtenons un terme équivalent $t' \in \mathbf{P}$ en remplaçant chaque sous-terme $\mathtt{b}t_1 \ldots t_m$ de t par $t_\mathtt{b}\{t_1/x_1, \ldots, t_m/x_m\}$ et chaque sous-terme $\mathtt{a}t_1 \ldots t_\ell$ par $t_\mathtt{a}\{t_1/x_1, \ldots, t_\ell/x_\ell\}$. Puisque les termes $t_\mathtt{a}$ et $t_\mathtt{b}$ sont linéaires et puisque $|t_\mathtt{a}|_\oplus = \ell$ et $|t_\mathtt{b}|_\wedge = m - 1$, alors

$$|t'| = |t'|_\oplus + |t'|_\wedge = \ell|t|_\mathtt{a} + (m-1)|t|_\mathtt{b} \leq \max(\ell, m-1)|t|.$$

Par conséquent, pour toute fonction f,

$$C_\mathbf{P}(f) \leq \max(\ell, m-1) C_{T(\mathtt{ab})}(f),$$

c'est-à-dire que $\mathbf{P} \preceq T(\mathtt{ab})$.

Dans l'autre sens, étant donné un terme $t \in \mathbf{P}$, nous pouvons obtenir un terme équivalent $t' \in T(\mathtt{ab})$ en remplaçant chaque sous-terme $\wedge(t_1, t_2)$ de t par $t_\wedge\{t_1/x_1, t_2/x_2\}$ et chaque sous-terme $\oplus(t_1, t_2)$ par $t_\oplus\{t_1/x_1, t_2/x_2\}$. Puisque les termes $t_\oplus$ et $t_\wedge$ sont linéaires et puisque $|t_\oplus|_\mathtt{a} \leq 2$ et $|t_\wedge|_\mathtt{b} = 1$, nous avons donc

$$|t'| = |t'|_\mathtt{a} + |t'|_\mathtt{b} \leq 2|t|_\oplus + |t|_\wedge \leq 2|t|.$$

Par conséquent, pour toute fonction f,

$$C_{T(\mathtt{ab})}(f) \leq 2 C_\mathbf{P}(f),$$

c'est-à-dire que $T(\mathtt{ab}) \preceq \mathbf{P}$.

Les autres équivalences peuvent être démontrées en utilisant des arguments similaires. □

Exemple 149. Par exemple, le terme

$$\wedge_5(x_1, x_2, x_3, x_4, x_5)$$

de taille 1 qui fait intervenir la conjonction 5-aire $\wedge_5$ est équivalent au terme

$$\wedge(x_1, \wedge(x_2, \wedge(x_3, \wedge(x_4, x_5))))$$

qui est de taille 4. De plus, le terme

$$\wedge(x_1, x_2)$$

est équivalent au terme

$$\wedge_5(x_1, x_2, 1, 1, 1).$$

D'après la Proposition 148, $\mathbf{C} = T(\wedge \vee \neg) \sim T(\wedge_5 \vee \neg)$. ■

6.4.2 NFSs non-monotones : considérations et conjectures

Le cas des NFSs non-monotones continue de nous échapper. Toutefois, nous conjecturons que les NFSs (quasi-)Sheffer non-monotones sont strictement plus efficaces que les autres NFSs.

Afin de motiver notre intuition, considérons l'ensemble de termes $\Sigma = T(\mathtt{s})$ dans lequel

$$\mathtt{s} := [(x_1 \wedge x_2) \ \vee \ (\neg x_1 \wedge x_3)].$$

Observons que $[\mathtt{s}]$ n'est pas pseudo-monotone, car $\mathtt{s}(x_1, 1, 0) \equiv x_1$ et $\mathtt{s}(x_1, 0, 1) \equiv \neg x_1$. De plus, les équivalences

$$\mathtt{s}(x_1, 0, 1) \equiv \neg x_1,$$
$$\mathtt{s}(x_1, x_2, 0) \equiv x_1 \wedge x_2$$

impliquent que Σ est un NFS Sheffer, car l'ensemble $\{\neg, \wedge\}$ est complet vis-à-vis de l'ensemble des fonctions Booléennes. Finalement, Σ est au moins aussi efficace que tout autre NFS Sheffer ou quasi-Sheffer.

Lemme 150. Pour tout ensemble de termes $T(\mathtt{a}\neg)$ (respectivement $T(\mathtt{a})$), nous avons

$$\Sigma \preceq T(\mathtt{a}\neg)$$

(respectivement $T(\mathtt{a})$).

Démonstration. La preuve repose sur le théorème d'expansion de Boole, aussi connu sous le nom de décomposition de Shannon [Sha49]). Pour tout connecteur $\mathtt{a}$ d'arité n,

$$\mathtt{a}(x_1, \ldots, x_n) \equiv t_j = \mathtt{s}(x_j, \mathtt{a}(x_1, \ldots, x_n)\{1/x_j\}, \mathtt{a}(x_1, \ldots, x_n)\{0/x_j\}).$$

Ainsi, $\forall j,\ |t_j|_{x_j} = 1$, $T(\mathtt{a}\neg) \sqsupseteq_\forall \Sigma$. D'après la Proposition 125, $\Sigma \preceq T(\mathtt{a}\neg)$. □

La réciproque semble toutefois peu probable, car $\mathtt{s}$ n'est ni croissante ni décroissante en x_1. Comme nous allons le voir, cela implique que x_1 doive apparaître plus d'une fois dans n'importe quel terme $t \in T(\mathtt{a}\neg)$ qui représente $\mathtt{s}$, si $[\mathtt{a}]$ est une fonction monotone, et ainsi que $\mathtt{a}$ doive apparaître plus d'une fois dans t.

Proposition 151. Soit $[\mathtt{a}]$ une fonction monotone. Si $t \in T(\mathtt{a}\neg)$ est un terme dans lequel la variable x_i apparaît exactement une fois, alors $[t]$ est soit croissante, soit décroissante en son i-ème argument.

Démonstration. La démonstration repose sur une induction sur la structure des termes. Si $t = x_i$, alors $[t] = e_i^{(n)}$ est une projection, qui est bien croissante en son i-ème argument.

Si $t = \neg t'$ pour un certain t' qui est tel que $[t']$ est croissante (respectivement décroissante) en son i-ème argument, alors $[t]$ est décroissante (respectivement croissante) en son i-ème argument.

Supposons à présent que $t = \mathtt{a}(t_1, \ldots, t_n)$ pour certains termes $t_1, \ldots, t_n \in T(\mathtt{a}\neg)$. Alors, x_i dans un des sous-termes $t_1, \ldots, t_n$ exactement. Supposons sans perte de généralité que x_i apparaît dans t_p pour un certain p. D'après l'hypothèse d'induction, $[t_p]$ est soit croissant soit décroissant en son i-ème argument. Soit $\mathbf{a} = (a_1, \ldots, a_n)$ et $\mathbf{b} = (a_1, \ldots, a_{i-1}, b_i, a_{i+1}, \ldots, a_n)$ avec $a_i \leq b_i$. Puisque pour tout $j \in [n] \setminus \{p\}$, la variable x_i n'apparaît pas dans t_j et ainsi $[t_j]$ ne dépend pas de son i-ème argument. Par conséquent, $[t_j](\mathbf{a}) = [t_j](\mathbf{b})$. Si $[t_p]$ est croissante en son i-ème argument, alors $[t_p](\mathbf{a}) \leq [t_p](\mathbf{b})$, et la monotonie de $[\mathtt{a}]$ implique que

$$\begin{aligned}[t](\mathbf{a}) &= \mathtt{a}([t_1](\mathbf{a}), \ldots, [t_p](\mathbf{a}), \ldots, [t_n](\mathbf{a})) \\ &\leq \mathtt{a}([t_1](\mathbf{b}), \ldots, [t_p](\mathbf{b}), \ldots, [t_n](\mathbf{b})) = [t](\mathbf{b}),\end{aligned}$$

et donc que $[t]$ est croissante en son i-ème argument. De manière similaire, si $[t_p]$ est décroissante en son i-ème argument, alors $[t]$ l'est aussi. □

Corollaire 152. Soient des connecteurs $\mathtt{a}$ et $\mathtt{b}$. Si α n'est pas pseudo-monotone mais que β est monotone, alors il n'existe pas de réduction universelle quasi-linéaire de $T(\alpha\neg)$ vers $T(\beta\neg)$.

Supposons à présent que $[\mathtt{b}]$ soit une fonction monotone et que $[\mathtt{a}]$ soit une fonction n-aire qui n'est pas pseudo-monotone. Alors, il existe un indice $i \in \{1, \ldots, n\}$ tel que $[\mathtt{a}]$ n'est ni croissante ni décroissante en son i-ème argument.

Soit t un terme de $T(\mathtt{b}\neg)$ tel que $t \equiv \mathtt{a}$. D'après la Proposition 151, nécessairement, nous avons $|t|_{x_i} > 1$. Considérons à présent les termes de $T(\mathtt{a})$ suivants : $s_1 := \mathtt{a}(x_1, \ldots, x_n)$, et pour $k \geq 1$,

$$s_{k+1} \quad := \quad \mathtt{a}(x_1, \ldots, x_{i-1}, s_k\{x_i/x_1, x_{i+1}/x_2, \ldots, x_{i+N(k)-1}/x_{N(k)}\}, x_{i+N(k)}, \ldots, x_{N(k+1)}),$$

où $N(k) := k \cdot (n-1) + 1$. Soit $f_k := [s_k]$, pour $k \geq 1$. Nous avons que $C_{T(\mathtt{a})}(f_k) \leq |s_k|_{\mathtt{a}} = k$. Nous pouvons obtenir un terme équivalent $s'_k \in T(\mathtt{a}\neg)$ en remplaçant dans s_k chaque sous-terme $\mathtt{a}(t_1, \ldots, t_n)$ par $t\{t_1/x_1, \ldots, t_n/x_n\}$. La taille des termes en résultant s'_k croît de manière exponentielle en k à cause de la variable qui est répétée, x_i, dans t.

Bien entendu, ce remplacement direct du connecteur $\mathtt{a}$ par le terme t ne produit pas nécessairement un terme de taille minimale de $T(\mathtt{b}\neg)$ qui représente f_k. Par conséquent, nous ne pouvons pas conclure quoi que ce soit sur $C_{T(\mathtt{b}\neg)}(f_k)$. Toutefois, ce résultat nous amène à formuler la conjecture suivante.

Conjecture 153. Si $\mathbf{A}$ est un NFS quasi-Sheffer non-monotone, alors $\mathbf{A} \prec \mathbf{M}$.

6.5 Conclusion

Dans ce chapitre, nous avons étudié la complexité de la représentation des fonctions Booléennes par des systèmes de formes normales (NFSs) de manière très générale, étendant ainsi les résultats précédemment obtenus sur l'efficacité des formes normales médianes pour représenter les fonctions Booléennes, par rapport aux formes normales disjonctives, conjonctive, polynomiales et polynomiales duales ([CFL06]). Nous avons défini une forme normale comme un ensemble de termes stratifiés par rapport à une certaine suite de connecteurs (avec donc de fortes conditions structurelles). Nous avons donné des moyens pour comparer efficacement les NFSs. Grâce à la théorie des clones, qui a orienté notre recherche de connecteurs adéquats, nous avons pu déterminer tous les NFSs possibles sous la condition de non-redondance, qui correspond à la recherche de NFSs utilisant le moins de connecteurs possibles. En particulier, nous avons montré que les résultats du théorème 115 ne dépendent pas du choix des connecteurs, et, en particulier, de leur arité, du moment que ces connecteurs sont pseudo-monotones (théorème 135). Nous avons démontré que les NFSs optimaux sont exactement ceux de la forme $T(\mathtt{a})$ ou $T(\neg\mathtt{a})$, où $[\mathtt{a}]$ est pseudo-monotone, c'est-à-dire où $[\mathtt{a}]$ est Sheffer ou quasi-Sheffer, respectivement. De plus, tous ces NFSs optimaux sont équivalents deux-à-deux.

Nous discutons des travaux futurs et des questions soulevées durant cette étude dans le chapitre 7.

7

Conclusion et travaux futurs

> Dans toute l'histoire de l'esprit humain l'étonnement a joué un rôle considérable et déployé de précieuses vertus. Non qu'il ait été suffisant ; mais il a été indispensable. C'est parce qu'un homme, ou plusieurs, se sont étonnés de certaines absurdités ou improbabilités choquantes admises autour d'eux que des progrès ont été faits dans la connaissance de la réalité.
>
> *Pour raison garder*, Jules Romains ([Rom61]).

Dans le Chapitre 5, nous avons discuté d'un formalisme fondé sur le connecteur médian `m` pour représenter de manière efficace les fonctions Booléennes monotones (ou non) ainsi que les fonctions latticielles sur des treillis distributifs. Pour cela, nous avons proposé une certaine notion de *formes normales médianes* définie comme étant un ensemble d'expressions médianes qui sont minimales par rapport à un ordre structurel sur les formules.

Nous avons également formalisé la tâche qui consiste à trouver des formules médianes ayant des représentations structurelles plus petites qu'une représentation donnée et avons étudié la complexité computationnelle de ce problème. Cette tâche, nous l'avons vu, est au plus modérément intractable. En fait, nous avons montré que le problème de décision correspondant, c'est-à-dire SMALLMED, se trouve soit dans Σ_2^{P}, soit dans co-NP, suivant que l'on place la représentation structurelle qui le borne parmi les entrées ou non. Par contre, la détermination de bornes inférieures est encore en cours d'investigation.

Par la suite, nous avons généralisé de manière naturelle ces résultats à des fonctions Booléennes arbitraires, c'est-à-dire en autorisant l'apparition de variables niées dans notre formalisme. Nous obtenons un résultat surprenant (Proposition 89) qui est que dans certains cas, l'utilisation de variables niées nous permet d'atteindre des représentations strictement plus petites que si nous n'avions pas autorisé la négation.

Une autre généralisation naturelle de la question de la représentation des fonctions Booléennes par les formes normales est celle de l'utilisation d'autres opérateurs, ce que nous avons étudié dans le Chapitre 6.

Nous avons étendu les résultats de [CFL06] sur la classification des NFS. En particulier, nous avons montré que les résultats sur les NFSs primaux ne dépendent pas du choix du connecteur

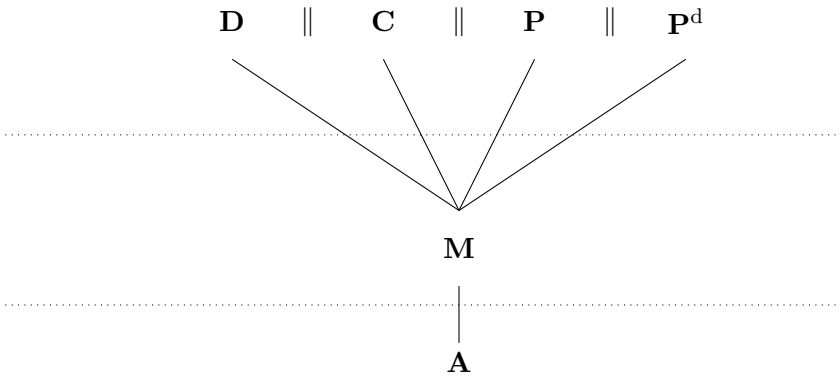


FIGURE 7.1 – Semi-treillis des NFSs ordonnés par $\preceq$.

parmi les générateurs des clones considérés (en particulier, les résultats ne dépendent pas de leur arité, ce qui peut paraître surprenant), tant que les fonctions sont pseudo-monotones. Les NFSs monotones optimaux sont, en effet, exactement ceux de la forme

$$T(\mathtt{a}) \qquad \text{ou} \qquad T(\mathtt{a}\neg),$$

où [a] est une fonction pseudo-monotone. Par exemple, représenter des fonctions Booléennes en utilisant la médiane d'arité 3 ou la médiane d'arité 5, deux générateurs du même clone des fonctions monotones auto-duales, ne présente pas de gain de complexité significatif. Finalement, les NFSs monotones optimaux sont tous équivalents deux-à-deux.

Notre contribution révèle des questions difficiles qui constituent des pistes de recherche future.

Classification complète des NFSs

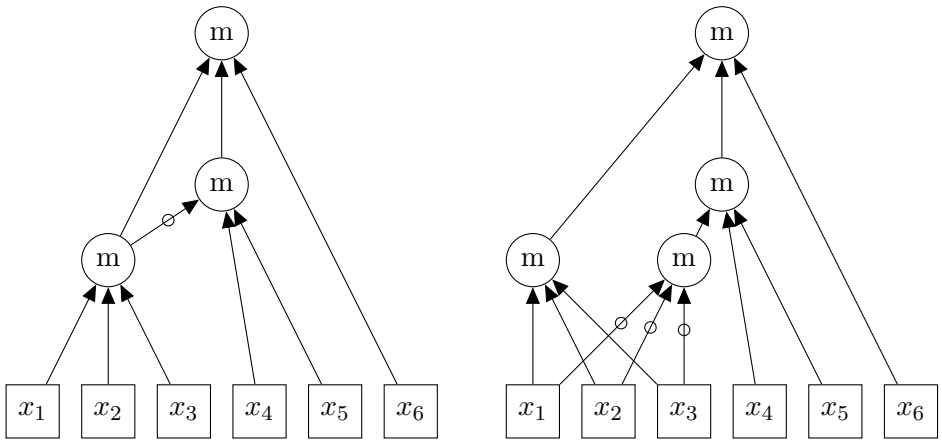
Rappelons la conjecture 153 :

« Si **A** est un NFS quasi-Sheffer non-monotone, alors $\mathbf{A} \prec \mathbf{M}$. »

Dans le but de faire lumière sur une classification complète des NFSs, cette conjecture est importante. Si effectivement nous démontrons que les NFSs quasi-Sheffer non-monotones **A** sont tels que $\mathbf{A} \prec \mathbf{M}$, le diagramme de la figure 6.5 a maintenant trois niveaux, comme indiqué dans la figure 7.1.

Une direction possible est la recherche d'une famille infinie de termes (minimaux) de **M** dont la taille croît de manière exponentielle en n, mais qui ont des représentations dans Σ dont la taille croît de manière polynomiale. Ici, une importante source de difficulté est le fait que nous n'avons pas de description explicite des termes minimaux de **M**, et qu'il est donc difficile de déterminer si un terme exponentiel peut être réécrit en un terme de taille polynomiale, sauf dans des cas extrêmes. Trouver des descriptions explicites de formes normales minimales est difficile, ce qui est sans doute due au fait que la médiane n'est pas *associative*. Au contraire, les formes normales telles que la DNF, CNF, ou la PNF sont construites à partir de la conjonction, disjonction, et somme modulo 2, qui sont associatifs. En particulier, la non-associativité de la médiane fait qu'une certaine quantité d'information sur la fonction à représenter est encodée dans la structure même de la formule médiane. Dans le cas de compositions de connecteurs associatifs tels que la conjonction, la disposition des connecteurs importe peu :

$$x \vee (y \vee z) \equiv (x \vee y) \vee z.$$



non-stratifié stratifié

FIGURE 7.2 – Circuits stratifiés ou non par rapport à la suite $\mathtt{m}, \neg$.

En revanche, la médiane n'est pas associative :

$$\mathtt{m}(\mathtt{m}(x,y,z),u,v) \not\equiv \mathtt{m}(x,\mathtt{m}(y,z,u),v),$$

car, par exemple, $\mathtt{m}(\mathtt{m}(1,1,1),0,0) \equiv 0$ et $\mathtt{m}(1,\mathtt{m}(1,1,0),0) \equiv 1$.

Systèmes redondants ou non bornés

Rappelons la première partie de la définition 100 :

> « Un système de formes normales est un ensemble non-redondant $T(\alpha_1 \cdots \alpha_n)$ de termes stratifiés tels que $[T(\alpha_1 \cdots \alpha_n)] = \Omega$ ».

Que se passe-t-il lorsque la redondance n'est pas requise ? Une conséquence immédiate est qu'alors nous pouvons considérer des systèmes à stratification non bornée, (ou alternés [CKS81]). En effet, le résultat de la proposition 108 :

> « Si $[\mathtt{a}_1], \ldots, [\mathtt{a}_n] \in \Omega \setminus \Omega(1)$ et si $T(\mathtt{a}_1 \cdots \mathtt{a}_n \neg)$ est un NFS, alors $n \leq 2$. »

dépend directement de la non-redondance des NFSs.

NFSs pour circuits

Nos définitions relatives aux NFSs s'étendent naturellement aux circuits. Nous pouvons considérer ainsi des circuits *stratifiés* par rapport à une série de connecteurs $(\alpha_1, \ldots, \alpha_n)$, c'est-à-dire des circuits tels que :

— les étiquettes des portes contiennent des symboles parmi $(\alpha_1, \ldots, \alpha_n, 0, 1)$; et
— tout sous-circuit de la forme $\alpha_j(c_1, \ldots, c_k)$, $j \in \{1, \ldots, n\}$, n'a aucun sous-circuit de la forme $\alpha_i(c'_1, \ldots, c'_\ell)$ avec $i < j$.

La figure 7.2 représente deux circuits qui sont respectivement stratifié et non-stratifié.

Un *système de formes normales pour circuits Booléens* (NFSC) peut ainsi être défini en considérant un ensemble de circuits stratifiés qui peut représenter toutes les fonctions Booléennes. La comparaison d'efficacité s'effectue en comptant le nombre de portes dans un circuits ; et

l'on obtient un ordre partiel sur les NFSCs en comparant, pour toute fonction, la taille des représentations minimales d'un NFSC à un autre, comme pour les NFSs.

Une question naturelle est de savoir si les NFSCs peuvent être classifiés de la même manière que les NFSs. La différence principale entre ces deux types de représentations est le partage de variables qui peut, dans certains cas, simplifier de manière significative les représentations.

Les Graphes Majorité-Inversion (Majority-Inverter Graphs, MIG) ont été introduits dans [AGDM14] ; ce sont des circuits construits à partir de portes Majorité (qui calculent la médiane à 3 entrées dans le cas Booléen) et de portes Inversion (qui calculent la négation). Dans [SAGDM17], les auteurs testent des méthodes de synthèse de fonctions Booléennes et d'optimisation de MIGs pour atteindre, sur des bancs d'essais de l'EPFL (École Polytechnique Fédérale de Lausanne), des représentations plus efficaces que, par exemple, celles produites avec des circuits de type AIGs (And-Inverter Graphs). Cela confirme l'hypothèse selon laquelle même dans le cadre des circuits, la médiane à $2n+1$ entrées conserve son efficacité pour la représentation des fonctions Booléennes, et ce même pour les NFSs non monotones, grâce, par exemple, au partage de sous-circuits. Dans la figure 7.2), la sous-figure de gauche, sous-titrée « non-stratifié », est plus petite que la figure de droite dans laquelle les seuls partages ont lieu au niveau des variables.

Le modèle des MIGs a également été étendu aux fonctions Majorité à $2n + 1$ branches [CHS+19]. Dans cet article, les auteurs montrent que, sur un banc d'essai de l'EPFL, leurs méthodes de synthèse de fonctions Booléennes en M_5IG (Majority-of-Five Inverter Graphs) atteignent une amélioration de 10.4% pour la taille des représentation et de 11.4% pour la profondeur par rapport à d'autres méthodes de synthèse exacte de l'état de l'art basées sur la Majorité à trois entrées décrites dans [SAGDM17].

NFSs pour opérations multi-valuées

Dans le chapitre 5, nous avons étudié la représentation de fonctions polynomiales latticielles, c'est-à-dire à valeurs dans un treillis, mais seulement en utilisant des formes médianes. Dans le chapitre 6, nous avons considéré tous les systèmes de formes normales, mais seulement dans le contexte de la représentation des fonctions Booléennes. Une piste de recherche future est l'extension de l'étude de ce chapitre aux opérations multi-valuées, c'est-à-dire définies sur un ensemble L de cardinalité au moins 3. Dans ce cas, une importante difficulté à surmonter vient du fait que l'ensemble des clones sur un ensemble fini contenant au moins 3 éléments a la cardinalité du continu, ou, en d'autres mots, n'est pas dénombrable (voir, par exemple, [IM59, BV02]), et, de plus, il n'existe pas de description complète du treillis de clones correspondant, ni de description complète des compositions de clones. Toutefois, il est possible d'étudier des parties ou des propriétés de ces treillis (voir, par exemple, [Lau06]). Certains clones ou sous-ensembles des treillis de clones sur plus de 3 éléments pourraient, comme lors de cette étude pour 2 éléments, fournir des connecteurs quasi-Sheffer qui, premièrement, engendrent toutes les opérations $L^n \to L$ (par composition et en autorisant les constantes), et, deuxièmement, engendrent des représentations efficaces. Par exemple, une description des *co-atomes* de ces treillis, c'est-à-dire des clones *maximaux* sur L, qui sont les clones couverts par $\mathbf{1}$, a été donnée par Post [Pos41] pour 2 éléments, comme nous l'avons vu [2], Jablonskii [Yab58] pour 3 éléments, Mal'cev pour 4 éléments, et par Rosenberg [Ros70] pour un nombre fini d'éléments.

2. Ce sont les clones T_1, T_2, L, M, S.

Bibliographie

[Aar05] Scott Aaronson. Guest column : NP-complete problems and physical reality. *ACM Sigact News*, 36(1) :30–52, 2005.

[AB09] Sanjeev Arora and Boaz Barak. *Computational complexity : a modern approach*. Cambridge University Press, 2009.

[ACM+14] Girolamo Arnaldi, Mario Caravale, Giacomo Martina, Antonio Menniti Ippolito, and Manlio Simonetti. *I papi. Da Pietro a Francesco*. Enciclopedia dei papi. Istituto della Enciclopedia italiana Treccani, 2014.

[Acz04] János Aczél. The associativity equation re-revisited. In *AIP Conference Proceedings*, volume 707, pages 195–203. AIP, 2004.

[AGCDM15] Luca Amarú, Pierre-Emmanuel Gaillardon, Anupam Chattopadhyay, and Giovanni De Micheli. A sound and complete axiomatization of majority-n logic. *IEEE Transactions on Computers*, 65(9) :2889–2895, 2015.

[AGDM14] Luca Amarú, Pierre-Emmanuel Gaillardon, and Giovanni De Micheli. Majority-inverter graph : a novel data-structure and algorithms for efficient logic optimization. In *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2014.

[AKGR20] Scott Aaronson, Gred Kuperberg, Christopher Granade, and Vincent Russo. The complexity zoo. https://complexityzoo.uwaterloo.ca/Complexity_Zoo, february 2020.

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Annals of mathematics*, pages 781–793, 2004.

[AM12] Alnur Ali and Marina Meilă. Experiments with Kemeny ranking : what works when ? *Mathematical Social Sciences*, 64(1) :28–40, 2012.

[Ari79] Aristote. *Métaphysique d'Aristote, traduite en français avec des notes perpétuelles par J. Barthélémy-Saint-Hilaire*. Librairie Germer-Baillières et Compagnie, 1879.

[BB84] Hans-Jürgen Bandelt and Jean-Pierre Barthelemy. Medians in median graphs. *Discrete Applied Mathematics*, 8(2) :131–142, 1984.

[BC17] Quentin Brabant and Miguel Couceiro. k-maxitive Sugeno integrals as aggregation models for ordinal preferences. *Fuzzy Sets and Systems*, 343, 06 2017.

[BH83] Hans-Jürgen Bandelt and Jarmila Hedlíková. Median algebras. *Discrete mathematics*, 45(1) :1–30, 1983.

[Bir40] Garrett Birkhoff. *Lattice theory*, volume 25. American Mathematical Soc., 1940.

[BK47] Garrett Birkhoff and Stephen A. Kiss. A ternary operation in distributive lattices. *Bulletin of the American Mathematical Society*, 53(8) :749–752, 1947.

[BM70] Roger Joseph Boscovich and Christopher Maire. *Voyage astronomique et géographique dans l'État de l'Eglise, entrepris par l'ordre et sous les auspices du pape Benoît XIV, pour mesurer deux degrés du méridien et corriger la carte dans l'État ecclésiastique, par les PP. Maire et Boscovich, de la Conpagnie de Jesus, traduit du latin, augmenté de Notes & d'extraits de nouvelles mesures de dégrés faites en Italie, en Allemagne, en Hongrie & en Amérique, Avec une nouvelle Carte des États du Pape levée géométriquement.* N.-M. Tilliard, 1770.

[BM81] Jean Pierre Barthelemy and Bernard Monjardet. The median procedure in cluster analysis and social choice theory. *Mathematical social sciences*, 1(3) :235–267, 1981.

[BM92] Hans-Jürgen Bandelt and Gerasimos C. Meletiou. The algebra of majority consensus. *algebra universalis*, 29(4) :546–555, 1992.

[BN99] Franz Baader and Tobias Nipkow. *Term rewriting and all that.* Cambridge university press, 1999.

[Boo47] George Boole. *The mathematical analysis of logic.* Philosophical Library, 1847.

[BTT89] John Bartholdi, Craig A. Tovey, and Michael A. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and welfare*, 6(2) :157–165, 1989.

[BV02] Elmar Böhler and Heribert Vollmer. Boolean functions and Post's lattice with applications to complexity theory. *Lecture Notes for Logic and Interaction*, 2002.

[CAS+16] Anupam Chattopadhyay, Luca Amarú, Mathias Soeken, Pierre-Emmanuel Gaillardon, and Giovanni De Micheli. Notes on majority Boolean algebra. In *2016 IEEE 46th International Symposium on Multiple-Valued Logic (ISMVL)*, pages 50–55. IEEE, 2016.

[CFL06] Miguel Couceiro, Stephan Foldes, and Erkko Lehtonen. Composition of Post classes and normal forms of Boolean functions. *Discrete mathematics*, 306(24) :3223–3243, 2006.

[CH11] Yves Crama and Peter L. Hammer. *Boolean functions : Theory, algorithms, and applications.* Cambridge University Press, 2011.

[CHS+19] Zhufei Chu, Winston Haaswijk, Mathias Soeken, Yinshui Xia, Lunyao Wang, and Giovanni De Micheli. Exact synthesis of boolean functions in majority-of-five forms. In *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE, 2019.

[Chu36] Alonzo Church. A note on the Entscheidungsproblem. *The journal of symbolic logic*, 1(1) :40–41, 1936.

[CJW06] James A. Carlson, Arthur Jaffe, and Andrew Wiles. *The millennium prize problems.* American Mathematical Society Providence, RI, 2006.

[CKS81] Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. Alternation. *J. ACM*, 28(1) :114–133, January 1981.

[CL61] Louis Couturat and Gottfried Wilhelm Leibniz. Opuscules et fragments inédits de Leibniz, paris, 1903. *Reprint Hildesheim*, 1961.

[CL12] Miguel Couceiro and Erkko Lehtonen. Galois theory for sets of operations closed under permutation, cylindrification, and composition. *Algebra universalis*, 67(3) :273–297, 2012.

[CLM+18] Miguel Couceiro, Erkko Lehtonen, Pierre Mercuriali, Romain Péchoux, and Mathias Soeken. Normal form systems generated by single connectives have mutually equivalent efficiency. In *DICE 2018 - Developments in Implicit Computational Complexity*, Thessaloniki, Greece, April 2018.

[CLMP20] Miguel Couceiro, Erkko Lehtonen, Pierre Mercuriali, and Romain Péchoux. On the efficiency of normal form systems for representing Boolean functions. *Theor. Comput. Sci.*, 813 :341–361, 2020.

[CLMW11] Miguel Couceiro, Erkko Lehtonen, Jean-Luc Marichal, and Tamás Waldhauser. An algorithm for producing median formulas for Boolean functions. In *Proc. of the Reed Muller 2011 Workshop*, pages 49–54, 2011.

[CLW12] Miguel Couceiro, Erkko Lehtonen, and Tamás Waldhauser. Decompositions of functions based on arity gap. *Discrete Mathematics*, 312(2) :238–247, 2012.

[CLW15] Miguel Couceiro, Erkko Lehtonen, and Tamás Waldhauser. A survey on the arity gap. *Multiple-Valued Logic and Soft Computing*, 24(1-4) :223–249, 2015.

[CM10] Miguel Couceiro and Jean-Luc Marichal. Characterizations of discrete Sugeno integrals as polynomial functions over distributive lattices. *Fuzzy Sets and Systems*, 161(5) :694 – 707, 2010. Theme : Non-Additive Measures Algebra.

[CM12] Miguel Couceiro and Jean-Luc Marichal. Aczélian n-ary semigroups. In *Semigroup Forum*, volume 85 : 1, pages 81–90. Springer, 2012.

[CMP17] Miguel Couceiro, Pierre Mercuriali, and Romain Péchoux. Sur l'efficacité des systèmes de formes normales de fonctions Booléennes. In *LFA 2017 - 26èmes Rencontres Francophones sur la Logique Floue et ses Applications*, pages 1–8, Amiens, France, October 2017.

[CMPS17] Miguel Couceiro, Pierre Mercuriali, Romain Péchoux, and Abdallah Saffidine. Median based calculus for lattice polynomials and monotone Boolean functions. In *47th IEEE International Symposium on Multiple-Valued Logic, ISMVL 2017, Novi Sad, Serbia, May 22-24, 2017*, pages 37–42. IEEE Computer Society, 2017.

[CMPS19] Miguel Couceiro, Pierre Mercuriali, Romain Péchoux, and Abdallah Saffidine. On the complexity of minimizing median normal forms of monotone Boolean functions and lattice polynomials. *J. Multiple Valued Log. Soft Comput.*, 33(3) :197–218, 2019.

[Coh81] Paul Moritz Cohn. *Universal algebra*, volume 159. Reidel Dordrecht, 1981.

[Coo71] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158, 1971.

[Cop19] B. Jack Copeland. The Church-Turing thesis. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, spring 2019 edition, 2019.

[Cou43] Antoine Augustin Cournot. *Exposition de la théorie des chances et des probabilités*. L. Hachette, 1843.

[CR36] Alonzo Church and J. Barkley Rosser. Some properties of conversion. *Transactions of the American Mathematical Society*, 39(3) :472–482, 1936.

[Csá05] Béla Csákány. Minimal clones—a minicourse. *Algebra Universalis*, 54(1) :73–90, 2005.

[CT90] Ramaswamy Chandrasekaran and Arie Tamir. Algebraic optimization : the Fermat-Weber location problem. *Mathematical Programming*, 46(1-3) :219–224, 1990.

[Dav58] Martin Davis. *Computability and unsolvability*. McGraw-Hill, 1958.

[DF91] Pierre De Fermat. *Oeuvres de Fermat*, volume 1. Gauthier-Villars et fils, 1891.

[Die63] Jean Alexandre Dieudonné. *Fondements de l'analyse moderne*. Gauthier-Villars, 1963.

[DL74] Pierre S. De Laplace. Mémoire sur la probabilité des causes par les événements. *Mémoire de mathématique et physique présentés à l'Acad'emie royale des sciences*, 6 :621–656, 1774.

[DL89] Pierre S. De Laplace. Sur les degrés mesurés des méridiens, et sur les longueurs observées sur pendule. *Histoire de L'Academic Royale des Inscriptions et Belles Lettres, avec les Mémoires de Littérature Tirez des Registres de Cette Académie*, 1789.

[DM47] Augustus De Morgan. *Formal logic : or, the calculus of inference, necessary and probable*. Taylor and Walton, 1847.

[DP92] Brian A. Davey and Hilary A. Priestley. *Introduction to lattices and order*. Cambridge University Press, 1992.

[Edm65] Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of mathematics*, 17 :449–467, 1965.

[Eid09] Jean-Denis Eiden. *Géométrie analytique classique*. Calvage & Mounet Paris, 2009.

[Eul22] Leonhard Euler. *Commentationes algebraicae ad theoriam combinationum et probabilitatum pertinentes*, volume 1. Springer Science & Business Media, 1922. Recueil de travaux Eulériens.

[F$^+$62] Eric Foxley et al. The determination of all Sheffer functions in 3-valued logic, using a logical computer. *Notre Dame journal of formal logic*, 3(1) :41–50, 1962.

[FH03] Lance Fortnow and Steve Homer. A short history of computational complexity. *Bulletin of the EATCS*, 80 :95–133, 2003.

[FP04] Stephan Foldes and Grant R. Pogosyan. Post classes characterized by functional terms. *Discrete Applied Mathematics*, 142(1-3) :35–51, 2004.

[Gal83] Francis Galton. *Inquiries into human faculty and its development*. Macmillan, 1883.

[Gas19] William I. Gasarch. Guest column : The third P= ? NP poll. *ACM SIGACT News*, 50(1) :38–59, 2019.

[GHM08] Judy Goldsmith, Matthias Hagen, and Martin Mundhenk. Complexity of DNF minimization and isomorphism testing for monotone formulas. *Information and Computation*, 206 :760–775, 06 2008.

[Gil77] John Gill. Computational complexity of probabilistic Turing machines. *SIAM Journal on Computing*, 6(4) :675–695, 1977.

[GJS74] Michael R. Garey, David S. Johnson, and Larry Stockmeyer. Some simplified NP-complete problems. In *Proceedings of the sixth annual ACM symposium on Theory of computing*, pages 47–63, 1974.

[GL81] Peter Gács and Laszlo Lovász. Khachiyan's algorithm for linear programming. In *Mathematical Programming at Oberwolfach*, pages 61–68. Springer, 1981.

[Glo92] Zygmunt Gloger. *Pieśni ludu (Chants du peuple)*. Skład gł. Gebethner i Wolff, 1892.

[GMMP11] Michel Grabisch, Jean-Luc Marichal, Radko Mesiar, and Endre Pap. Aggregation functions : means. *Information Sciences*, 181(1) :1–22, 2011.

[GMN+60] Eiichi Goto, Kenro Murata, Kisaburo Nakazawa, Keisuke Nakagawa, Tohru Moto-Oka, Y Matsuoka, Yoshihiro Ishibashi, Haruhisa Ishida, Takashi Soma, and Eiiti Wada. Esaki diode high-speed logical circuits. *IRE Transactions on electronic computers*, 1 :25–29, 1960.

[Gé27] I. I. (Жегалкин, И. И.) Gégalkine. О технике вычислений предложений в символической логике. (Sur le calcul des propositions dans la logique symbolique). Математический сборник, 34(1) :9–28, 1927.

[HA28] D. Hilbert and W. Ackerman. *Theoretische Logik*. Springer, 1928.

[Har79] Harman Leon Harter. *A chronological annotated bibliography on order statistics*, volume 1. Air Force Flight Dynamics Laboratory, Air Force Wright Aeronautical Laboratories, Air Force Systems Command, United States Air Force, 1979.

[Has48] Helmut Hasse. Existenz und mannigfaltigkeit abelscher algebren mit vorgegebener galoisgruppe über einem teilkörper des grundkörpers. i. *Mathematische Nachrichten*, 1(1) :40–61, 1948.

[Has53] Helmut Hasse. *Über die Klassenzahl abelscher Zahlkörper*. Springer-Verlag Berlin Heidelberg, 1953.

[HGC79] B. Haile, I.W. Goossen, and Black Mustache Circle. *Waterway : A Navajo Ceremonial Myth*. American tribal religions. Museum of Northern Arizona Press, 1979.

[HMU01] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. Introduction to automata theory, languages, and computation. *Acm Sigact News*, 32(1) :60–65, 2001.

[Hoe16] Carl Hoefer. Causal determinism. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, spring 2016 edition, 2016.

[IM59] Iu I. Ianov and A. A. Muchnik. О существовании замкнутых классов к-значной логики без конечного базиса (the existence of k-valued closed classes having no finite basis). Доклады Академии Наук СССР *(Proceedings of the USSR Academy of Sciences)*, 127(1) :44–46, 1959.

[Ins20] Clay Math Institute. Millenium prize problems - P versus NP. http://www.claymath.org/millennium-problems/p-vs-np-problem, January 2020.

[Ish20] Richard Ishida. Text size in translation. https://www.w3.org/International/articles/article-text-size.en, January 2020.

[Juk12] Stasys Jukna. *The Mystery of Negations*, pages 285–300. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[Kar72] Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[KB70] Donald E. Knuth and Peter B. Bendix. Simple word problems in universal algebras. computational problems in abstract algebra. *Leech, J.(ed.)*, pages 263–297, 1970.

[Kha79] Leonid Genrikhovich Khachiyan. Полиномиальные алгоритмы в линейном программировании (a polynomial algorithm in linear programming). In Доклады Академии Наук *(Proceedings of the Academy of Sciences)*, volume 244, pages 1093–1096. Russian Academy of Sciences, 1979.

[Klo92] Jan Willem Klop. *Term rewriting systems*, volume 2. Oxford University Press, 1992.

[KPS14] Sebastian Kerkhoff, Reinhard Pöschel, and Friedrich Martin Schneider. A short introduction to clones. *Electronic Notes Theoretical Computer Science*, 303 :107–120, March 2014.

[Lau06] Dietlinde Lau. *Function algebras on finite sets : Basic course on many-valued logic and clone theory*. Springer, 2006.

[LB04] TOB La Bible. Épître à Tite ; traduction œcuménique de la bible. *Paris-Villiers-le-Bel, Cerf-Bibli'O*, 2004.

[Lev73] Leonid Anatolevich Levin. Универсальные задачи перебора (universal search problems). Проблемы передачи информации *(Problems of Information Transmission)*, 9(3) :115–116, 1973.

[Lip10] Richard J. Lipton. *The* P = NP *Question and Gödel's Lost Letter*. Springer Science & Business Media, 2010.

[LLTdBN11] Denis Lanier, Jean Lejeune, Didier Trotoux, and IREM de Basse-Normandie. L'invention de la médiane. *Circulation, transmission, héritage*, pages 387–414, 2011.

[Mar52] Norman Marshall Martin. *Sheffer functions and axiom sets in m-valued propositional logic*. PhD thesis, University of California, Los Angeles, 1952.

[Mar00] Jean-Luc Marichal. On choquet and sugeno integrals as aggregation functions. *Fuzzy Measures and Integrals-Theory and Applications*, pages 247–272, 2000.

[Mar09a] Jean-Luc Marichal. Aggregation functions for decision making. *arXiv preprint arXiv :0901.4232*, 2009.

[Mar09b] Jean-Luc Marichal. Weighted lattice polynomials. *Discrete Mathematics*, 309(4) :814–820, 2009.

[May50] Tobias Mayer. Abhandlung über die umwälzung des monds um seine axe, und die scheinbare bewegung der mondsflecken. *Kosmographische Nachrichten und Sammlungen auf das Jahr*, 1748 :41–51, 1750.

[Men36] Karl Menger. New foundations of projective and affine geometry. *Annals of Mathematics*, pages 456–482, 1936.

[Mon12] Bernard Monjardet. Marc barbut au pays des médianes. *Mathématiques et sciences humaines. Mathematics and social sciences*, 200 :p–67, 2012.

[Mou37] Émile Moussat. *Être chic ! De la morale du sport à une morale sportive*. Albert Messein Editeur, 1937.

[Mul54] David E. Muller. Application of Boolean algebra to switching circuit design and to error detection. *Transactions of the IRE professional group on electronic computers*, 3 :6–12, 1954.

[MW62] H.S. Miller and Richard O. Winder. Majority-logic synthesis by geometric methods. *IRE Transactions on Electronic Computers*, pages 89–90, 1962.

[Pap03] Christos H. Papadimitriou. *Computational complexity*. Addison-Wesley, 2003.

[Pea89] Giuseppe Peano. *Arithmetices principia : Nova methodo exposita.* Fratres Bocca, 1889.

[Per14] Sylvain Perifel. *Complexité algorithmique.* Ellipses, 2014.

[Pla27] Platon. *Oeuvres complètes de Platon, traduites du grec en français, accompagnées de notes, et précédées d'une introduction sur la philosophie de Platon, par Victor Cousin, Ex-maître de conférences à l'ancienne école normale, professeur suppléant de l'histoire de la philosophie moderne, à la faculté des lettres de l'académie de Paris.*, volume 4. Paris : Bossange frères, libraires, quai Voltaire, n. 11. In-8, 1827.

[Pla93] David A Plaisted. Equational reasoning and term rewriting systems. *Handbook of logic in artificial intelligence and logic programming*, 1 :273–364, 1993.

[Pos40] Emil L. Post. Polyadic groups. *Transactions of the American Mathematical Society*, 48(2) :208–350, 1940.

[Pos41] Emil L. Post. *The Two-Valued Iterative Systems of Mathematical Logic.(AM-5)*, volume 5. Princeton University Press, 1941.

[Pra74] Vaughan R Pratt. The power of negative thinking in multiplying boolean matrices. In *Proceedings of the sixth annual ACM symposium on Theory of computing*, pages 80–83, 1974.

[Raz85a] A. A. Razborov. Lower bounds on monotone complexity of the logical permanent. *Mathematical notes of the Academy of Sciences of the USSR*, 37(6) :485–493, Jun 1985.

[Raz85b] A. A. Razborov. Нижние оценки монотонной сложности некоторых булевых функций (lower bounds on the monotone complexity of some Boolean functions). Доклады Академии Наук СССР *(Proceedings of the USSR Academy of Sciences)*, 281(4) :798–801, 1985.

[Ree53] Irving S. Reed. A class of multiple-error-correcting codes and the decoding scheme. Technical report, Massachusett's Institute of Tech. Lexington Lincoln Lab., 1953.

[Rom61] Jules Romains. *Pour raison garder.* Flammarion, 1961.

[Ros70] I.G. Rosenberg. Uber die funktionale vollstandigkeit in den mehrwertigen logiken (struktur der funktionen von mehreren veranderlichen auf endlichen mengen). *Akademie Věd. Ser. Math. Nat. Sci*, 80 :4, 1970.

[Rus06] Bertrand Russell. Les paradoxes de la logique. *Revue de métaphysique et de morale*, 14(5) :627–650, 1906.

[Rus08] Bertrand Russell. Mathematical logic as based on the theory of types. *American journal of mathematics*, 30(3) :222–262, 1908.

[Ruz81] Walter L. Ruzzo. On uniform circuit complexity. *Journal of computer and system sciences*, 22 :365–383, 1981.

[SAGDM17] Mathias Soeken, Luca Gaetano Amaru, Pierre-Emmanuel Gaillardon, and Giovanni De Micheli. Exact synthesis of majority-inverter graphs and its applications. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 36(11) :1842–1855, 2017.

[Sal63] A. Salomaa. On essential variables of functions, especially in the algebra of logic. *Annales Academiæ Scientiarum Fennicæ*, 1963.

[San69] Eugene S. Santos. Probabilistic turing machines and computability. *Proceedings of the American Mathematical Society*, 22(3) :704–710, 1969.

[Sav98] John E. Savage. *Models of computation*, volume 136. Addison-Wesley Reading, MA, 1998.

[SB81] Hanamantagouda P. Sankappanavar and Stanley Burris. A course in universal algebra. *Graduate Texts Math*, 78, 1981.

[Sha49] Claude E. Shannon. The synthesis of two-terminal switching circuits. *The Bell System Technical Journal*, 28(1) :59–98, 1949.

[Sho52] Marlow Sholander. Trees, lattices, order, and betweenness. *Proceedings of the American Mathematical Society*, 3(3) :369–381, 1952.

[Sho54a] Marlow Sholander. Medians and betweenness. *Proceedings of the American Mathematical Society*, 5(5) :801–807, 1954.

[Sho54b] Marlow Sholander. Medians, lattices, and trees. *Proceedings of the American Mathematical Society*, 5(5) :808–812, 1954.

[SM73] Larry J. Stockmeyer and Albert R. Meyer. Word problems requiring exponential time (preliminary report). In *Proceedings of the fifth annual ACM symposium on Theory of computing*, pages 1–9, 1973.

[Sto76] Larry J. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1) :1–22, 1976.

[Sto88] Ivan Stojmenovic. On Sheffer symmetric logic. *Discrete applied mathematics*, 22(89) :267–2, 1988.

[SU02] Marcus Schaefer and Christopher Umans. Completeness in the polynomial-time hierarchy : A compendium. *SIGACT news*, 33(3) :32–49, 2002.

[Sug74] Michio Sugeno. Theory of fuzzy integrals and its applications. *Doct. Thesis, Tokyo Institute of technology*, 1974.

[TL14] Evangelista Torricelli and Gino Loria. *Opere di Evangelista Torricelli*, volume 1. Stab, tipo-Lit. G. Montanavi, 1914.

[Tur36] Alan Mathison Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London mathematical society*, 42(1) :230–265, 1936.

[Tur38] Alan Mathison Turing. On computable numbers, with an application to the Entscheidungsproblem. a correction. *Proceedings of the London Mathematical Society*, 2(1) :544–546, 1938.

[Uma01] Christopher Umans. The minimum equivalent DNF problem and shortest implicants. *Journal of Computer and System Sciences*, 63(4) :597–611, 2001.

[Val79] Leslie G. Valiant. The complexity of computing the permanent. *Theoretical computer science*, 8(2) :189–201, 1979.

[VM06] R. Veroff and W. McCune. First-order proof of a median algebra problem. http://www.cs.unm.edu/~veroff/MEDIAN_ALGEBRA/, September 2006.

[Vog95] Henri Vogt. *Leçons sur la résolution algébrique des équations*. Nony, 1895.

[VZ00] Yehuda Vardi and Cun-Hui Zhang. The multivariate l_1-median and associated data depth. *Proceedings of the National Academy of Sciences*, 97(4) :1423–1426, 2000.

[Wer42] William Wernick. Complete sets of logical functions. *Transactions of the American Mathematical Society*, 51(1) :117–132, 1942.

[Wil96] Ross Willard. Essential arities of term operations in finite algebras. *Discrete Mathematics*, 149(1-3) :239–259, 1996.

[Wra76] Celia Wrathall. Complete sets and the polynomial-time hierarchy. *Theoretical Computer Science*, 3(1) :23–33, 1976.

[Wym83] Leland C. Wyman. Navajo ceremonial system. *Handbook of North American Indians*, 10(536-557), 1983.

[Yab58] Sergei Vsevolodovich Yablonskii. Функциональные построения в k-значной логике (functional constructions in a k-valued logic). Сборник статей по математической логике и ее приложениям к некоторым вопросам кибернетики, 51 :5–142, 1958.

[You98] Robert Young. *Young's literal translation of the Holy Bible*. Baker Book House, 1898.

[Zel68] Bohdan Zelinka. Medians and peripherians of trees. *Archivum Mathematicum*, 4(2) :87–95, 1968.

Table des figures

Index

Résumé

Les fonctions peuvent être représentées de différentes manières, qui ne sont pas toutes aussi efficaces. Les tables de vérités qui décrivent explicitement le comportement d'une fonction ont une taille exponentielle en le nombre d'arguments de cette fonction. En revanche, des représentations syntaxiques comme les circuits ou les termes peuvent être beaucoup plus petites. Pourtant, même entre deux représentations syntaxiques, il peut exister des différences exponentielles : considérons, par exemple, une conjonction de variables différentes qui, une fois convertie en DNF, a une taille exponentielle en le nombre de variables. Cette différence est le résultat d'une forte contrainte structurelle entre les représentations en CNF et en DNF. Les languages que nous étudions dans cette thèse sont les *formes normales*, c'est-à-dire, des expressions formelles qui vérifient des contraintes structurelles similaires, étant donnés des ensembles de connecteurs ordonnés. Cela soulève plusieurs questions qui sont au coeur de cette thèse.

1. Peut-on définir proprement une notion d'efficacité des formes normales, et peut-on comparer proprement l'efficacité de deux formes normales ?
2. Peut-on considérer certaines formes comme étant *optimales*, c'est-à-dire plus efficaces que toute autre forme normale ?
3. Si oui, comment obtenir et optimiser des expressions en forme normale de manière algorithmique, et quelle est la complexité de ces procédures algorithmiques ?

Mesurer la complexité de fonctions multi-valuées, et en particulier des fonctions Booléennes, est un sujet central en théorie de la complexité calculatoire et de la complexité des circuits. Compter le nombre de connecteurs, ou de portes dans le cas des circuits, dans la représentation la plus petite d'une fonction particulière est une manière naturelle de mesurer la complexité. Nous pouvons par la suite chercher et obtenir des minorants, et les langages peuvent être comparés en quantifiant sur toutes les fonctions d'une classe particulière. Il a été démontré que la forme normale médiane pour les fonctions Booléennes, ou MNF, c'est-à-dire un ensemble de termes construits en utilisant la médiane, des constantes, des variables, et des variables niées, est polynomialement plus efficace que la DNF, la CNF, la forme normale polynomiale et son dual, et au moins aussi efficace polynomialement que toute autre forme normale dite « non-redondante ». Des résultats expérimentaux récents sur la minimalisation et la synthèse de circuits confirment l'optimalité de la médiane $2n + 1$-aire pour representer certaines classes de fonctions.

Grâce à la théorie des clones, nous pouvons décrire de manière exhaustive les NFSs pour les fonctions Booléennes. Nous décrivons les NFSs monotones, qui sont minimales suivant un pré-ordre sur la taille des représentations minimales. Nous montrons que les NFSs monotones optimaux sont exactement ceux qui n'utilisent qu'un seul connecteur ou un connecteur et la négation, telle que la MNF (médiane et négation). Puis, nous montrons que l'optimalité ne dépend pas de l'arité des connecteurs utilisés : en particulier, utiliser la médiane ou la médiane à 5 branches ne change pas la complexité des représentations en MNF. Nous explorons également des problèmes de complexité liés et montrons que le problème de décision qui consiste à déterminer si une formule est en MNF, c'est-à-dire, le problème de minimalisation de la forme normale médiane d'une fonction Booléenne monotone, est dans Σ_2^{P}, au second niveau de la hiérarchie polynomiale. Nous montrons également ce résultat pour les fonctions Booléennes arbitraires.

Mots-clés: complexité, forme normale, médiane

Abstract

Functions can be represented in many different ways, which are not equally efficient. Truth tables that explicitly describe the behavior of a function have a size exponential in the number of function arguments. However, syntactic representations such as circuits and terms can be much smaller. Nevertheless, even between syntactic representations, exponential differences still arise : consider, for instance, a conjunction of different variables which, once converted into a DNF, is of size exponential in the number of variables. This difference is the result of the strong structural constraint between CNF and DNF representations.The languages we study in this thesis are the so-called *normal forms*, i.e., formal expressions that obey similar structural constraints, given ordered sets of connectives. This raises several questions which are the main subject matter of this thesis :

1. Can the notion of efficiency of normal forms be properly defined, and the efficiency of two normal forms properly compared ?
2. Can some normal forms be deemed *optimal*, that is, more efficient than any other normal form ?
3. If so, how to algorithmically obtain and optimize normal form expressions, and how complex are these algorithmic procedures ?

Measuring the complexity of multivariate functions, and in particular of Boolean functions, is a central topic in computational complexity theory and circuit complexity theory. Counting the number of connectives, or gates in the case of circuits, in the smallest representation of a particular function is a natural way to measure complexity. Lower bounds can then be searched for and obtained, and languages can be compared by quantifying over all functions of a certain class. It has been shown that the median normal form for Boolean functions, or MNF for short, that is a set of terms built using only the median, constants, variables, and negated variables, is polynomially more efficient than the DNF, CNF, Polynomial Normal Form, and its dual, and at least polynomially as efficient as any other "irredundant" normal form. Recent experimental results on minimization and logical synthesis confirm this optimality of $2n + 1$-ary median to represent certain classes of functions.

Using clone theory, we are able to describe exhaustively NFSs for Boolean functions. We describe optimal monotonic NFSs, that are minimal with respect to a preorder on the size of minimal representations. Furthermore, we show that optimal monotonic NFSs are exactly those that use a single connective or one connective and the negation, such as the MNF (median and negation). Finally, we show that optimality does not depend on the arity of the connective : in particular, using the ternary or the 5-ary median does not change the complexity of the MNF representations. We investigate as well related computational complexity issues and show that the problem of deciding if a formula is in MNF, that is, minimizing the median form of a monotone Boolean function, is in Σ_2^{P}, at the second level of the polynomial hierarchy ; we also show this result for arbitrary Boolean functions.

Keywords: complexity, normal form, median