N° d'ordre : 6797

UNIVERSITÉ PARIS-SUD
UFR SCIENTIFIQUE D'ORSAY

THÈSE

Présentée par

BENJAMIN MONATE

Pour obtenir le grade de

DOCTEUR EN SCIENCES DE L'UNIVERSITÉ PARIS XI

Spécialité : INFORMATIQUE

Sujet :

PROPRIÉTÉS UNIFORMES DE FAMILLES DE SYSTÈMES DE RÉÉCRITURE DE MOTS PARAMÉTRÉES PAR DES ENTIERS

Soutenue le 7 janvier 2002 devant la commission d'examen composée de

M. HUBERT COMON	rapporteur
Mme EVELYNE CONTEJEAN	
M. BERNHARD GRAMLICH	rapporteur
M. JEAN-PIERRE JOUANNAUD	
Mme BRIGITTE ROZOY	présidente
M. CLAUDE KIRCHNER	

À ma mère et à mon père

Remerciements

Mes premiers remerciements vont à ma directrice de thèse, Evelyne Contejean dont la disponibilité, le soutien, la rigueur scientifique et les gâteaux m'ont été précieux pendant ces années. Un grand merci à mon autre directeur de thèse, Jean-Pierre Jouannaud, qui a accepté de me laisser une grande liberté tout en continuant à orienter mon travail. Merci à Claude Marché qui, tout au long de cette thèse, m'a offert un regard critique toujours constructif. Merci à Hubert Comon et Bernhard Gramlich d'avoir accepté d'être les rapporteurs de cette thèse et de leur lecture attentive ainsi que de leurs nombreuses suggestions qui ont largement contribué à la clarté de ce manuscrit. Merci enfin à Claude Kirchner d'avoir accepté de participer à mon jury ainsi qu'à Brigitte Rozoy de l'avoir présidé.

Ce travail n'aurait pas le même visage si je n'avais pas été accueilli avec tant de convivialité par l'équipe Démons du Laboratoire de Recherche en Informatique. Je remercie tout particulièrement Alexandre Boudet, Ralf Treinen et Christine Paulin dont les conseils scientifiques me furent précieux. Merci à Laurence Puel qui m'a donné l'envie de choisir l'informatique théorique. Merci aussi à ceux qui contribuèrent à l'ambiance agréable : Jean-Christophe Filliâtre, Judicaël Courant, Pierre Courtieu, Bruno Guillaume, Daria Walukiewicz, Jacek Chrzaszcz, Delia Kesner et Julien Forest. Pardon à Alice Marché de lui avoir emprunté ses parents. Et comment ne pas citer Xavier Urbain, dont la présence fut toujours « intense ». Ces années resteront de merveilleux souvenirs.

Merci à ma maman qui m'a toujours soutenu et qui a tout fait pour que je puisse un jour écrire cette page. Merci à mes grand-parents et à toute ma famille pour leur soutien indéfectible. Merci à mes amis qui ont fait tous leurs efforts pour comprendre mon travail. Merci à Elwood et à Spooky de leur solidarité et de leur naïve incompréhension.

Merci enfin à Diane de sa confiance pendant toutes ces années et pour la vie que nous continuons de construire.

Table des matières

Première partie

Entrée en matière

Introduction

Les équations sont parmi les objets mathématiques les plus répandus dans les sciences modernes. Elles apparaissent comme des outils favorisant une grande rigueur tant dans les raisonnements que dans les modélisations de phénomènes. Les équations dont nous allons parler sont celles qui permettent de décrire des structures algébriques mathématiques. Prenons un exemple issu de la géométrie élémentaire du plan muni d'un point appelé origine. Notons r la rotation d'angle $\pi/2$ autour de l'origine, c'est-à-dire d'un quart de tour. Notons aussi s la symétrie par rapport à l'axe des abscisses. Nous savons que si l'on applique 4 fois la rotation r à un point, il revient à la même place. Si l'on note ϵ la transformation du plan qui laisse immobiles tous les points du plan, ce comportement peut se représenter avec l'équation $r^4 = \epsilon$. De même deux applications successives de la symétrie s laissent tous les points immobiles donc $s^2 = \epsilon$. Un raisonnement élémentaire montre aussi que $(sr)^2 = \epsilon$. Maintenant nous pouvons nous poser quelques questions concernant s et r. Par exemple est-ce que faire une rotation, une symétrie puis encore une rotation est la même chose que faire une symétrie ? Cette question peut se formuler : "Est-ce que $rsr = s$?" Cette question est une instance d'un problème général qui est appelé "problème du mot". Nous pouvons faire un raisonnement géométrique pour répondre à cette question particulière, mais cela demande de réfléchir à la signification géométrique de la question. C'est un travail qui ne peut pas être fait mécaniquement sans formaliser toute la théorie de la géométrie planaire. Par contre, nous pouvons appliquer les équations déjà prouvées pour montrer ce nouveau résultat. Ainsi en appliquant l'équation $s^2 = \epsilon$, nous avons $rsr = \epsilon rsr = s^2rsr$. En appliquant l'équation $(sr)^2 = \epsilon$, nous avons $s = s\epsilon = s(sr)^2$. En développant les puissances, nous avons bien montré que $rsr = s$. Cette technique de preuve est appelée "remplacement des égaux par les égaux" et fut consacrée par Leibnitz. En fait les trois équations que nous avons données suffisent pour démontrer *toutes* les équations vraies que l'on peut essayer de vérifier. Un tel ensemble d'équations sera appelé *présentation équationelle* de la structure formée par notre symétrie et notre rotation. Par contre il n'est pas possible en général de prouver par remplacement d'égaux qu'une équation est fausse puisque les preuves peuvent nécessiter a priori un nombre infini d'étapes de remplacement. De plus la recherche de preuve mécaniquement peut être très longue puisqu'à chaque étape, nous pouvons revenir à l'étape précédente : la recherche de preuve est hautement *non déterministe*. Partant de ce constat, la *réécriture* a été introduite : au lieu d'écrire des équations applicables dans les deux sens, nous écrivons des *règles* qui se comportent comme des équations mais ne s'appliquent que dans un seul sens. Dans notre exemple, nous pouvons choisir d'orienter nos trois équations ainsi : $s^2 \to \epsilon; r^4 \to \epsilon; (sr)^2 \to \epsilon$. Avec ce sytème, nous pouvons montrer que $rssrrr = \epsilon$ en appliquant la première règle sur $rssrrr$ nous obtenons $rrrr$, sur lequel nous appliquer la seconde règle et obtenir ϵ. La preuve est plus facile puisque nous n'avons pas eu à choisir une orientation à chaque étape. Reprenons l'équation $rsr = s$: cette fois nous ne pouvons pas appliquer de règle pour la démontrer, ce qui signifie que des théorèmes qui étaient prouvables par application des équations ne le sont plus en appliquant les règles ! En fait pour conserver les mêmes théorèmes qu'avec les équations non orientées, il faut ajouter d'autres règles. Nous verrons plus loin que $rsr \to s$ est une des règles possible à ajouter. Notons que le fait d'utiliser des règles s'il limite le non determinisme de la recherche des preuves par remplacement, grâce à l'orientation, ne le supprime pas complétement. Ainsi, dans le mot s^2r^4, les deux premières règles peuvent s'appliquer. Nous chercherons donc à construire des systèmes tels que les choix éventuels lors des étapes de remplacement ne changent pas le résultat. Un système ayant cette propriété sera dit *confluent*. Si en partant d'un mot quelconque on ne peut faire qu'un nombre fini d'étapes, nous dirons que le système *termine*. Les systèmes qui terminent et confluent sont fondamentaux puisqu'ils permettent de résoudre le problème du mot en général : il suffit d'appliquer les règles de chaque côté de l'équation à prouver. Nous sommes sûr que le processus se terminera en un nombre fini d'étapes. Nous obtenons une équation sur laquelle aucune règle ne peut s'appliquer. Soit les deux côtés sont syntaxiquement égaux et notre équation de départ est "vraie", soit ils sont différents et dans ce cas l'équation de départ était fausse.

Dans cette thèse nous allons proposer une généralisation du problème précédent. Imaginons que nous voulions décrire la structure définie par la symétrie s et une rotation d'angle $2\pi/N$ où N est un entier arbitraire. Pour chaque valeur de N nous pouvons utiliser la méthode décrite ci-dessus. Mais qu'en est-il du problème pour tout N ? Nous dirons que N est un paramètre du problème. Cette fois les équations que nous utilisons pour décrire notre structure sont les suivantes : $s^2 = \epsilon$, $r^N = \epsilon$ et $(sr)^2 = \epsilon$. Cette structure, appelée groupe dihédral d'ordre $2N$, peut être utilisée pour faire des preuves pour tout N. Nous allons mettre en place un cadre formel pour décrire ce système, puis nous donnerons des règles de remplacement des égaux par les égaux. En orientant ces équations nous définirons des règles et établirons des critères de confluence. In fine nous vérifierons que cet ensemble d'équations décrit bien le groupe dihédral pour toute valeur de N et nous donnerons des algorithmes pour résoudre le problème du mot pour tout N.

Après quelques rappels élémentaires sur les contraintes et la réécriture de mots, nous introduirons la notion de signature paramétrée, ce qui nous permettra de définir des alphabets éventuellement infinis. Dans un second temps nous présenterons un formalisme de réécriture sur ces alphabets : la réécriture paramétrée avec indices. Nous montrerons que ce formalisme permet de décrire des familles infinies de systèmes de réécriture et de prouver des propriétés de ces familles. Ensuite nous généraliserons ces notions à la réécriture paramétrée avec indices et exposants. Cela nous permettra de décrire par exemple la famille des groupes diédraux. Enfin une dernière généralisation nous permettra de formaliser la notion de réécriture paramétrée avec indices et produits finis. Pour tous ces formalismes nous donnerons des critères de confluence qui se transposeront dans les familles de systèmes traditionnels représentés. Nous donnerons aussi des méthodes pour vérifier que des systèmes équationnels sont bien des présentations de structures mathématiques finies données, en utilisant des informations concernant leur cardinal.

Chapitre 1

Contraintes sur les entiers

Les contraintes permettent de définir des ensembles de données à l'aide formules logiques. Dans ce chapitre nous allons présenter les notions élémentaires permettant de définir les contraintes sur les entiers. En général de telles contraintes ne peuvent pas se résoudre dès que le langage est assez expressif. En particulier, l'arithmétique de Peàno est indécidable, ce qui signifie intuitivement qu'il n'existe pas d'algorithme qui prend en entrée une formule quelconque et répond si elle est vraie ou non. Nous allons en nous intéresser à une classe décidable de contraintes arithmétiques : l'arithmétique de Presburger.

1.1 Arithmétique de Presburger

L'arithmétique de Presburger a été défini par Presburger en 1929 dans [29]. Cette arithmétique, sans véritable multiplication, est assez pauvre pour ne pas être indécidable.

Définition 1 *Soit X un ensemble dénombrable de variables. Les expressions de l'arithmétique de Presburger sont définies par la grammaire suivante :*

$$\begin{array}{rlll} Expr := & & c & \text{où } c \in \mathbb{Z} \\ & | & c.x & \text{où } c \in \mathbb{Z}, x \in X \\ & | & Expr + Expr & \\ & | & Expr - Expr & \end{array}$$

Les formules de l'arithmétique de Presburger sont définies par la grammaire :

$$\begin{array}{rll}
\Phi := & \top & \\
| & \bot & \\
| & \Phi \ \wedge \ \Phi & \\
| & \Phi \vee \Phi & \\
| & \neg\Phi & \\
| & \exists x \Phi & \textit{où } x \in X \\
| & \forall x \Phi & \textit{où } x \in X \\
| & Expr = Expr & \\
| & Expr < Expr & \\
| & Expr > Expr &
\end{array}$$

Le domaine de cette logique est $\mathbb{Z}$ et chaque symbole est interprété par les opérations et relations éponymes sur $\mathbb{Z}$.

Cette définition est bien sûr loin d'être minimale mais contient tout le sucre syntaxique que nous utiliseront dans la suite de ce manuscrit.

Définition 2 *Les variables d'une expressions e, notées $Var(e)$ sont définies par*

$$\begin{array}{lcl}
Var(c) & = & \emptyset \\
Var(c.x) & = & \{x\} \\
Var(e_1 + e_2) & = & Var(e_1) \cup Var(e_2) \\
Var(e_1 - e_2) & = & Var(e_1) \cup Var(e_2)
\end{array}$$

Les variables libres d'une formule sont définies par :

$$\begin{array}{lcl}
Var(\top) & = & \emptyset \\
Var(\bot) & = & \emptyset \\
Var(\phi \ \wedge \ \psi) & = & Var(\phi) \cup Var(\psi) \\
Var(\phi \vee \psi) & = & Var(\phi) \cup Var(\psi) \\
Var(\neg\phi) & = & Var(\phi) \\
Var(\exists x \phi) & = & Var(\phi) \setminus \{x\} \\
Var(\forall x \phi) & = & Var(\phi) \setminus \{x\} \\
Var(e_1 = e_2) & = & Var(e_1) \cup Var(e_2) \\
Var(e_1 < e_2) & = & Var(e_1) \cup Var(e_2) \\
Var(e_1 > e_2) & = & Var(e_1) \cup Var(e_2)
\end{array}$$

Théorème 1 *L'arithmétique de Presburger est décidable.*

Il existe de nombreuses preuves la décidabilité. Elles peuvent être purement syntaxiques, comme celle de Presburger en 1929 qui se fonde sur l'élimination des quantificateurs. Elles peuvent être plus sémantiques comme les méthodes fondées sur les automates due à Alexandre Boudet et Hubert Comon [4]. Ces dernières permettent de construire des algorithmes de complexité optimale pour la décision de la satisfiabilité.

1.2 Autres arithmétiques

Dans toute la suite de ce manuscrit nous pourrons remplacer l'arithmétique de Presburger par une autre logique. Dans le chapitre concernant les mots paramétrés avec indices, cette logique pourra être arbitraire, sans même s'interpréter forcément sur $\mathbb{Z}$. Si elle n'est pas décidable, les algorithme que nous donnons, seront simplement des semi-algorithmes à moins de disposer d'un oracle qui permet de vérifier si une formule est valide ou non. Dans les chapitres sur les mots paramétrés avec indices et exposants, la logique devra être interprétée sur le domaine $\mathbb{Z}$. De nouveau sa décidabilité ne sera pas nécessaire si l'on dispose d'un oracle. Dans le chapitre sur les mots paramétrés avec indices et produits finis, nous devrons nous restreindre à la logique de Presburger puisque la linéarité des expressions utilisées en indices devient fondamentale dans les algorithmes.

Chapitre 2

Réécriture de mots - Systèmes de Thue

Dans ce chapitre nous allons rappeler les notions classiques de réécriture de mots nécessaires dans la suite de ce manuscrit. Une présentation détaillée est donnée dans le livre de R.V. Book et F. Otto [2].

2.1 Mots

Un alphabet Σ est un ensemble dénombrable d'éléments que nous appellerons *lettres*. L'ensemble de tous les mots construits sur cet alphabet, noté Σ^*, est l'ensemble des suites finies d'éléments de Σ. La suite vide est appelée *mot vide* et est notée ϵ. La longueur d'un mot w, notée $|w|$, est définie par $|\epsilon| = 0$ et $|aw| = 1 + |w|$.

Exemple 3 Donnons nous un alphabet $\Sigma = \{a; b\}$. L'ensemble des mots sur cet alphabet est

$$\Sigma^* = \{\epsilon; a; b; aa; ab; ba; bb; aaa; aab; \cdots\}$$

La longueur du mot $ababa$ est $|ababa| = 5$.

2.2 Équations, modèles, quotients

Fixons un alphabet Σ. Soit R un sous-ensemble du produit cartésien $\Sigma^* \times \Sigma^*$. R est appelé *système de réécriture*. Les éléments de R sont appelés *règles de réécriture* ou plus succinctement *règles*. Une règle $(l, r) \in R$ est toujours notée $l \to r$ et l (resp. r) est appelé membre gauche (resp. droit) de la règle.

Exemple 4 En reprenant l'exemple précédent, $R = \{aa \to bb; aba \to a\}$ est un système de réécriture, ayant deux règles. D'une part aa et aba sont les membres gauches, d'autre part bb et a sont les membres droits.

Sur Σ^*, la relation R induit la *relation de réduction en un pas*, notée $\to_R$ et définie par

$$\to_R = \{(ulv, urv) \ / \ u, v \in \Sigma^* \text{ et } l \to r \in R\}$$

Pour tous mots w_1 et w_2 tels que $w_1 \rightarrow_R w_2$, nous dirons qu'il existe un *pas de réécriture* de w_1 à w_2.

La clôture réflexive et transitive de la relation de réduction en un pas, notée $\rightarrow_R^*$ est appelée *relation de réduction* ou encore *relation de réécriture.*

Exemple 5 Reprenons le système de l'exemple précédent, $R = \{aa \rightarrow bb; aba \rightarrow a\}$, et donnons quelques pas de réécriture.

- $aa \rightarrow_R bb$
- $baab \rightarrow_R bbbb$
- $aaba \rightarrow_R aa$
- $aaba \rightarrow_R bbba$

Dans la relation de réduction nous avons par exemple $aaba \rightarrow_R^* bb$ à cause des deux étapes successives $aaba \rightarrow_R aa \rightarrow_R bb$.

Si $u \rightarrow_R^* v$, nous dirons que v est un *descendant* de u. Nous noterons $\rightarrow_R^+$ la clôture transitive de $\rightarrow_R$.

La clôture réflexive, transitive et *symétrique* et $\rightarrow_R$ est notée $\leftrightarrow_R^*$. C'est clairement une congruence sur Σ^*. Elle est appelée *congruence de Thue* ou encore *relation équationelle* engendrée par R. Deux mots u et v tels que $u \leftrightarrow_R^* v$ sont dits *congruents modulo R.* Notons $[u]_R$ la classe d'équivalence de u, c'est-à-dire

$$[u]_R = \{v \ / \ u \leftrightarrow_R^* v\}$$

Rappelons qu'un monoïde est un ensemble muni d'une opération interne associative possédant un élément neutre. L'ensemble des classes d'équivalences est un monoïde pour l'opération $[u]_R \circ [v]_R = [uv]_R$ avec comme élément neutre $[\epsilon]_R$. Ce monoïde est le quotient

- du monoïde libre engendré par Σ^* muni de l'opération de concaténation des mots et du neutre ϵ
- par la congruence de Thue associée à R.

Ce monoïde sera noté $\Sigma^*/\leftrightarrow_R^*$ ou plus simplement par abus de la notation du quotient mathématique classique Σ^*/R. Le couple (Σ, R) est appelé *présentation équationnelle* du monoïde $\Sigma^*/\leftrightarrow_R^*$. Le monoïde est dit *finiment engendré* si l'alphabet Σ est fini. Il est dit *finiment présenté* si R est fini.

Un mot w est dit en *forme normale* ou *irréductible* pour un système de réécriture R si aucun mot v ne vérifie $w \rightarrow_R v$. Un système de réécriture *termine* si la relation $\rightarrow_R^+$ est noetherienne, c'est-à-dire s'il n'existe pas de chaînes infinies d'étapes de réécritures.

Un système R est dit *localement confluent* si pour tous mots u, v et w tels que $u \rightarrow_R v$ et $u \rightarrow_R w$, il existe un mot m tel que $v \rightarrow_R^* m$ et $w \rightarrow_R^* m$.

Un système R est dit *confluent* si pour tous mots u, v et w tels que $u \rightarrow_R^* v$ et $u \rightarrow_R^* w$, il existe un mot m tel que $v \rightarrow_R^* m$ et $w \rightarrow_R^* m$.

Ces deux notions de confluence sont équivalentes quand le système R termine comme le montre le lemme du à M.H.A. Newman dit "lemme du diamant".

Lemme 1 (lemme du diamant [26]) *Une relation qui termine est confluente si et seulement si elle est localement confluente.*

Deux systèmes de réécriture R_1 et R_2, sur le même alphabet Σ, sont dits *équivalents* s'ils engendrent la même congruence équationelle, c'est-à-dire si $\leftrightarrow^*_{R_1} = \leftrightarrow^*_{R_2}$. Deux systèmes R et R' sur des signatures distinctes sont dits *isomorphes* si les monoïdes engendrés par ces sytèmes sont isomorphes.

Un système qui termine et conflue est dit *convergent*. De tels systèmes sont fondamentaux puisqu'ils permettent d'exprimer de nombreuses propriétées du monoïde présenté par R.

2.3 Utilisation des systèmes convergents

Nous donnons deux applications essentielles des systèmes convergents.

Décision du problème du mot dans le monoïde engendré

Il s'agit de donner un algorihme qui étant donné deux mots u et v de Σ^* dit si $u \leftrightarrow^*_R v$. L'algorithme est simple quand R est convergent puisqu'il suffit de calculer la forme normale de u et de v et celle de vérifier qu'elles sont égales. En général le problème du mot dans un monoïde est indécidable [28]. En 1985, Kapur et Narendran [17] ont exhibé le système $\{aba = bab\}$ dont le monoïde associé a un problème du mot décidable, mais qui n'admet pas de système fini convergent. En ajoutant un symbol c, on peut construire le système isomorphe suivant :

$$\left\{ \begin{array}{l} ab \to c; \\ ca \to bc; \\ bcb \to cc; \\ ccb \to acc \end{array} \right\}$$

Il est bien convergent. Squier établit ensuite, en 1987, dans [30] qu'il existe des monoïdes qui ne sont pas présentables par des systèmes finis convergents, même en ajoutant un nombre arbitraire de symboles. Squier exhibe l'exemple suivant :

$$\left\{ \begin{array}{l} ab \to a; \\ da \to ac; \\ d'a \to ac \end{array} \right\}$$

qui possède cette propriété. Notons par contre qu'il existe un système canonique infini qui présente ce monoïde :

$$\left\{ \begin{array}{l} ac^n b \to ac^n \text{ pour tout } n \geq 0; \\ da \to ac; \\ d'a \to ac \end{array} \right\}$$

Squier donne une caractérisation algébrique fondée sur la notion d'homotopie des systèmes finiment engendrés.

Notons qu'à toute présentation finie d'un monoïde, nous pouvons ajouter des règles afin de présenter le *groupe* engendré par ce monoïde. Il suffit pour chaque lettre de l'alphabet d'ajouter une nouvelle lettre a' et les règles $aa' \to \epsilon$ et $a'a \to \epsilon$. a' représente ainsi l'inverse de a. Le monoïde engendré par ce système enrichi est bien un groupe.

Si ce dernier est fini, nous pouvons appliquer l'algorithme de résolution du problème du mot à de nombreux problèmes de théorie des groupes. Par exemple :

- Le calcul des constantes de structures,
- La recherche des sous-groupes distingués,
- La présentation du centre de l'anneau du groupe.

Étude des formes normales

E. Contejean , A. Coste et moi-même avons donné dans [11] un algorithme qui permet de calculer le cardinal d'un groupe (ou d'un monoïde) finiment présenté par un système convergent. Dans le cas où le groupe présenté est infini nous exhibons des familles infinies d'éléments distincts deux à deux. Ce travail possède une application en physique théorique et permet d'étudier précisemment des modèles d'Ising à deux dimensions. La technique développée est la suivante :

- Nous calculons un automate de formes normales, c'est-à-dire un automate deterministe dont le langage est exactement l'ensemble des mots irréductibles pour le système de réécriture initial. Nous proposons pour cela un algorithme efficace de construction de l'automate de forme normal inspiré de nos travaux antérieurs [24, 25]. Cet algorithme évite l'utilisation des coûteuses opérations booléennes sur les automates.
- Nous dénombrons le nombre de chemins acceptants distincts dans l'automate construit. Chaque chemin correspondant à un élément différent de la structure engendrée, nous obtenons le cardinal du groupe (ou du monoïde) s'il est fini. Dans le cas contraire, nous pouvons exhiber un langage régulier dont tous les éléments sont distincts dans le groupe (ou le monoïde).

Cette technique est implantée dans le logiciel CiME[12] et permet de traiter raisonablement des exemples de taille importante. Nous avons pu par exemple dénombrer les éléments du groupe symétrique $\mathcal{S}_n$ pour des valeur de n allant jusqu'à $n = 20$.

2.4 Recherche de systèmes convergents

Étant donné un système de réécriture R, il est fondamental de trouver des systèmes convergents qui lui sont équivalents ou isomorphes.

La première technique développée pour trouver un système localement confluent à partir d'un système non localement confluent est la *complétion*. Introduite par Knuth et Bendix en 1970 dans [19], cette technique, quand elle termine, permet de contruire un système localement confluent équivalent au système initial.

Cet algorithme est fondé sur la notion de *paires critiques* que Knuth et Bendix définissent formellement les premiers. Mais cette notion était implicite dans les travaux de Dehn concernant les groupes discrets comme l'a montré clairement P. Le Chenadec en 1987 [14, 21]. Nous retrouvons aussi cette notion pour le cas des bases de Gröbner[5, 22, 23].

L'idée principale est d'étudier toutes les superpositions entre les membres gauches des règles afin de trouver toutes les sources de non confluence locale. Plus précisemment, étant données deux règles $l_1 \to r_1$ et $l_2 \to r_2$ nous énumérons tous les mots qui ont l_1 et l_2 pour facteur, dont une position est à la fois une lettre de l_1 et une lettre de l_2 et qui sont de longueur au plus $|l_1| + |l_2|$. Il existe un nombre fini de tel mots, notons u un de ceux-ci. Clairement u est réductible par les règles $l_1 \to r_1$ et $l_2 \to r_2$ en un pas. Notons u_1 et u_2 les réduits respectifs. La paire (u_1, u_2) est appelée *paire critique* des deux règles. Si u_1 et u_2 ont un réduit commun, cette paire est dite *joignable* ou confluente.

Lemme 2 *Un système est localement confluent si et seulement si toutes ses paires critiques sont joignables.*

La complétion nécessite un ordre noethérien sur les mots. Notons un tel ordre $>$. Le principe est de forcer la confluence des paires critiques en ajoutant des règles. Ainsi si (u_1, u_2) est une paire critique nous ajoutons la règle $u_1 \to u_2$ (resp. $u_2 \to u_1$) si $u_1 > u_2$ (resp. si $u_2 > u_2$). Si u_1 et u_2 sont incomparables, l'algotihme échoue. Nous recommençons jusqu'à ce que toutes les paires critiques entre toutes les règles aient été traitées. Ce processus peut bien sûr ne pas terminer, chaque nouvelle règle introduite pouvant créér de nouvelles paires critiques. Dans ce cas une analyse de la divergence peut permettre de trouver des règles à ajouter.

La tentative de maîtrise de cette divergence a amené à considérer des systèmes de réécriture infinis présentant un certain nombre de régularités que nous allons présenter dans le chapitre suivant.

2.5 Réécriture contrainte et paramétrisation

Il existe un certain de nombre de formalismes qui ont été développés depuis un peu plus de 15 ans qui permettent de raisonner sur des familles de système de réécritures. Initialement ces formalismes étaient conçus pour tenter de maîtriser les problèmes de divergence de la complétion de Knuth et Bendix. Ainsi dans [18] les auteurs introduisent la notion de "méta-règle" afin de pouvoir schématiser des ensembles infinis de règles engendré par des complétions à succès infini. Ces présentations sont liées à la réécriture contrainte multi-sortée qui sert de support à l'établissement d'une procédure de complétion des systèmes finis de méta-règles. Les contraintes permettent en effet de représenter de façon finie des ensembles infinis. De plus les contraintes sont aussi utilisées afin d'optimiser la réécriture modulo certaines théories en reportant les coûteux calculs d'unification modulo dans la contrainte.

Un autre formalisme de schématisation de règles est très lié à ces études de divergences : il s'agit de la réécriture conditionnelle. Il est particulièrement utilisé pour définir la logique équationnelle avec appartenance. Dans ces formalismes, la logique utilisée dans les contraintes a le même domaine que les termes eux-mêmes.

Dans le cadre de la réécriture contrainte, les paramètres sont vus comme des variables de sorte *nat* et les constructeurs sont de sorte $nat * term \to term$. On construit les termes paramétrés sur la signature

$$\Sigma = \{a : \mathrm{int} * \mathrm{term} \to \mathrm{term}; \epsilon : \mathrm{term}\}$$

Par exemple le mot paramétré $a_i a_{i+1} | i > 0$ peut s'écrire $a(i, a(i+1, \epsilon)))|i > 0$. Cette approche présente un inconvénient majeur : les spécifications sont peu lisibles et éloignées des notations mathématiques classiques. La notation en présence d'exposants est bien plus complexe pour les termes que pour les mots, comme en témoigne le terme $[a(i, a(j(\diamond)))]^k(\epsilon)$ comparé au mot équivalent $(a_i a_j)^k$. Un autre point technique gênant avec la définition à l'aide de termes est l'expression des contextes : ils nécessitent des variables d'ordre 2 que l'on doit manipuler avec précautions[8, 9].

Nous verrons dans la section 5 page 65 que notre formalisme permet de mélanger les indices et les exposants et d'exprimer par exemple le mot $(a_i a_j)^{i+j} | 0 \leq i + j < N$. Les travaux sur la réécriture de termes avec exposants comme [6, 10] supposent toujours que les variables d'exposants sont indépendantes des variables de termes. Nous lèverons partiellement cette restriction dans ce manuscrit.

Deuxième partie

La réécriture paramétrée

Chapitre 3

Signatures paramétrées

3.1 Introduction

Dans la théorie des ensembles, on utilise classiquement deux façons de décrire un ensemble : en extension ou bien en compréhension. Pour décrire des ensembles finis de taille donnée, les deux formulations sont équivalentes. En revanche, dès que l'on veut manipuler symboliquement des ensembles infinis ou bien des ensembles dont la taille dépend d'un paramètre, on doit utiliser une notation en compréhension. Ce raisonnement s'applique aux alphabets qui sont des ensembles de lettres, briques de base de la réécriture de mots. Si l'on veut manipuler formellement des alphabets de taille paramétrée, ou infinis, on doit utiliser la syntaxe formelle de la définition en compréhension des ensembles en explicitant les paramètres. Un paramètre formel est un symbole qui s'interprète dans un ensemble que nous préciserons. Souvent nous l'interpréterons dans les entiers relatifs. Ainsi, si l'on veut décrire l'alphabet $\{a_1; \cdots; a_N\}$ pour un paramètre formel $N \in \mathbb{Z}$, on va utiliser la notation $\{a_i | 1 \leq i \leq N\}_{\{N\}}$. L'alphabet infini $\{a_0; a_1; a_2; \cdots\}$ sera noté $\{a_i | i \geq 0\}$. Ces alphabets sont indispensables dès que l'on veut raisonner de façon uniforme sur des structures comme les groupes cycliques, les groupes des permutations ou toute autre structure algébrique possédant une représentation équationelle dépendant d'un paramètre formel, mais ayant une "certaine uniformité" suivant les valeurs du paramètre.

Dans ce chapitre, nous allons d'abord présenter les définitions et les notations des *signatures paramétrées*, puis donner une sémantique qui sera utilisée dans toute la suite de cette thèse.

3.2 Syntaxe

Les signatures paramétrées ont une syntaxe formelle similaire à celle des ensembles définis en compréhension à l'aide d'une formule logique. La logique doit être décidable pour que l'on puisse utiliser la signature. Dans toute la suite de ce document, on considérera l'arithmétique de Presburger [29] qui suffira pour la pluspart des exemples. Plus généralement, on pourra remplacer l'arithmétique de Presburger par toute logique du premier ordre satisfaisant les hypothèses de la section 1.2.

Dans tout ce chapitre, nous fixons un ensemble infini dénombrable I dont les éléments sont appelés *variables*. Les variables sont notées par des lettres minuscules. I ne sera jamais précisé.

Définition 6 *On appelle* signature paramétrée *un triplet* $\Sigma = (e, \varphi, \mathcal{N})$ *que l'on note*

$$\Sigma = \{a_e | \varphi\}_{\mathcal{N}}$$

avec

- $\mathcal{N}$ *un ensemble fini d'éléments appelés* paramètres *tel que* $I \cap \mathcal{N} = \emptyset$. *Ses éléments sont notés par des lettres majuscules.*
- φ *une formule de l'arithmétique de Presburger ayant comme variables des éléments de* $\mathcal{N}$ *et de* I.
- e *une combinaison linéaire à coefficients entiers d'éléments de* $\mathcal{N}$ *et de* I.

Pour alléger les notations, on omettra souvent $\mathcal{N}$ *s'il est vide ainsi que* φ *si c'est une tautologie.*

Exemple 7 Les exemples suivants seront repris dans la suite de ce chapitre.

- $\Sigma_1 = \{a_P | P = 0\}_{\{P\}}$.
- $\Sigma_2 = \{a_i | 1 \leq i \leq N-1\}_{\{N\}}$. Cette signature correspond à la représentation du groupe symétrique $\mathcal{S}_N$, où chaque a_i représente la transposition $(i \quad i+1)$.
- $\Sigma_3 = \{a_i | 0 \leq i \leq 3\}_{\{N\}}$. On note que cette signature ne dépend pas de son paramètre N.
- $\Sigma_4 = \left\{ a_{\frac{i+N}{2}} | 0 \leq \frac{N+P}{2} \leq N-1 \right\}_{\{N;P\}}$.

3.3 Sémantique

Chaque signature paramétrée s'interprète comme une famille d'alphabets finis ou infinis. On pourra donc raisonner sur la représentation symbolique de la signature paramétrée, puis essayer d'interpréter les propriétés prouvées dans les alphabets classiques correspondants.

Définition 8 *Pour toute interprétation* $\nu : \mathcal{N} \to \mathbb{Z}$, *la* ν-interprétation *est la fonction* $[\![.]\!]_\nu$ *qui associe à une signature paramétrée l'alphabet obtenu en instanciant les paramètres par* ν *et les variables par des entiers quelconques satisfaisant respectant la contrainte* φ.

$$[\![\{a_e|\varphi\}_N]\!]_\nu = \bigcup_{\substack{\mu : I \to \mathbb{Z} \\ \nu(\mu(\varphi)) \text{ vraie}}} \left\{a_{\nu(\mu(e))}\right\}$$

Cet alphabet peut être éventuellement vide ou infini.

Notons que ν est implicitement étendue en un morphisme des expressions de l'arithmétique de Presburger.

Exemple 9 Reprenons les signatures paramétrées de l'exemple 7 page précédente et donnons leurs différentes ν-interprétations.

- Pour Σ_1 :
  - si $\nu(P) = 0$ alors $[\![\Sigma_1]\!]_\nu = \{a_0\}$
  - si $\nu(P) \neq 0$ alors $[\![\Sigma_1]\!]_\nu = \emptyset$
- Pour Σ_2 :
  - si $\nu(N) = 0$ alors $[\![\Sigma_2]\!]_\nu = \emptyset$
  - si $\nu(N) = 1$ alors $[\![\Sigma_2]\!]_\nu = \emptyset$
  - si $\nu(N) = n \in \mathbb{Z}$ et $n \geq 2$ alors $[\![\Sigma_2]\!]_\nu = \{a_1; \cdots; a_{n-1}\}$

  Dans ce cas, il y a une infinité de signatures finies distinctes.
- Pour Σ_3, quelle que soit l'interprétation ν, $[\![\Sigma_3]\!]_\nu = \{a_0; a_1; a_2; a_3\}$. Notons que l'interprétation est indépendante de la valeur du paramètre.
- pour $\Sigma_4 = \left\{a_{\frac{i+N}{2}} | 0 \leq i \leq \frac{N+P}{2} \leq N-1\right\}_{\{N;P\}}$ nous avons par exemple :
  - si $\nu(N) = 0$ alors $[\![\Sigma_4]\!]_\nu = \emptyset$
  - si $\nu(N) = 3$ et $\nu(P) = 1$ alors $[\![\Sigma_4]\!]_\nu = \{a_1; a_2\}$

Par abus de notation, on s'autorisera quand il n'y a pas d'ambiguïté à noter N au lieu de $\{N\}$ le singleton contenant le paramètre N.

On peut très facilement généraliser la définition de deux façons. Tout d'abord on peut autoriser plus d'une lettre de base. On parlera par exemple de la signature $\{a_i; b_j | 1 \leq i \leq N-1 \ \wedge\ j > 0\}_N$. On peut aussi accepter des n-uplet d'indices au lieu d'indices uniques comme dans l'exemple $\{a_{i,j} | 1 \leq i \leq N-1 \ \wedge\ j > 0\}_N$. Toutes les définitions et preuves des sections suivantes s'étendent naturellement à ces deux cas. Les exemples de la suite utiliseront ces extensions, mais les algorithmes et les preuves seront toujours donnés conformément à la définition formelle avec une seule lettre et un seul indice.

Quand une signature paramétrée est fixée, nous devons adapter la notion habituelle de variable d'une expression et d'une formule afin de pouvoir distinguer clairement le rôle de paramètres de celui des variables de la signature.

Définition 10 *Soit e une expression arithmétique, $Var(e)$ désigne l'ensemble de ses variables qui ne sont pas des paramètres. Pour une formule φ nous définissons $Var(\varphi)$ l'ensemble des variables libres de la formule φ qui ne sont pas des paramètres.*

3.4 Signatures réduites

Dans cette section, nous allons montrer que les signatures paramétrées peuvent être sans perte de généralité mises dans une forme plus simple dite *réduite*. Nous commençons par une définition de l'équivalence entre deux signatures.

Définition 11 *Deux signatures Σ_1 et Σ_2 sont dites* équivalentes *si et seulement si pour toute interprétation ν des paramètres, leurs ν-interprétations sont égales c'est-à-dire :*

$$\forall \nu : \mathcal{N} \to \mathbb{Z}, [\![\Sigma_1]\!]_\nu = [\![\Sigma_2]\!]_\nu$$

Exemple 12
– $\{a_i|0 < i < N\}$ est équivalente à $\{a_{j+1}|0 \leq j < N-1\}$
– $\{a_i|0 < i < N\}$ n'est pas équivalente à $\{a_{j+1}|0 < j < N\}$

Définition 13 *Une signature est dite* réduite *si elle est de la forme :*

$$\Sigma = \{a_i|\varphi\}_{\mathcal{N}}$$

où $i \in I$ est l'unique variable libre de φ.

Exemple 14 Avec la convention que les minuscules désignent des variables et les majuscules des paramètres, la signature $\{a_i|N > i > 0\}$ est réduite, mais les deux suivantes ne le sont pas :
– $\{a_{i+j}|j > i > 0\}$: l'indice n'est pas une variable.
– $\{a_i|i > k\}$: la contrainte a une variable libre k.

Proposition 1 *Toute signature paramétrée est équivalente à une signature réduite.*

Cette proposition se déduit immédiatement du lemme suivant, qui permet de faire un simple changement de variable dans les signatures.

Lemme 3 *Pour toute interprétation $\nu : \mathcal{N} \to \mathbb{Z}$ des paramètres de la signature $\{a_e|\varphi\}_{\mathcal{N}}$, si l'on note $Var(e) \cup Var(\varphi) = \{x_1; \cdots; x_k\}$,*

$$[\![\{a_e|\varphi\}]\!]_\nu = [\![\{a_i|\exists x_1 \cdots x_k\ \varphi\ \wedge\ e = i\}]\!]_\nu$$

où $i \in I$ et $i \notin Var(e) \cup Var(\varphi)$

Preuve On se donne ν une interprétation des paramètres. Soient

$$E_1 = [\![\{a_e|\varphi\}_{\mathcal{N}}]\!]_\nu$$

$$E_2 = [\![\{a_i|\exists x_1 \cdots x_k\ \varphi\ \wedge\ e = i\}]\!]_\nu$$

Donnons-nous $j \in \mathbb{Z}$ tel que $a_j \in E_1$. Montrons que $a_j \in E_2$. On pose $\mu(i) = j$. Il suffit de vérifier que $\nu(\mu(\exists x_1 \cdots x_k\ \varphi\ \wedge\ e = i)$ est vraie puisque i est la seule variable libre de $\exists x_1 \cdots x_k\ \varphi\ \wedge\ e = i$. Comme $a_j \in E_1$, il existe une interpétation μ_0 de support $Var(\varphi) \cup Var(e)$ telle que $\mu_0(e) = j$ et $\nu(\mu_0(\varphi))$ est vraie. Donc $\nu(\mu(\exists x_1 \cdots x_k \varphi\ \wedge\ e = i)$ est vraie en interprétant les x_i par $\mu_0(x_i)$ et e par $\mu_0(e)$. Il est donc établi que $E_1 \subset E_2$. Réciproquement, on se donne $j \in \mathbb{Z}$ tel que $a_j \in E_2$. Montrons que $a_j \in E_1$. La formule $\exists x_1 \cdots x_k\ \varphi\ \wedge\ e = j$ est vraie, donc on peut construire une interprétation μ des x_i telle que $\nu(\mu(\varphi))$ est vraie et $\nu(\mu(e)) = j$. La réciproque est démontrée. ◇

Exemple 15 La signature $\{a_{2j}|1 \leq j+k \leq P\}_{\{P\}}$ est équivalente à $\{a_i|\exists j \exists k\ i = 2j\ \wedge\ 1 \leq j + k \leq P\}_{\{P\}}$.

Remarque En appliquant le théorème 1 page ci-contre, nous pouvons toujours nous ramener sans perte de généralité au cas où la signature est réduite. Pour la théorie, nous nous restreindrons toujours à ce cas, mais les exemples utiliseront souvent des expressions comme indice.

Notons qu'un alphabet classique peut toujours trivialement être vu comme une signature paramétrée. Il suffit d'associer un entier à chaque symbole de l'alphabet. Par exemple $\{a; b; c\}$ peut être représenté par $\{a_i | i = 1 \vee i = 2 \vee i = 3\}$.

Chapitre 4

Réécriture paramétrée avec indices

4.1 Mots paramétrés avec indices

Dans toute cette section nous fixons un ensemble infini dénombrable de variables I et une signature paramétrée réduite $\Sigma = \{a_i|\varphi\}_{\mathcal{N}}$ avec $i \in I$.

4.1.1 Définition et sémantique

Définition 16 *Soit k un entier positif ou nul. Un* mot paramétré avec indices *w est défini par*

$$a_{e_1} \cdots a_{e_k}|\psi$$

avec :

- *ψ une formule de l'arithmétique de Presburger dont les variables sont dans $I \cup \mathcal{N}$, appelée* contrainte du mot*,*
- *k un entier positif ou nul,*
- *$e_1, \cdots, e_k$ des expressions arithmétiques sur les entiers dont les variables sont dans $I \cup \mathcal{N}$.*

Si $k = 0$ le mot w sera noté $\epsilon|\psi$ et appelé mot vide.

On dira que a_{e_i} pour i entre 1 et k est la lettre *du mot w à la position i.*

La longueur k du mot w *est notée $|w|$.*

Nous notons de plus $w|_{p,l} = a_{e_p} \ldots a_{e_{p+l}}$ le facteur de w à la position p et de longueur $l+1$. $Var(u|\psi)$ désigne l'ensemble $Var(e_1) \cup \ldots \cup Var(e_k) \cup Var(\psi)$.

Dans toute la suite de cette section, les mots paramétrés avec indices seront appelés simplement "mots paramétrés" s'il n'y a pas d'ambiguïté.

Nous nous autoriserons à omettre la contrainte d'un mot paramétré si cette dernière est une tautologie : le mot $w|\top$ sera noté w.

Nous donnons maintenant une sémantique pour les mots paramétrés.

Définition 17 *Étant donnée une interprétation $\nu : \mathcal{N} \to \mathbb{Z}$,* la ν-interprétation d'un mot paramétré *est la fonction partielle définie sur l'ensemble des interprétations $\mu : I \to \mathbb{Z}$*

par :

- $[\![a_{e_1} \cdots a_{e_k}|\psi]\!]_\nu(\mu) = a_{\nu(\mu(e_1))} \cdots a_{\nu(\mu(e_k))}$ *si*
  1. $\nu(\mu(\psi))$ *est vraie et*
  2. *pour tout* $1 \leq r \leq k$, $a_{\nu(\mu(e_r))} \in [\![\Sigma]\!]_\nu$ *est vraie (c'est-à-dire* $(\nu(\mu(\varphi(e_r)))$ *est vraie)*
- $[\![a_{e_0} \cdots a_{e_k}|\psi]\!]_\nu(\mu)$ *n'est pas définie sinon.*

Comme ν et μ sont à valeur dans $\mathbb{Z}$ et à supports disjoints, elles commutent.

Dans cette définition nous remarquons qu'une ν-interprétation d'un mot paramétré est une fonction dont le résultat est un mot construit sur la ν-interprétation de la signature paramétrée.

Exemple 18 Nous construisons des mots sur les signatures définies dans l'exemple 7 page 28.

- Dans $\Sigma_1 = \{a_P|P = 0\}_{\{P\}}$
  - si $\nu(P) = 0$ alors $[\![a_P a_P a_P]\!]_\nu(\mu) = a_0 a_0 a_0$ pour tout μ.
  - si $\nu(P) \neq 0$ alors $[\![a_P a_P a_P]\!]_\nu(\mu)$ est indéfini pour tout μ.
- Dans $\Sigma_2 = \{a_i|1 \leq i \leq N-1\}_{\{N\}}$
  - si $\nu(N) = 0$ ou $\nu(N) = 1$ alors $[\![a_i a_1]\!]_\nu(\mu)$ est indéfini pour tout μ.
  - si $\nu(N) = 2$ et $\mu(i) = 1$ alors $[\![a_i a_1]\!]_\nu(\mu) = a_1 a_1$
  - si $\nu(N) = 3$ alors
    - si $\mu(i) = 1$ alors $[\![a_i a_1]\!]_\nu(\mu) = a_1 a_1$
    - si $\mu(i) = 2$ alors $[\![a_i a_1]\!]_\nu(\mu) = a_2 a_1$
    - sinon $[\![a_i a_1]\!]_\nu(\mu)$ est indéfini.

4.1.2 Substitutions et renommages des mots paramétrés

Nous étendons la définition des substitutions sur les expressions et les contraintes.

Définition 19 *Soit* $\sigma : I \cup \mathcal{N} \to Expr(I, \mathcal{N})$ *une substitution des variables et des paramètres de* Σ. *La substitution du mot paramétré* $a_{e_1} \dots a_{e_k}|\psi$ *par* σ *est le mot paramétré*

$$\sigma(a_{e_1} \dots a_{e_k}|\psi) = a_{\sigma(e_1)} \dots a_{\sigma(e_k)}|\sigma(\psi)$$

Une bijection entre deux sous-ensembles (non nécessairement disjoints) de I *sera appelée* renommage. *Si les deux ensembles sont disjoints, le renommage est idempotent. PAr la suite nous ne considérerons que des renommages idempotents.*

Exemple 20 Posons $\sigma(i) = j$, nous avons $\sigma(a_i|i > N) = a_j|j > N$. C'est bien un renommage idempotent.

Définition 21 *Soit* $u|\psi$ *un mot paramétré. Une substitution* σ *qui transforme des variables de* $Var(u|\psi)$ *en des variables qui n'apparaissent pas dans* $Var(u|\psi)$ *est appelée renommage de* $u|\psi$. *Nous dirons aussi que* $\sigma(u|\psi)$ *est un renommage de* $u|\psi$.

Lemme 4 *Soit ν une interprétation des paramètres et $u|\phi$ un mot paramétré. L'ensemble $\{[\![u|\phi]\!]_\nu(\mu) \ / \ \mu : I \to \mathbb{Z}\}$ est invariant par renommage des variables de $u|\phi$.*

Preuve En effet si ρ est un renommage des variables de $u|\phi$, $\{[\![u|\phi]\!]_\nu(\mu) \ / \ \mu : I \to \mathbb{Z}\}$ est égal à $\{[\![\rho(u|\phi)]\!]_\nu(\mu) \ / \ \mu \circ \rho : I \to \mathbb{Z}\}$. $\diamond$

Lemme 5 *Soient ν une interprétation des paramètres, μ une interprétation des variables et $u|\phi$ un mot paramétré de longueur $k \in \mathbb{N}$. Si le mot $[\![u|\phi]\!]_\nu(\mu)$ est défini, il est de longueur k.*

Preuve Par définition, les mots du codomaine de la ν-interprétation ont tous la même longueur que le mot paramétré. $\diamond$

Pour simplifier l'écriture des contraintes, nous introduisons la notation suivante :

Définition 22 *Soient $u|\psi = a_{e_1} \dots a_{e_k}|\psi$ et $u'|\psi' = a_{e'_1} \dots a_{e'_k}|\psi'$ deux mots paramétrés de même longueur. La contrainte de l'arithmétique de Presburger*

$$\bigwedge_{1 \le j \le k} e_j = e'_j$$

est notée

$$u \equiv v$$

Si u et v sont deux mots de longueurs distinctes, $u \equiv v$ désigne la formule $\perp$.

4.1.3 Équivalence globale - Forme réduite

Définition 23 *Deux mots paramétrés $u|\phi$ et $v|\psi$ sont dits globalement équivalents si et seulement si pour toute interprétation des paramètres ν*

$$\{[\![u|\phi]\!]_\nu(\mu) \ / \ \mu : I \to \mathbb{Z}\} = \{[\![v|\psi]\!]_\nu(\mu) \ / \ \mu : I \to \mathbb{Z}\}$$

Exemple 24 Les mots $a_i a_j | \exists i' \ i = 2i' \ \wedge \ i' + j = N$ et $a_{2k} a_l | k = N - l$ sont globalement équivalents. Notons que la notion d'équivalence globale est un peu plus générale que la simple notion habituelle de renommage des variables.

Deux mots paramétrés globalement équivalents décrivent le même ensemble de mots dans l'interprétation de la signature. C'est donc une notion sémantique. Nous verrons, par contre, plus loin que l'équivalence globale, n'est pas une notion assez fine pour décrire les relations syntaxiques qui peuvent lier les mots.

Proposition 2 *Pour tout mot paramétré $u|\psi$, il existe un mot paramétré $v|\xi$, non nécessairement unique, tel que :*

- *$u|\psi$ et $v|\xi$ sont globalement équivalents*
- *$v|\xi$ est de la forme $a_{i_1} \dots a_{i_k}|\xi$ où les i_k sont des variables deux à deux distinctes et sont les seules variables libres de ξ.*

Le mot paramétré $v|\xi$ *est dit en* forme réduite simple*.*

La preuve de cette proposition découle immédiatement du lemme suivant qui donne un changement de variable.

Lemme 6 *Soit* $a_{e_1} \cdots a_{e_k}|\psi$ *un mot paramétré de longueur* k *tel que* $Var(a_{e_1} \cdots a_{e_k}|\psi) = \{j_1; \cdots; j_l\}$ *. Pour toute interprétation* ν *des paramètres*

$$\{[\![a_{e_1} \cdots a_{e_k}|\psi]\!]_\nu(\mu) \ / \ \mu : I \to \mathbb{Z}\}$$
$$=$$
$$\{[\![a_{i_1} \cdots a_{i_k}|\exists j_1 \ldots \exists j_l \ i_1 = e_1 \ \wedge \ \ldots \ \wedge \ i_k = e_k \ \wedge \ \psi]\!]_\nu(\mu) \ / \ \mu : I \to \mathbb{Z}\}$$

si $\{i_1, \cdots, i_k\} \cap \{j_1; \cdots; j_l\} = \emptyset$*.*

Preuve Fixons une interprétation des paramètres ν. De façon similaire à la preuve du lemme 3 page 30, nous montrons que la double inclusion de

$$E_1 = \{[\![a_{e_1} \cdots a_{e_k}|\psi]\!]_\nu(\mu) \ / \ \mu : I \to \mathbb{Z}\}$$

et

$$E_2 = \{[\![a_{i_1} \cdots a_{i_k}|\exists j_1 \ldots \exists j_l \ i_1 = e_1 \ \wedge \ \ldots \ \wedge \ i_k = e_k \ \wedge \ \psi]\!]_\nu(\mu) \ / \ \mu : I \to \mathbb{Z}\}$$

- $E_1 \subset E_2$: soit $u \in E_1$, il existe μ une interprétation des variables $j_1, \ldots, j_l$ telle que $u = \nu(\mu(a_{e_1} \cdots a_{e_k}))$ et $\nu(\mu(\psi))$ est vraie. Définissons l'interprétation μ' telle que pour tout l tel que $1 \leq l \leq k$, $\mu'(i_l) = \mu(e_l)$. Nous avons clairement $u = \nu(\mu'(a_{i_1} \ldots a_{i_k}))$ et
$$\nu(\mu'(\exists j_1 \ldots \exists j_l \ i_1 = e_1 \ \wedge \ \ldots \ \wedge \ i_k = e_k \ \wedge \ \psi))$$
$$=$$
$$\exists j_1 \ldots \exists j_l \ \nu(\mu(e_1)) = \nu(e_1) \ \wedge \ \ldots \ \wedge \ \nu(\mu(e_k)) = \nu(e_k) \ \wedge \ \nu(\psi)$$
Cette dernière formule est vraie puisqu'en instanciant les j_r par $\mu(j_r)$ nous obtenons la formule
$$\mu(\nu(\mu(e_1))) = \mu(\nu(e_1)) \ \wedge \ \ldots \ \wedge \ \mu(\nu(\mu(e_k)) = \mu(\nu(e_k)) \ \wedge \ \mu(\nu(\psi))$$
Comme ν et μ ont des supports disjoints et sont closes, elles commutent. Ainsi en permutant μ et ν nous obtenons une formule exactement équivalente à $\nu(\mu(\psi))$, qui vraie par hypothèse.
- $E_2 \subset E_1$: fixons $u \in E_2$. Il existe μ une interprétation des variables $i_1, \ldots, i_k$ telle que $u = \nu(\mu(a_{i_1} \cdots a_{i_k}))$ et
$$\nu(\mu(\exists j_1 \ldots \exists j_l \ i_1 = e_1 \ \wedge \ \ldots \ \wedge \ i_k = e_k \ \wedge \ \psi))$$
est vraie. Il existe donc μ' une interprétation des variables $j_1, \ldots j_l$ telle que
$$\mu'(\nu(\mu(i_1 = e_1 \ \wedge \ \ldots \ \wedge \ i_k = e_k \ \wedge \ \psi)))$$

est une formule vraie sans variables. En distribuant les interprétations et en les permutant nous obtenons la formule vraie :

$$[\mu(i_1) = \nu(\mu'(e_1)) \ \wedge \ \ldots \ \wedge \ \mu(i_k) = \nu(\mu'(e_k))] \ \wedge \ \nu(\mu'(\psi))$$

dont nous pouvons déduire que $u = [\![a_{e_1} \cdots a_{e_k}|\psi]\!]_\nu(\mu')$ est un élément de E_1.

◇

Exemple 25

Mot paramétré	Mot paramétré réduit simple
$a_{i+j}\|0 < i + j < N$	$a_k\|\exists i \exists j \ k = i + j \ \wedge \ 0 < i + j < N$
$a_{i+2}\|0 < i - j < N$	$a_k\|\exists i \exists j \ k = i + 2 \ \wedge \ 0 < i - j < N$
$a_i a_{i+1} a_{i+2}\|0 < i < N$	$a_k a_l a_m\|\exists i \ k = i \ \wedge \ l = i + 1 \ \wedge \ m = i + 2 \ \wedge \ 0 < i < N$

Nous allons opérer une deuxième réduction syntaxique sans perte de généralité.

En effet, les mots qui vont nous intéresser dans toute la suite de ce document sont ceux dont la contrainte est suffisamment forte pour que toutes les lettres s'interprètent en une lettre de l'interprétation de la signature.

Définition 26 *Un mot paramétré $u = a_{e_1} \cdots a_{e_l}|\psi$ sur la signature réduite $\Sigma = \{a_i|\varphi\}$ est dit* justement contraint *si $\psi \Rightarrow (\varphi[i \mapsto e_1] \ \wedge \ \cdots \ \wedge \ \varphi[i \mapsto e_l])$ est une formule valide.*

Exemple 27 Le mot $a_i|i \geq 2$ sur la signature $\Sigma_2 = \{a_i|1 \leq i \leq N-1\}_{\{N\}}$ n'est pas justement contraint car $i \geq 2 \Rightarrow 1 \leq i \leq N - 1$ n'est pas valide. En effet pour $\nu(N) = 1$, le mot peut sembler avoir une interprétation si l'on ne se préoccupe pas de la signature. Pourtant $i = 2$ vérifie la contrainte du mot et a_2 n'est pas une lettre de $[\![\Sigma_2]\!]$. Par contre le mot $a_i|N - 1 \geq i \geq 2$ est justement contraint.

Remarquons que les deux mots de l'exemple précédent représentent intuitivement le même ensemble de mots. En fait, nous pouvons toujours renforcer la contrainte d'un mot pour qu'il devienne justement contraint.

Lemme 7 *Pour tout mot simple paramétré $u|\phi$, il existe un mot simple paramétré justement contraint $u|\phi'$ globalement équivalent à $u|\phi$.*

Preuve On écrit $u|\phi$ sous la forme $a_{e_1} \cdots a_{e_l}|\phi$. Le mot

$$a_{e_1} \cdots a_{e_l}|\phi \ \wedge \ (\phi \Rightarrow (\varphi[i \mapsto e_1] \ \wedge \ \cdots \ \wedge \ \varphi[i \mapsto e_l])$$

convient. ◇

Les transformations d'un mot paramétré en un mot justement contraint et en un mot simplement réduit peuvent se combiner.

Définition 28 *Un mot est dit de* réduit *s'il est justement contraint et en forme réduite simple.*

Proposition 3 *Tout mot paramétré est globalement équivalent à un mot réduit.*

Preuve

Appliquons la construction de la preuve du lemme 7 page précédente. Nous obtenons un mot $a_{e_1} \dots a_{e_k}|\psi$ justement contraint équivalent au mot de départ. Puis nous appliquons la construction utilisée dans la preuve du lemme 6 page 36. Nous obtenons le mot paramétré simplement réduit

$$a_{i_1} \cdots a_{i_k}|\exists j_1 \dots \exists j_l \; i_1 = e_1 \;\wedge\; \dots \;\wedge\; i_k = e_k \;\wedge\; \psi$$

Il est justement contraint si et seulement si

$$(\exists j_1 \dots \exists j_l \; i_1 = e_1 \;\wedge\; \dots \;\wedge\; i_k = e_k \;\wedge\; \psi) \Rightarrow (\varphi[i \mapsto i_1] \;\wedge\; \cdots \;\wedge\; \varphi[i \mapsto i_k])$$

est valide. Or cette formule est équivalente à la formule

$$\psi \Rightarrow (\varphi[i \mapsto e_1] \;\wedge\; \cdots \;\wedge\; \varphi[i \mapsto e_k]$$

qui exprime la juste contrainte de $a_{e_1} \dots a_{e_k}|\psi$ et est donc valide d'après la remarque faite au début de la pruve. $\diamond$

Dans toute la suite, sauf mention explicite du contraire, tous les mots que nous écrirons seront sous forme réduite. Dans les exemples, nous omettrons souvent la partie de la contrainte que l'on peut déduire de la signature afin d'alléger la notation.

4.1.4 Similitude locale

Définition 29 *Deux mots paramétrés $u|\phi$ et $v|\psi$ sont dits* localement similaires *si et seulement si pour toutes interprétations $\nu : \mathcal{N} \to \mathbb{Z}$ et $\mu : I \to \mathbb{Z}$*

- *soit $\nu(\mu(\phi))$ et $\nu(\mu(\psi))$ sont fausses*
- *soit $\nu(\mu(\phi))$ et $\nu(\mu(\psi))$ sont vraies et*

$$[\![u|\phi]\!]_\nu(\mu) = [\![v|\psi]\!]_\nu(\mu)$$

Exemple 30 Dans l'exemple 24 page 35, nous avons établi que $a_i a_j|i+j = N$ et $a_k a_l|k = N - l$ sont globalement équivalents. Par contre ils ne sont pas localement similaires puisque si nous posons :

- $\nu(N) = 1$
- $\mu(i) = 0$
- $\mu(j) = 1$
- $\mu(k) = 1$
- $\mu(l) = 0$

nous avons bien $\nu(\mu(i + j = N))$ et $\nu(\mu(k = N - l))$ vraies, mais $a_0 a_1 \neq a_1 a_0$. Par contre les deux mots $a_i a_j|i + j < N$ et $a_i a_k|i + k < N \;\wedge\; k = j$ sont localement similaires.

Proposition 4 *La similitude locale est une relation d'équivalence.*

Preuve Comme $\Leftrightarrow$ et l'égalité entre mots sont des relations d'équivalence, la similitude locale en est une aussi. $\diamond$

La similitude locale est ainsi un raffinement de l'équivalence globale. Intuitivement, c'est une relation d'équivalence qui considère que les variables sont globales à toute la signature et ne peuvent être instanciées de deux façons différentes simultanément. A contrario, l'équivalence globale se place purement au niveau sémantique et ne se préoccupe pas des noms de variables partagées entre les mots.

Nous donnons maintenant un critère de décision pour la similitude locale de deux mots paramétrés.

Lemme 8 *Soient deux mots paramétrés $u|\phi$ et $v|\psi$. Ils sont* localement similaires *si et seulement si*

$$\phi \ \wedge \ \psi \Rightarrow u \equiv v \quad \text{et} \quad \phi \Leftrightarrow \psi$$

sont des formules valides.

Preuve
- Supposons que les mots $u|\phi$ et $v|\psi$ sont localement similaires. Fixons une interprétation ν des paramètres et une interprétation μ des variables.
  - Si $\nu(\mu(\phi)) \wedge \nu(\mu(\psi))$ est fausse, la prémisse de la première formule est fausse donc l'implication est vraie. Sinon, en appliquant la définition de la similitude locale, $[\![u|\phi]\!]_\nu(\mu) = [\![v|\psi]\!]_\nu(\mu)$ c'est-à-dire $\nu(\mu(u)) = \nu(\mu(v))$. Nous avons donc établi que $\phi \ \wedge \ \psi \Rightarrow u \equiv v$.
  - $\phi \Leftrightarrow \psi$ est une formule valide par définition de la similitude locale.
- Réciproquement supposons que $\phi \ \wedge \ \psi \Rightarrow u \equiv v$ et $\phi \Leftrightarrow \psi$ sont des formules valides, les deux propriétés de similitude locales sont clairement vérifiées.

$\diamond$

Exemple 31 Les deux mots $a_i|i > 2$ et $a_j|j - 2 > 0$ ne sont pas localement similaires puisque $i > 2 \ \wedge \ j - 2 > 0 \Rightarrow i = j$ n'est pas valide. Ainsi le renommage d'un mot n'est pas localement équivalent à ce mot.

Nous pouvons simplement comparer les relations d'équivalence globale et de similitude locale : si deux mots sont localement similaires alors ils sont globalement équivalents.

4.2 Équations - Règles - Réécriture

4.2.1 Équations de mots

Syntaxe des équations

Définition 32 *Une* équation paramétrée *est un couple $(\{u; v\}, \psi)$ noté $(u = v)|\psi$ où*
- *ψ est une formule de la logique de Presburger et*
- *$u|\psi$ et $v|\psi$ sont deux mots paramétrés tels que le mot paramétré $uv|\psi$ est réduit.*

Exemple 33 Dans $\Sigma_2 = \{a_i | 1 \leq i \leq N-1\}_{\{N\}}$, $(a_j a_i = a_k a_l) | j = l \ \wedge \ i = k \ \wedge \ (i - j \geq 2) \ \wedge \ (1 \leq i, j \leq N-1)$ est une équation paramétrée.

Nous pouvons noter qu'il faut veiller à ne pas confondre le symbole = de l'arithmétique de Presburger avec le = syntaxique que nous avons introduit pour désigner les équations paramétrées.

Une équation s'interprète comme un ensemble d'équations entre mots au sens usuel.

Définition 34 *Soit ν une interprétation des paramètres. La ν-*interprétation *d'une équation paramétrée $(u = v)|\psi$ est définie comme l'ensemble des équations obtenues en instanciant les paramètres par ν puis les variables par des entiers satisfaisant la contrainte de l'équation.*

$$[\![(u = v)|\psi]\!]_\nu = \bigcup_{\substack{\mu : I \to \mathbb{Z} \\ \nu(\mu(\psi)) \text{ est vraie}}} \{\nu(\mu(u)) = \nu(\mu(v))\}$$

La ν-interprétation d'une équation paramétrée est donc un ensemble (fini, infini ou vide) d'équations entre mots construits sur la ν-interprétation de la signature.

Exemple 35 Dans $\Sigma_2 = \{a_i | 1 \leq i \leq N-1\}_{\{N\}}$,

$$[\![(a_i a_j = a_k a_l) | i = l \ \wedge \ j = k \ \wedge \ (i - j \geq 2) \ \wedge \ (1 \leq i, j \leq N-1)]\!]_{N \mapsto 5} \\ = \\ \{a_3 a_1 = a_1 a_3; a_4 a_1 = a_1 a_4; a_4 a_2 = a_2 a_4\}$$

Lemme 9 *La ν-interprétation d'une équation paramétrée est invariante par renommage des variables.*

Preuve C'est clair puisque les variables sont toutes collectivisées dans l'ensemble. ◇

Nous pouvons remarquer que dans la définition d'une équation, la restriction $uv|\psi$ réduit ne limite pas la généralité d'après la proposition 2 puisque nous pouvons toujours réduire l'équation en une équation équivalente où les deux membres sont réduits.

Étape équationnelle

Définition 36 *Le mot paramétré $u|\psi$ est convertible en un pas en un mot paramétré $v|\chi$ par l'équation $l = r|\xi$ à la position $p \in \mathbb{N}$, et on note*

$$u|\psi \xleftrightarrow[p]{l=r|\xi} v|\chi$$

si et seulement si

il existe une substitution σ de support $Var(l = r|\xi)$ telle que

1. *Conditions de filtrage :*
   (a) *u a un facteur de longueur $|l|$ à la position p, i.e.*

$$\begin{array}{l} u = a_{e_0} \cdots a_{e_{p-1}} a_{e_p} \cdots a_{e_{p+|l|-1}} \; a_{e_{p+|l|}} \cdots a_{e_m} \\ l = \qquad\qquad\quad a_{i_1} \cdots a_{i_{|l|}} \end{array}$$

(b) $\left\{ \begin{array}{lcl} \sigma(i_1) & = & e_p \\ & \vdots & \\ \sigma(i_{|l|}) & = & e_{p+|l|+1} \end{array} \right.$

(c) $\psi \Rightarrow \exists x_1 \cdots \exists x_n \; \sigma(\xi)$ *est valide, où* $\{x_1; \cdots; x_n\} = Var(\sigma(\xi)) \setminus Var(\psi)$.

2. *Conditions de production :*
   (a) $v = a_{i_1} \cdots a_{i_{p-1}} \sigma(r) a_{i_{p+|l|}} \cdots a_{i_m}$ *et*
   (b) $\chi \Leftrightarrow \psi \;\wedge\; \sigma(\xi)$ *est valide.*

Commentons chaque condition.

1. Les conditions de filtrage sont exactement celles qui permettent de dire que l'équation peut s'appliquer sur u. Leur but est
   (a) de préciser les notations et de vérifier les contraintes de longueur,
   (b) d'assurer que $\sigma(l)$ est un facteur de u à la position p
   (c) de vérifier la condition d'application de la règle. Les variables quantifiées existentiellement correspondent aux variables du membre droit de l'équation qui n'apparaissent pas dans le membre gauche de l'équation.
2. Les conditions de production permettent de s'assurer que v est bien issu de u par l'application de l'équation :
   (a) $\sigma(r)$ est un facteur de v à la position p, le reste du mot étant inchangé,
   (b) les contraintes du mot produit sont équivalentes à celles du mot original plus celles induites par l'application de l'équation. Ces dernières contiennent des relations entre les nouvelles variables introduites par la règle.

Proposition 5 (Sémantique d'une étape équationnelle) *Si*

$$u|\psi \xleftrightarrow[p]{l=r|\xi} v|\chi$$

est une étape équationnelle alors pour tout $\nu : \mathcal{N} \to \mathbb{Z}$ *et tout* $\mu : I \to \mathbb{Z}$ *tels que* $[\![u|\psi]\!]_\nu(\mu)$ *et* $[\![v|\chi]\!]_\nu(\mu)$ *sont définis,*

$$[\![u|\psi]\!]_\nu(\mu) \xleftrightarrow[p]{[\![l=r|\xi]\!]_\nu} [\![v|\chi]\!]_\nu(\mu)$$

Preuve Fixons $\nu : \mathcal{N} \to \mathbb{Z}$ et $\mu : I \to \mathbb{Z}$ tels que $u_0 = [\![u|\psi]\!]_\nu(\mu)$ et $v_0 = [\![v|\chi]\!]_\nu(\mu)$ sont définis, c'est-à-dire tels que $\nu(\mu(\psi))$ et $\nu(\mu(\chi))$ sont vraies. On note $u = C[u_1]_p$ et $v = C[v_1]_p$.

Soit σ définie par $\sigma(l) = u_1$ et $\sigma(r) = v_1$. Par définition d'une étape équationnelle, $\chi \Leftrightarrow \psi \;\wedge\; \sigma(\xi)$ donc comme $\nu(\mu(\chi))$ est vraie, $\nu(\mu(\sigma(\xi)))$ est vraie. Donc

$$(\nu(\mu(\sigma(l))) = \nu(\mu(\sigma(r)))) \in [\![l = r|\xi]\!]_\nu$$

De plus on a bien $\nu(\mu(\sigma(l))) = \nu(\mu(u_1))$ et $\nu(\mu(\sigma(r))) = \nu(\mu(v_1))$. En ajoutant le contexte, on a bien la propriété voulue. $\diamond$

La réciproque de cette proposition est fausse comme le montre l'exemple suivant.

Exemple 37 $a_i|0 \leq i \leq N$ n'est pas convertible en $a_0|0 \leq N$ par l'équation $a_j = a_0|0 < j \leq N$. En effet la seconde condition de filtrage est avec la substitution $\sigma : j \mapsto i$: à la position 1, $0 \leq i \leq N \Rightarrow 0 < i \leq N$. Elle n'est pas valide (elle est fausse pour $i = 0$). Par contre, dans toutes les interprétations, les deux mots sont convertibles. Par exemple, avec $\nu(N) = 2$ nous obtenons :
- $[\![a_j = a_0|0 < j \leq N]\!]_\nu = \{a_1 = a_0; a_2 = a_0\}$
- l'ensemble des interprétations possibles de $a_i|0 \leq i \leq N$ est $\{a_0; a_1; a_2\}$ et tous ses éléments sont convertibles en a_0 avec une ou zéro étape équationnelle.

Nous pouvons remarquer que si un mot n'a aucune interprétation (c'est-à-dire si sa contrainte est insatisfiable), il n'est convertible qu'avec des mots de la même longueur sans interprétation. Par contre, si un mot a au moins une interprétation, il n'est convertible qu'avec un mot ayant au moins une interprétation.

Congruence équationnelle

Définition 38 *Un* système équationnel paramétré *est un ensemble fini d'équations paramétrées. Deux mots paramétrés sont dits convertibles par ce système si et seulement si il existe une séquence finie d'étapes équationnelles entre eux.*

Notons qu'un ensemble fini d'équation ne peut pas être décrit par une seule équation paramétrée dès qu'il contient deux règle dont les membres droits ou gauches sont de longueur différente.

Nous établissons maintenant la symétrie de la relation définie par une étape équationnelle.

Lemme 10 *Si*

$$u|\psi \xleftrightarrow[p]{l=r|\xi} v|\chi \qquad (1)$$

est une étape équationnelle alors

$$v|\chi \xleftrightarrow[p]{r=l|\xi} u|\psi \qquad (2)$$

est une étape équationnelle.

Preuve Supposons que

$$u|\psi \xleftrightarrow[p]{l=r|\xi} v|\chi$$

est une étape équationnelle. Soit σ la substitution définie par $\sigma(l) = u|_{p,|l|}$ et $\sigma(r) = v|_{p,|r|}$. On remarque que $Var(\sigma(\xi)) \setminus Var(\chi) = \emptyset$ puisque

$$Var(\chi) = Var(\psi) \cup Var(\sigma(\xi))$$

Donc la condition de filtrage pour l'étape équationnelle (2) est

$$\chi \Rightarrow \sigma(\xi)$$

Comme (1) est une étape équationnelle, $\chi \Leftrightarrow \psi \wedge \sigma(\xi)$, et donc $\chi \Rightarrow \sigma(\xi)$ est équivalent à

$$\psi \ \wedge \ \sigma(\xi) \Rightarrow \sigma(\xi)$$

qui est clairement valide.

De plus $\chi \ \wedge \ \sigma(\xi) \Leftrightarrow \psi$ est équivalente à $\psi \ \wedge \ \sigma(\xi) \Leftrightarrow \psi$ qui se réduit à $\psi \Rightarrow \sigma(\xi)$. C'est une formule valide d'après la condition de filtrage de (1). $\diamond$

Nous montrons maintenant que la convertibilité est une congruence. Pour cela nous montrons que la relation est stable par mise sous contexte, par renforcement de la contrainte et par instanciation.

Lemme 11 *Si*

$$u|\psi \xleftrightarrow[p]{l=r|\xi} v|\chi$$

est une étape équationnelle alors pour toute formule ϕ,

$$u|\psi \ \wedge \ \phi \xleftrightarrow[p]{l=r|\xi} v|\chi \ \wedge \ \phi$$

est une étape équationnelle.

Preuve Supposons que

$$u|\psi \xleftrightarrow[p]{l=r|\xi} v|\chi$$

est une étape équationnelle et notons σ la substitution de filtrage. Soit ϕ une formule quelconque. Les mots sans leur contrainte n'étant pas changés, il suffit de vérifier que les contraintes permettent bien l'étape de réécriture

$$u|\psi \ \wedge \ \phi \xleftrightarrow[p]{l=r|\xi} v|\chi \ \wedge \ \phi$$

Avec les notations de la définition d'une étape de conversion,

$$\psi \Rightarrow \exists x_1 \cdots \exists x_n \sigma(\xi)$$

est valide. Comme $\psi \ \wedge \ \phi \Rightarrow \psi$ est valide on a bien

$$\psi \ \wedge \ \phi \Rightarrow \exists x_1 \cdots \exists x_n \sigma(\xi)$$

C'est exactement la condition 1)b) de filtrage quitte à renommer les variables x_i. La condition de production 2)b)

$$\chi \ \wedge \ \phi \Leftrightarrow \psi \ \wedge \ \phi \ \wedge \ \sigma(\xi)$$

est clairement valide puisque

$$\chi \Leftrightarrow \psi \ \wedge \ \sigma(\xi)$$

est valide. Nous avons bien montré le lemme. $\diamond$

Lemme 12 *Si*

$$u|\psi \xleftrightarrow[p]{l=r|\xi} v|\chi$$

est une étape équationnelle alors pour tout contexte justement contraint $C[\diamond]_q|\phi$,

$$C[u]_q|\psi \ \wedge \ \phi \xleftrightarrow[p+q]{l=r|\xi} C[v]_q|\chi \ \wedge \ \phi$$

est une étape équationnelle.

Preuve Supposons que

$$u|\psi \xleftrightarrow[p]{l=r|\xi} v|\chi$$

est une étape équationnelle. En appliquant le lemme 11 page précédente, nous obtenons que

$$u|\psi \ \wedge \ \phi \xleftrightarrow[p]{l=r|\xi} v|\chi \ \wedge \ \phi$$

est une étape équationnelle. Ainsi les condition 1)b) et 2)b) sont vérifiées pour l'étape que l'on cherche. Comme u a un facteur de taille $|l|$ à la position p, $C[u]_q$ a un facteur de taille $|l|$ à la position $p+q$. Cela établit la condition 1)a). Le contexte se reporte à droite de la production syntaxiquement, ce qui permet de vérifier trivialement la condition 2)a). $\diamond$

Lemme 13 *Si*

$$u|\psi \xleftrightarrow[p]{l=r|\xi} v|\chi$$

est une étape équationnelle alors pour toute instanciation ρ *telle que* $\rho(u|\psi)$ *et* $\rho(v|\chi)$ *sont justement contraints,*

$$\rho(u|\psi) \xleftrightarrow[p]{l=r|\xi} \rho(v|\chi)$$

est une étape équationnelle

Preuve Comme la validité d'une formule est stable par instanciation, nous pouvons trivialement établir le lemme en considérant la substitution $\sigma\circ\rho$ à la place de la substitution de conversion σ. $\diamond$

Proposition 6 *La convertibilité est une congruence.*

Preuve La preuve est une conséquence immédiate des lemmes et propositions de cette section. $\diamond$

4.2.2 Règles de réécriture

Définition

Nous introduisons maintenant la notion de *règle de réécriture sur les mots paramétrés.*

Définition 39 *Une* règle sur les mots paramétrés *est un couple* $((u, v), \psi)$, *noté* $u \to v|\psi$, *tel que*

- ψ *est une formule de la logique de Presburger et*
- $u|\psi$ *et* $v|\psi$ *sont deux mots tels que* $uv|\psi$ *est réduit.*

u *est appelé* membre gauche *de la règle,* v *est appelé* membre droit *de la règle et* ψ *est appelé* contrainte *de la règle.*

Un système de réécriture sur les mots paramétrés *est défini comme un ensemble fini de règles et est noté :*

$$R = \left\{ \begin{array}{ccc} u_1 \to v_1 & | & \psi_1; \\ & \vdots & \\ u_k \to v_k & | & \psi_k \end{array} \right\}$$

La sémantique des règles est très similaire à celle des équations.

Définition 40 *Pour une interprétation des paramètres fixée* ν, *la* ν-interprétation *d'une règle est définie par*

$$[\![(u \to v)|\psi]\!]_\nu = \bigcup_{\substack{\mu : I \to \mathbb{Z} \\ \nu(\mu(\psi)) \text{ valide}}} \{\nu(\mu(u)) \to \nu(\mu(v))\}$$

La ν*-interprétation d'une règle entre mots simples est donc un ensemble de règles entre des mots sur la* ν*-interprétation de la signature.*

La ν*-interprétation d'un système de réécriture est l'union des* ν*-interprétations de chaque règle du système.*

Ainsi l'interprétation d'une règle paramétrée est exactement un système de réécriture de mots classique (éventuellement infini).

Exemple 41 Dans $\Sigma_2 = \{a_i | 1 \leq i \leq N - 1\}_{\{N\}}$,

$$[\![a_i \to a_j | 0 \leq i < j \leq P]\!]_{P \mapsto 2} = \{a_0 \to a_1; a_0 \to a_2; a_1 \to a_2\}$$

$$[\![a_i a_j \to a_j a_i | i - j \geq 2]\!]_{N \to 5} = \{a_3 a_1 \to a_1 a_3; a_4 a_1 \to a_1 a_4; a_4 a_2 \to a_2 a_4\}$$

En prenant la signature $\Sigma = \{a_i | i \geq 0\}$ sans paramètres, nous pouvons considérer la règle de réécriture

$$a_i \to a_j | i > j \geq 0$$

dont la ν-interprétation est pour tout ν le système de réécriture de mots infini

$$[\![a_i \to a_j | i > j \geq 0]\!]_\nu = \{a_1 \to a_0; a_2 \to a_0; a_2 \to a_1; a_3 \to a_0; \cdots\}$$

Nous allons maintenant définir l'exécution de la réécriture de mots paramétrés.

Réduction

La définition est exactement celle d'une étape équationnelle, mais en choisissant une orientation.

Définition 42 *Le mot paramétré $u|\psi$ se réduit en un mot paramétré $v|\chi$ par la règle $l \to r|\xi$ à la position $p \in \mathbb{N}$ et on note*

$$u|\psi \xrightarrow[p]{l \to r|\xi} v|\chi$$

si et seulement si

$$u|\psi \xleftrightarrow[p]{l = r|\xi} v|\chi$$

Notons qu'une étape de réduction est une étape équationnelle par définition. Ainsi nous pouvons appliquer la proposition 5 page 41 directement pour énoncer :

Proposition 7 *Si*

$$u|\psi \xrightarrow[p]{l \to r|\xi} v|\chi$$

est une étape de réécriture alors pour tout $\nu : \mathcal{N} \to \mathbb{Z}$ et tout $\mu : I \to \mathbb{Z}$ tels que $[\![u|\psi]\!]_\nu(\mu)$ et $[\![v|\chi]\!]_\nu(\mu)$ sont définis,

$$[\![u|\psi]\!]_\nu(\mu) \xrightarrow[p]{[\![l \to r|\xi]\!]_\nu} [\![v|\chi]\!]_\nu(\mu)$$

4.2.3 Exemples de pas de réécriture

Tous les exemples de cette section seront construits sur la signature $\Sigma_2 = \{a_i | 1 \leq i \leq N-1\}_{\{N\}}$. Tous les mots, règles et équations seront justement contraints implicitement. Nous chercherons pour chaque exemple si la réécriture donnée est correcte.

Exemple 43 A-t-on $a_0|N \geq 1 \xrightarrow[0]{a_0 \to a_1|N \geq 1} a_1|N \geq 1$? Cette question peut se reformuler :

$$a_0|N \geq 1 \xrightarrow[0]{a_j \to a_k|N \geq 1 \ \wedge\ j=0 \ \wedge\ k=1} a_1|N \geq 1$$

Nous définissons la substitution σ par :

$$\sigma(j) = 0$$
$$\sigma(k) = 1$$

1. Filtrage :
   (a) En utilisant les mêmes notations que dans la définition, $u = a_0$ et $l = a_0$ donc u a bien un facteur de longueur 1 à la position 0,
   (b) σ convient bien,

   (c) $\psi = N \geq 1$ donc il faut que $N \geq 1 \Rightarrow \sigma(N \geq 1 \ \wedge \ j = 0 \ \wedge \ k = 1)$ soit valide. C'est-à-dire
$$N \geq 1 \Rightarrow N \geq 1 \ \wedge \ 0 = 0 \ \wedge \ 1 = 1$$
   qui est bien valide.

2. Production :
   (a) $v = a_{\sigma(k)}$
   (b) la contrainte de v est bien équivalente à $N \geq 1 \ \wedge \ \sigma(k = 1)$

La réécriture est donc correcte. Notons que les contraintes $N \geq 1$ sur la règle et le mot réécrit $a_1|N \geq 1$ sont *indispensables* pour assurer que tout est justement contraint. Ce n'est que pour alléger les exemples qu'on ne les écrit pas. Si l'on omettait la contrainte du mot simple $a_0|N \geq 1$, la condition 1.b) ne serait plus valide. En effet si l'on choisit la ν-interprétation $\nu(N) = 0$, la règle $a_0 \to a_1|N \geq 1$ n'a pas d'interprétation. Dans ce cas la ν-interprétation de $a_0|N \geq 1$ n'est jamais définie, alors que celle de a_0 (i.e $a_0|\top$) est définie partiellement. Ce type de comportement non uniforme suivant la valeur des paramètres est évité par la définition.

L'exemple précédent met bien en évidence que la réductibilité d'un mot paramétré ($a_0|N \geq 1$) implique la réductibilité des mots de sa ν-interprétation. Ici $a_0|N \geq 1$ est réductible et tous les mots de sa ν-interprétation sont réductibles. Par contre la non-réductibilité n'est pas stable par ν-interprétation : $a_0|N \geq 0$ est irréductible, mais pour $N = 1$ sa ν-interprétation est toujours a_0 qui est un mot réductible.

Exemple 44

$$a_{i_1}a_{i_2}a_{i_3}|i_1 = i_2 \xrightarrow[2]{a_i a_j \to a_k|k=i+j} a_{i_1}a_{j'}|i_1 = i_2 \ \wedge \ i_2 + i_3 = j'$$

est une étape de réécriture. Il suffit de poser

$$\begin{array}{ll} \sigma(i) & = i_2 \\ \sigma(j) & = i_3 \\ \sigma(k) & = j' \end{array}$$

et de vérifier que
- $i_1 = i_2 \Rightarrow \exists j' \ \sigma(k = i + j)$ et
- $(i_1 = i_2 \ \wedge \ i_2 + i_3 = j') \Leftrightarrow i_1 = i_2 \ \wedge \ \sigma(k = i + j)$

sont valides.

4.2.4 Algorithme de filtrage

L'algorithme de filtrage permet de décider si une règle paramétrée peut s'appliquer sur un mot paramétré réduit donné.

Algorithme 1 (Filtrage)

Entrée : $u|\psi$ *et* $a_{i_1} \dots a_{i_k} \to r|\xi$

Résultat : $(\sigma : I \to Expr(I, N), p \in \mathbb{N})$ *ou* $\perp$

Algorithme : *Soit* S *l'ensemble des couples* (v, p) *où* $v|\psi$ *est un facteur de longueur* $|l|$ *à la position* p *de* $u|\psi$

- *Si* $S = \emptyset$ *renvoyer* $\perp$
- *Soit* $(a_{e_1} \dots a_{e_l}, p) \in S$
  1. *Soit* σ *la substitution telle que* $\sigma(i_1) = u_1, \dots, \sigma(i_k) = u_k$
  2. *Posons* $\{x_1; \dots; x_r\} = Var(\sigma(\xi)) \setminus Var(\psi)$. *Si* $\psi \Rightarrow \exists x_1 \dots x_r \sigma(\xi)$ *est valide alors on renvoie* (σ, p) *sinon on recommence avec* $S \setminus \{(a_{e_1} \dots a_{e_l}, p)\}$.

Exemple 45 Filtrage de $u = (a_i a_j a_k | i + j + k \leq P)$ par la règle $a_f a_g \to r | g - f \leq P$. On pose $\sigma(f) = i$ et $\sigma(g) = j$. Toutes les variables de l sont instanciées, on n'a donc pas de variables existentielles à ajouter. Comme $i + j + k \leq P \Rightarrow j - i \leq P$ est valide, le σ construit convient.

Les variables existentielles permettent de traiter le cas des règles qui introduisent des variables dans leur membre droit.

Exemple 46 $u = (a_i | i \leq P)$ est-il filtré par $a_j \to a_k | j \leq k \leq P$? L'algorithme nous donne $\sigma(j) = i$ et il faut vérifier si $i \leq P \Rightarrow \exists k | i \leq k \leq P$. C'est bien une formule valide. Donc u est bien filtré par la règle.

Proposition 8 *Un mot* $u|\psi$ *est réductible par une règle* $l \to r|\xi$ *si et seulement si l'algorithme de filtrage de* $u|\psi$ *par* $l \to r|\xi$ *calcule une substitution.*

Preuve Il suffit d'appliquer la définition de la réductibilité. $\diamond$

4.3 Propriétés fondamentales

Dans cette section nous allons étudier les différentes propriétés élémentaires de la réécriture paramétrée avec indices ainsi que leur interprétation en réécriture de mots classique.

4.3.1 Relèvement

Nous allons étudier une propriété technique qui permet de simuler les étapes de réécriture de l'interprétation au niveau paramétré de façon exacte. Cette propriété, dite de *relèvement*, sera cruciale dans les preuves qui concernent la transmission d'une propriété paramétrée vers les interprétations comme dans les sections 4.3.3 page 54 et 4.3.4 page 60.

Lemme 14 *Pour toute instanciation des paramètres* $\nu : \mathcal{N} \to \mathbb{Z}$ *et tout mot* $u \in [\![\Sigma]\!]^*_\nu$, *il existe un mot paramétré justement contraint* $\bar{u}$ *de* Σ^* *tel que*

- *pour tout* $\mu : I \to Expr(\mathcal{N}, \mathbb{Z})$, $[\![\bar{u}]\!]_\nu(\mu)$ *est défini et* $[\![\bar{u}]\!]_\nu(\mu) = u$ *et*

– *pour toute instanciation $\nu' : \mathcal{N} \to \mathbb{Z}$, s'il existe $\mu : I \to Expr(\mathcal{N}, \mathbb{Z})$ telle que $[\![\bar{u}]\!]_\nu(\mu)$ est définie alors $\nu' = \nu$.*

Nous dirons que $\bar{u}$ est un ν-relèvement *de u.*

Preuve En appliquant le lemme 3 page 30, nous nous ramenons à une signature ne contenant qu'une variable libre. Nous noterons $\varphi(x)$ la contrainte de la signature dans laquelle cette variable est substituée par x. Notons les paramètres $\mathcal{N} = \{N_1; \cdots; N_r\}$.

Pour tout mot $u = a_{j_1} \ldots a_{j_k}$ où les j sont des entiers, on construit

$$\bar{u} = a_{j_1} \ldots a_{j_k} | \bigwedge_{1 \leq k \leq r} N_k = \nu(N_k)$$

Soit $\mu : I \to Expr(\mathcal{N}, \mathbb{Z})$,

$$\nu(\mu(\ \wedge_{\ 1 \leq k \leq r} N_k = \nu(N_k))) = (\ \wedge_{\ 1 \leq k \leq r} \nu(N_k) = \nu(N_k)) = \top$$

puisque les N_k sont des paramètres et pas des variables et que les $\nu(N_k)$ sont des entiers. Ainsi $[\![\bar{u}]\!]_\nu(\mu)$ est définie et a pour valeur

$$[\![\bar{u}]\!]_\nu(\mu) = a_{\nu(\mu(j_1))} \ldots a_{\nu(\mu(j_k))} = a_{j_1} \ldots a_{j_k}$$

Nous avons bien établi la première condition. Fixons maintenant une interprétation des paramètres $\nu' : \mathcal{N} \to \mathbb{Z}$ et une substitution des variables $\mu : I \to Expr(\mathcal{N}, \mathbb{Z})$. $[\![\bar{u}]\!]_{\nu'}(\mu)$ est définie si et seulement si

$$\nu'(\mu(\bigwedge_{1 \leq k \leq r} N_k = \nu(N_k))) = \top$$

C'est-à-dire

$$(\bigwedge_{1 \leq k \leq r} \nu'(N_k) = \nu(N_k)) = \top$$

On en déduit que $\nu = \nu'$ indépendamment de μ. $\diamond$

Exemple 47 Soit $\Sigma_2 = \{a_i | 1 \leq i \leq N - 1\}_N$ un signature paramétrée. Dans $[\![\Sigma_2]\!]_{N \mapsto 3} = \{a_1; a_2\}$, le mot a_2 a pour relèvement

$$\overline{a_2} = (a_2 | N = 3)$$

Intuitivement, on peut penser qu'un relèvement devrait plutôt être de la forme $a_i | i = 2$. Ce n'est pas possible de prendre cette définition pour le relèvement, car sinon le lemme suivant, qui permet le relèvement des étapes de réécriture, n'est plus correct. En effet, $a_i | i = 2$ a strictement plus d'interprétations que $a_2 | N = 3$ et ces différentes interprétations ne sont pas nécessairement toutes réductibles même si a_2 est réductible pour $N = 3$. Notre définition permet donc de relever de la façon la moins générale possible un mot en un mot paramétré.

Lemme 15 *Soit $l \to r|\xi$ une règle de réécriture paramétrée sur Σ. Soit ν une interprétation des paramètres. Soient u et v deux mots de $[\![\Sigma]\!]_\nu^*$. Si $u \xrightarrow[p]{[\![l\to r|\xi]\!]_\nu} v$ alors il existe $\bar{u}$ un ν-relèvement de u et $\bar{v}$ un ν-relèvement de v tels que*

$$\bar{u} \xrightarrow[p]{l\to r|\xi} \bar{v}$$

.

Preuve Donnons quelques notations :
- Soit $\nu : \{N_1; \cdots; N_r\} \to \mathbb{Z}$,
- $l = a_{i_1} \dots a_{i_q}$
- $r = a_{j_1} \dots a_{j_{q'}}$
- $u = C[a_{n_1} \dots a_{n_q}]_p$ avec $n_j \in \mathbb{Z}$ et $C[.]_p$ un contexte clos,
- $\bar{u} = C[a_{n_1} \dots a_{n_q}]_p \mid \wedge_{1\leq k\leq r} N_k = \nu(N_k)$ un ν-relèvement de u
- $\mu : I \to Expr(\mathcal{N}, \mathbb{Z})$ telle que

  $$(a_{n_1} \dots a_{n_q} \to a_{m_1} \dots a_{m_{q'}}) = \nu(\mu(l)) \to \nu(\mu(r))$$

  et

  $$\nu(\mu(\xi)) = \top$$
- $v = C[a_{m_1} \dots a_{m_{q'}}]_p$ avec $m_j \in \mathbb{Z}$
- $\bar{v} = C[a_{m_1} \dots a_{m_q}]_p| \ \wedge_{1\leq k\leq r} N_k = \nu(N_k) \ \wedge \ \mu(\xi)$. $\bar{v}$ est bien un ν-relèvement de v puisque la formule $\wedge_{1\leq k\leq r} N_k = \nu(N_k) \Rightarrow \mu(\xi)$, dont les seuls symboles non instanciés sont les paramètres, est valide.

Montrons que $\bar{u} \xrightarrow[p]{l\to r|\xi} \bar{v}$.

1. (a) $\bar{u}$ a un facteur de longueur $|l|$ à la position p, car l'interprétation préserve la longueur des mots, et que u a un facteur de longueur $|l|$ à la position $|p|$.
   (b) Notons $\sigma : I \to Expr(\mathcal{N}, \mathbb{Z})$ la substitution définie par
   - $\sigma(i_1) = n_1$
   - $\vdots$
   - $\sigma(i_q) = n_q$
   - $\sigma(j_1) = m_1$
   - $\vdots$
   - $\sigma(j_{q'}) = m_{q'}$

   (c) Il suffit de montrer que

   $$(\wedge_{1\leq k\leq r} N_k = \nu(N_k)) \Rightarrow \sigma(\xi)$$

   est valide puisque toutes les variables sont instanciées de façon close. Soit $\nu' : \mathcal{N} \to \mathbb{Z}$. Si $\nu' \neq \nu$ alors la prémisse de l'implication est fausse et on peut conclure. Sinon $\nu' = \nu$, et il faut montrer que la formule close $\nu(\sigma(\xi))$ est vraie.

   $$\nu(\sigma(\xi)) = \mu(\nu(\sigma(\xi)))$$

car $\nu(\sigma(\xi))$ est close.

$$\mu(\nu(\sigma(\xi))) = \nu(\mu(\sigma(\xi)))$$

car μ ne porte pas sur les paramètres.

$$\nu(\mu(\sigma(\xi))) = \sigma(\nu(\mu(\xi)))$$

car σ et $\nu \circ \mu$ coïncident sur le support de σ (les $i_1, \ldots, i_p, j_1, \ldots j_{q'}$) et donc commutent. Or $\nu(\mu(\xi))$ est vraie, donc l'implication est vraie.

2. (a) Clairement

$$\bar{v} = C[\sigma(a_{j_1} \ldots a_{j_{p'}})]_p$$

(b) La dernière condition est

$$(\bigwedge_{1\leq k\leq r} N_k = \nu(N_k) \ \wedge \ \sigma(\xi)) \Leftrightarrow (\wedge_{\ 1\leq k\leq r} N_k = \nu(N_k) \ \wedge \ \sigma(\xi))$$

C'est immédiat.

◇

Exemple 48 Soit

$$R = \{a_{i+1} \rightarrow a_i \mid N-2 \geq i \geq 0\}$$

on peut faire un $\{N \mapsto 8\}$-relèvement de $a_7 \rightarrow a_6$ en

$$(a_7|N=8) \rightarrow (a_6|N=8 \ \wedge \ 0 \leq 6 \leq N-2)$$

4.3.2 Interprétation des preuves paramétrées - Sémantique

Une conséquence fondamentale du lemme de relèvement est que nous pouvons "relever" toute preuve dans une interprétation en une preuve paramétrée. Soit E un système équationnel paramétré avec indices. Notons $\leftrightarrow_E^*$ la clôture transitive de la relation de convertibilité paramétrée. L'ensemble des théorèmes de E est défini par $Th(E) = \{(u,v) \ / \ u \leftrightarrow_E^* v\}$. Pour les systèmes non paramétrés, cette notion coïncide avec celle de théorie équationnelle.

Théorème 2 *Soit E un système équationnel paramétré avec indices et $E' = Th(E)$ et ν une interprétation des paramètres. Le diagramme*

$$\begin{array}{ccc} E & \stackrel{Th}{\longrightarrow} & E' \\ \nu\downarrow & & \downarrow\nu \\ [\![E]\!]_\nu & \stackrel{Th}{\longrightarrow} & [\![E']\!]_\nu \end{array}$$

commute.

Preuve Nous procédons en deux étapes :

- Soit $(u, v) \in [\![Th(E)]\!]_\nu$, en appliquant le lemme 5 page 41 nous interprétons chaque étape de la preuve dans E en une étape de preuve dans $[\![E]\!]_\nu$. Nous avons donc $[\![Th(E)]\!]_\nu \subset Th([\![E]\!]_\nu)$.
- Réciproquement soit $(u, v) \in Th([\![E]\!]_\nu)$, nous appliquons le lemme 15 page 50 pour relever les étapes de preuve dans $[\![E]\!]_\nu$ en des étapes de preuves dans E. En appliquant le lemme 14, il est clair que l'interprétation de cette preuve est identique à la preuve de départ dans $[\![E]\!]_\nu$ et que c'est donc bien une preuve dans l'interprétation. Nous avons bien $Th([\![E]\!]_\nu) \subset [\![Th(E)]\!]_\nu$.

$\diamond$

Ce théorème fondamental montre que les théorèmes que nous pouvons montrer au niveau paramétré, s'ils ont une interprétation, s'interprètent comme des théorèmes au niveau interprété. Réciproquement, tout théorème au niveau interprété est l'interprétation d'un théorème au niveau paramétré.

Nous définissons maintenant les modèles de la réécriture paramétrée.

Définition 49 *Soit Σ une signature paramétrée. Une* interprétation *de Σ est une paire (A, ν) où ν est une interprétation des paramètres et A est une $[\![\Sigma]\!]_\nu$-algèbre. Nous dirons que (A, ν) est un modèle de l'équation paramétrée e si A est un modèle de l'ensemble d'équations $[\![e]\!]_\nu$. Dans ce cas, nous noterons $(A, \nu) \models e$. Si E est un système équationnel paramétré nous noterons $\mathcal{A}_E = \{(A, \nu) \ / \ \forall e \in E \quad (A, \nu) \models e\}$ l'ensemble des modèles de toutes les équations de E.*

Cette définition peut aussi s'exprimer par «$(A, \nu) \models e$ si et seulement si $A \models [\![e]\!]_\nu$».

Définition 50 *Soient E un système équationnel paramétré et e une équation paramétrée. Nous dirons que e est une conséquence logique de E, et nous noterons $E \models e$, si tout modèle $(A, \nu) \in \mathcal{A}_E$, est un modèle de e, c'est-à-dire si pour tout f dans E, $(A, \nu) \models f$ alors $(A, \nu) \models e$.*

Nous définissons maintenant la notion de dérivabilité des équations.

Définition 51 *Soient E un système équationnel paramétré et e une équation paramétrée. L'équation e est* dérivable *par E, et l'on note $E \vdash e$, si pour tout ν, interprétation des paramètres, il existe un système équationel paramétré E' tel que $[\![E]\!]_\nu \vdash [\![E']\!]_\nu$ et $[\![E']\!]_\nu = [\![e]\!]_\nu$.*

Notons bien que dans cette définition $[\![E]\!]_\nu \vdash [\![E']\!]_\nu$ est la dérivabilité au sens classique dans les $[\![\Sigma]\!]_\nu$-algèbres de mot sans paramètres ni exposants.

Remarquons que si deux mots sont convertibles au niveau paramétré, l'équation composée de ces deux mots est dérivable au sens de la définition précédente. Par contre la réciproque est fausse comme nous le verrons dans l'exemple suivant. Ce phénomène s'observe aussi dans la réécriture contrainte [18].

Théorème 3 *Soient E un système équationnel paramétré et e une équation paramétrée.*

$$E \vdash e \text{ ssi } E \models e$$

Preuve

1. Correction : si $E \vdash e$ alors $E \models e$. Nous suppposons que $E \vdash e$. Fixons ν une interprétation des paramètres. En appliquant la définition de $E \vdash e$ nous avons $[\![E]\!]_\nu \vdash [\![e]\!]_\nu$. Nous appliquons le sens direct du théorème de Birkhoff [1] et obtenons que $[\![E]\!]_\nu \models [\![e]\!]_\nu$. Nous en déduisons $E \models e$.
2. Complétude : si $E \models e$ alors $E \vdash e$. Supposons que $E \models e$. Soit ν une interprétation des paramètres. Par définition de $E \models e$, $[\![E]\!]_\nu \models [\![e]\!]_\nu$. Le théorème de Birkhoff permet d'affirmer que $[\![E]\!]_\nu \vdash [\![e]\!]_\nu$. Définissons

$$E' = \overline{[\![e]\!]_\nu} = \bigcup_{e' \in [\![e]\!]_\nu} \overline{e'}$$

le système équationnel paramétré composé de l'union des ν-relèvements des équations de $[\![e]\!]_\nu$. En appliquant le lemme 15 page 50 de relèvement des preuves paramétrées sur chaque équation de $[\![e]\!]_\nu$, nous obtenons $E \vdash \overline{[\![e]\!]_\nu}$, c'est-à-dire $E \vdash E'$. De plus par définition du ν-relèvement, $[\![E']\!]_\nu = [\![e]\!]_\nu$. Nous en déduisons $E \vdash e$.

◇

Remarquons que la convertibilité permet de prouver strictement moins de théorèmes que le système de déduction $\vdash$. En effet la convertibilité ne donne que des preuves *uniformes*, c'est-à-dire des preuves qui s'interprètent en des preuves qui sont toutes instances d'une même preuve paramétrée. En particulier toutes ces preuves sont de la même longueur.

Donnons deux exemples de l'application de ce théorème.

1. Considérons un cas sans paramètre. Nous sommes donc dans le même cadre que la réécriture contrainte. Nous notons $\Sigma = \{a_i | i \geq 0\}$, $E = \{a_{i+1} = a_i | i \geq 0\}$ et $e = \ a_i = a_0 | i \geq 0$. Explicitons la preuve de la complétude dans ce cas. Nous avons bien $\{a_{i+1} = a_i | i \geq 0\} \models a_i = a_0 | i \geq 0$. Comme il n'y a pas de paramètres, nous prennons un ν de support vide. Nous obtenons

$$\bigcup_{i \in \mathbb{N}} \{a_{i+1} = a_i\} \models \bigcup_{i \in \mathbb{N}} \{a_i = a_0\}$$

D'après le théorème de Birkhoff nous avons donc

$$\bigcup_{i \in \mathbb{N}} \{a_{i+1} = a_i\} \vdash \bigcup_{i \in \mathbb{N}} \{a_i = a_0\}$$

Posons $E' = \bigcup_{i \in \mathbb{N}} \{a_i = a_0 | \top\}$. C'est un système équationnel paramétré infini. Nous avons bien $E \vdash E'$ en appliquant le théorème de relèvement sur la chaque preuve de la forme

$$a_i \leftrightarrow_{[\![E]\!]_\nu} \leftrightarrow_{[\![E]\!]_\nu} a_{i-1} \leftrightarrow_{[\![E]\!]_\nu} \cdots \leftrightarrow_{[\![E]\!]_\nu} a_0$$

pour tout $i \in \mathbb{N}$. Chaque preuve est de longueur i. De plus nous avons aussi $[\![E']\!]_\nu = [\![e]\!]_\nu$. Ce qui permet de conclure que $E \vdash e$. Notons bien que les preuves ne sont pas de longueur constante. Nous ne pouvons donc pas espérer les construire comme instance d'une suite d'étapes équationnelles paramétrées. Dans ce cas la forme normale uniforme de $a_i|i \geq 0$ est a_0 mais n'est pas atteignable par une preuve équationnelle. Ceci montre que si l'on dispose d'un oracle sachant donner une forme normale uniforme, nous améliorons l'approximation de la prouvabilité définie par la congruence équationnelle en cherchant des preuves dont la longueur n'est plus constante.

2. Considérons maintenant $\mathcal{N} = \{N\}$ et $\Sigma = \{a_i|N \geq i \geq 0\}$, $E = \{a_N = a_i|N \geq i \geq 0\}$ et $e = \ a_i = a_0|N \geq i \geq 0$. Fixons ν une interprétation pour N dans $\mathbb{N}$. Clairement $\{a_{\nu(N)} = a_i|\nu(N) \geq i \geq 0\} \models \{a_i = a_0|\nu(N) \geq i \geq 0\}$ avec le sens classique de la validité dans une algèbre. Posons $E' = \bigcup_{0 \leq i \leq \nu(N)} \{a_i = a_0|N = \nu(N)\}$. C'est un système équationnel paramétré qui dépend de ν. Nous avons clairement $E \vdash E'$ et $[\![E']\!]_\nu = [\![e]\!]_\nu$, ce qui permet de conclure que $E \vdash e$. Dans cet exemple on constate que les preuves dans E' dépendent de la valeur du paramètre N. Ce ne sont pas ces preuves que l'on trouve en utilisant la congruence équationnelle. Le preuves que l'on peut construire sont uniformes en N.

Ces deux exemples illustrent l'approximation que nous faisont en ne considérant que les preuves uniformes.

4.3.3 Confluence locale

Définition et sémantique

Définition 52 *Un système paramétré est dit* localement confluent *si pour tout $u|\phi$ tel que ϕ est satisfiable, et tous $v_1|\psi_1$ et $v_2|\psi_2$ tels que*

- $u|\phi \xrightarrow{R} v_1|\psi_1$,
- $u|\phi \xrightarrow{R} v_2|\psi_2$ *et*
- $Var(v_1|\psi_1) \cap Var(v_2|\psi_2) \subset Var(u|\phi)$

il existe deux mots $v_1'|\psi_1'$ et $v_2'|\psi_2'$ tels que

- $Var(v_1'|\psi_1') \cap Var(v_2'|\psi_2') \subset Var(u|\phi)$
- $v_1|\psi_1 \xrightarrow{R}{}^* v_1'|\psi_1'$,
- $v_2|\psi_2 \xrightarrow{R}{}^* v_2'|\psi_2'$ *et*
- $\psi_1' \ \wedge \ \psi_2' \Rightarrow (v_1' \equiv v_2') \ \wedge \ \psi_2' \ \wedge \ \psi_1'$ *est valide.*

Les mots $v_1|\psi_1$ et $v_2|\psi_2$ sont dits joignables.

Imposer la satisfiabilité de la contrainte du mot au sommet d'un pic permet d'éliminer les pics qui n'ont pas d'interprétation. Notons que cette condition n'est pas nécessaire pour le lemme qui suit puisque de toute façon de tels pics ne donnent pas lieu à des pics dans l'interprétation. Cela permet d'augmenter la classe des systèmes qui seront localement confluents au niveau paramétré.

Notons bien que la joignabilité de deux mots n'est pas équivalente à l'existence de réduits localement similaires. Par exemple les mots $a_j|j = 1$ et $a_{j'}|j' = 1$ ne sont pas localement similaires, mais sont joignables en 0 pas.

Lemme 16 *Soit R un système paramétré localement confluent. Toute ν-interprétation de R est localement confluente.*

Preuve Donnons nous une interprétation ν des paramètres . Si u est un mot classique de $[\![\Sigma]\!]_\nu^*$ qui se réécrit en u' et u'' avec le système $[\![R]\!]_\nu$, le lemme 15 page 50 permet de dire que $\overline{u}$ se réécrit en $\overline{u'}$ et en $\overline{u''}$ tels que $Var(\overline{u'}) \cap Var(\overline{u''}) \subset Var(\overline{u})$. Notons ϕ la contrainte de $\overline{u'}$ qui est ausi la contrainte de $\overline{u''}$ d'après la construction du relèvement.

Par confluence locale du système paramétré, il existe deux mots paramétrés $v'|\psi'$ et $v''|\psi''$, réduits respectifs de $\overline{u'}$ et de $\overline{u''}$ tels que

$$\phi \Rightarrow (v' \equiv v'') \ \wedge \ \psi' \ \wedge \ \psi''$$

est valide.

Comme $\nu(\phi) = \top$ par définition du relèvement, il existe une interprétation des variables μ et un mot v_0 tels que $v_0 = [\![v']\!]_\nu(\mu)$ et $v_0 = [\![v'']\!]_\nu(\mu)$. Comme $[\![\overline{u}]\!]_\nu(\mu) = u$, $[\![\overline{u'}]\!]_\nu(\mu) = u'$ et $[\![\overline{u''}]\!]_\nu = u''$, en appliquant la propostion 5 page 41, on a établi que v_0 est un réduit commun de u' et u''. L'interprétation du système est donc localement confluente. $\diamond$

Ce lemme justifie l'intérêt de la réécriture paramétrée.

Sa réciproque est fausse comme le prouve l'exemple suivant.

Exemple 53 Soit $R = \{a_{i+1} \to a_i | 1 < i < P; a_P \to a_0\}$. Ce système est équivalent à $\{a_i \to a_j | i = j+1 \ \wedge \ 1 < j < P; a_k \to a_l | k = P \ \wedge \ l = 0\}$ Toutes ses interprétations sont évidemment localement confluentes. Mais le mot $a_i|i = P$ admet les deux réécritures suivantes :

- $(a_i|i = P) \xrightarrow[0]{a_k \to a_l | k=P \ \wedge \ l=0} (a_l | i = P \ \wedge \ i = P \ \wedge \ l = 0)$
- $(a_i|i = P) \xrightarrow[0]{a_i \to a_j | i=j+1 \ \wedge \ 1<j<P} (a_j | i = P \ \wedge \ i = j+1 \ \wedge \ 1 < j < P)$

Ces deux mots sont en forme normale pour le système paramétré, mais ils ne vérifient pas la condition de validité de

$$(i = P \ \wedge \ i = P \ \wedge \ l = 0) \ \wedge \ (i = P \ \wedge \ i = j+1 \ \wedge \ 1 < j < P) \Rightarrow l = j$$

Notion de paire critique

Nous définissons d'abord une notion technique qui permet de décomposer un mot en deux ou trois facteurs.

Définition 54 *Soit $u|\phi$ un mot paramétré avec indices. Nous notons $Dec(u|\phi)$ l'ensemble des découpages du mot $u|\phi$ en un préfixe, un facteur et un suffixe :*

$$Dec(a_1 \dots a_k|\phi) = \{(a_1 \dots a_{k'}, a_{k'+1} \dots a_{k''}, a_{k''+1} \dots a_k, \phi) \ / \ 0 \le k' \le k'' \le k\}$$

Nous notons $Split(u|\phi)$ *l'ensemble des découpages en un préfixe et un suffixe :*

$$Split(u|\phi) = \{(p, s, \phi) \ / \ (p, \epsilon, s, \phi) \in Dec(u|\phi)\}$$

Nous notons aussi l'ensemble des préfixes d'un mot par

$$Pref(u|\phi) = \{(p, \phi) \ / \ \exists s(p, s, \phi) \in Split(u|\phi)\}$$

et l'ensemble des ses suffixes par

$$Suff(u|\phi) = \{(s, \phi) \ / \ \exists p(p, s, \phi) \in Split(u|\phi)\}$$

L'ensemble des contextes du mot est

$$Cont(u|\phi) = \{(p, s, \phi) \ / \ \exists q(p, q, s, \phi) \in Dec(u|\phi)\}$$

L'ensemble des facteurs est

$$Sub(u|\phi) = \{(q, \phi) \ / \ \exists p \exists s(p, q, s, \phi) \in Dec(u|\phi)\}$$

Notons que $Split(u|\phi)$ contient (ϵ, u, ϕ) et (u, ϵ, ϕ).

Exemple 55

$$Split(a_i a_j | i + j = N) = \{(\epsilon, a_i a_j, i + j = N); (a_i, a_j, i + j = N); (a_i a_j, \epsilon, i + j = N)\}$$

Définition 56 *Soient* $l_1 \longrightarrow r_1|\psi_1$ *et* $l_2 \longrightarrow r_2|\psi_2$ *deux règles réduites paramétrées sans variables communes. Définissons les quatre ensembles :*

$$PC_1 = \{ \quad (c_1 r_1, r_2 c_2, \psi_1 \ \wedge \ \psi_2 \ \wedge \ (c_1 l_1 \equiv l_2 c_2)) \quad / \quad \begin{array}{l} (c_1, \psi_2) \in Pref(l_2|\psi_2) \\ (c_2, \psi_1) \in Suff(l_1|\psi_1) \end{array} \}$$

$$PC_2 = \{ \quad (r_1 c_1, c_2 r_2, \psi_1 \ \wedge \ \psi_2 \ \wedge \ (l_1 c_1 \equiv c_2 l_2)) \quad / \quad \begin{array}{l} (c_1, \psi_2) \in Suff(l_2|\psi_2) \\ (c_2, \psi_1) \in Pref(l_1|\psi_1) \end{array} \}$$

$$PC_3 = \{ \quad (c_1 r_1 c_1', r_2, \psi_1 \ \wedge \ \psi_2 \ \wedge \ (c_1 l_1 c_1' \equiv l_2)) \quad / \ (c_1, c_1', \psi_2) \in Cont(l_2|\psi_2)\}$$

$$PC_4 = \{ \quad (r_1, c_2 r_2 c_2', \psi_1 \ \wedge \ \psi_2 \ \wedge \ (l_1 \equiv c_2 l_2 c_2')) \quad / \ (c_2, c_2', \psi_1) \in Cont(l_1|\psi_1)\}$$

Nous appelons ensemble des paires critiques *entre les deux règles l'ensemble*

$$PC = \{(u, v, \chi) \in PC_1 \cup PC_2 \cup PC_3 \cup PC_4 \ / \ \chi \ est\ satisfiable\}$$

Un élément de PC est appelé paire critique. Une telle paire critique (u, v, χ) *est dite* confluente *si le mot* $u|\chi$ *et le mot* $v|\chi$ *sont joignables.*

Ces deux mots sont appelés mots critiques associés à la paire critique. Une paire critique confluente sera parfois aussi dite, conformément à l'usage en réécriture classique, joignable.

Nous remarquons que la paire (u, v, χ) est joignable si et seulement si les mots $u|\chi$ et $v|\chi$ sont localement similaires puisque la contrainte est la même sur les deux mots.

Notons aussi que nous avons en fait énuméré finiment toutes les superpositions, possibles entre deux membres gauches de règles. Cette énumération n'est pas minimale puisque de nombreuses paires critiques se retrouvent dans plusieurs ensembles. De nombreux éléments des ensembles PC_i ont une contrainte insatisfiable pour des raisons de longueur puisque $u \equiv v$ est $\perp$ dès que u et v sont de longueur différente et ne sont donc pas présents dans PC.

Exemple 57 Définissons la signature $\Sigma_2 = \{a_i | 1 \leq i \leq N-1\}_{\{N\}}$ et le système

$$R = \{a_i a_j \to a_k | i + j = k \ \wedge \ 1 \leq i, j, k \leq N-1\}$$

Donnons une paire critique de la règle de ce système avec un de ses renommages

$$a_{i'} a_{j'} \to a_{k'} | i' + j' = k' \ \wedge \ 1 \leq i', j', k' \leq N-1$$

Dans PC_1, par exemple, avec $c_1 = \epsilon$ et $c_2 = \epsilon$, nous avons la paire critique

$$\begin{array}{lll} (a_k, \quad a_{k'}, & & (a_i a_j \equiv a_{i'} a_{j'}) \\ & \wedge & (i + j = k \ \wedge \ 1 \leq i, j, k \leq N-1) \\ & \wedge & (i' + j' = k' \ \wedge \ 1 \leq i', j', k' \leq N-1) \\ & \wedge & \top \ \wedge \ \top) \end{array}$$

c'est-à-dire en simplifiant

$$\begin{array}{lll} (a_k, \quad a_{k'}, & & i = i' \ \wedge \ j = j' \\ & \wedge & i + j = k \ \wedge \ i' + j' = k' \\ & \wedge & 1 \leq i, j, k, i', j', k' \leq N-1) \end{array}$$

La contrainte est satisfiable et cette paire est confluente puisque les deux mots critiques sont localement similaires d'après le lemme 8 page 39.

Lemme 17 *Un système paramétré est localement confluent si et seulement si toutes ses paires critiques sont confluentes.*

Preuve Il est évident que la confluence locale implique la confluence de toutes les paires critiques.

La preuve de la réciproque est similaire à [20], mais les cas à distinguer sont plus nombreux.

Fixons un mot $u|\phi$ réductible par un système R dont toutes les paires critiques sont confluentes et notons

- $u|\phi \xrightarrow[p_1]{l_1 \to r_1 | \psi_1} v_1 | \chi_1$ avec la substitution σ_1
- $u|\phi \xrightarrow[p_2]{l_2 \to r_2 | \psi_2} v_2 | \chi_2$ avec la substitution σ_2

Quitte à renommer les variables, on suppose que les deux règles appliquées ont des variables distinctes et donc σ_1 et σ_2 sont des substitutions à supports disjoints. De plus, par définition de la confluence locale, les seul pics à considérer vérifient que

$$Var(v_2|\chi_2) \cap Var(v_1|\chi_1) \subset Var(u|\phi)$$

Nous nous ramenons de plus au cas où $Var(u|\phi) = Var(\phi)$, quitte à ajouter des tautologies entre les variables de u qui ne sont pas des variables de ϕ.

Nous pouvons donc noter $\sigma = \sigma_1 \circ \sigma_2 = \sigma_2 \circ \sigma_1$. Nous notons aussi $u = u_1 l'_1 v_1 = u_2 l'_2 v_2$ avec $l'_1 = u|_{p_1,|l_1|}$ et $l'_2 = u|_{p_2,|l_2|}$.

1. Supposons que $(u_1 l'_1, \phi) \in \mathit{Pref}(u_2|\phi)$. Il existe donc w_1 tel que $(u_1, l'_1, w_1, \phi) \in \mathit{Dec}(u_2|\phi)$. Nous sommes dans le cas où les redex sont disjoints. Nous avons

$$u|\phi = u_1 l'_1 w_1 l'_2 v_2|\phi$$

En appliquant la définition de la réécriture, nous avons

(r1) $u|\phi \xrightarrow[p_1]{l_1 \to r_1|\psi_1} u_1\sigma(r_1)w_1 l'_2 v_2|\phi \ \wedge \ \sigma(\psi_1)$

(r2) $u|\phi \xrightarrow[p_2]{l_2 \to r_2|\psi_2} u_1 l'_1 w_1 \sigma(r_2) v_2|\phi \ \wedge \ \sigma(\psi_2)$

En notant $p'_2 = |u_1| + |\sigma_1(r_1)| + |w_1|$ et $C[\diamond]_{p'_2} = u_1\sigma(r_1)w_1 \diamond v_2$, montrons que

$$C[l'_2]_{p'_2}|\phi \ \wedge \ \sigma(\psi_1) \xrightarrow{l_2 \to r_2|\psi_2} C[\sigma(r_2)]_{p'_2}|\phi \ \wedge \ \sigma(\psi_1) \ \wedge \ \sigma(\psi_2)$$

est un pas de réécriture.

(a) $C[l'_2]_{p'_2}$ a un facteur de longueur $|l_2| = |l'_2|$ à la position p'_2.

(b) La formule

$$\phi \ \wedge \ \sigma(\psi_1) \Rightarrow \exists Var(\sigma(\psi_2)) \setminus (Var(\sigma(\psi_1)) \cup Var(\phi)) \ \sigma(\psi_2)$$

est valide, car

$$\phi \Rightarrow \exists Var(\sigma(\psi_2)) \setminus Var(\phi) \ \sigma(\psi_2)$$

est valide. En effet,

$$(Var(\sigma(\psi_2)) \cap Var(\sigma(\psi_1))) \subset (Var(v_2|\chi_2) \cap Var(v_1|\chi_1)) \subset Var(\phi)$$

i. σ vérifie bien la première condition de production.

ii. la formule $(\phi \ \wedge \ \sigma(\psi_1)) \ \wedge \ \sigma(\psi_2)$ est bien équivalente à $\phi \ \wedge \ \sigma(\psi_1) \ \wedge \ \sigma(\psi_2)$.

Nous pouvons de façon parfaitement symétrique démontrer que

$$u_1 l'_1 w_1 \sigma(r_2) v_2|\phi \ \wedge \ \sigma(\psi_2)$$

se réduit aussi en

$$u_1\sigma(r_1)w_1\sigma(r_2)v_2|\phi \ \wedge \ \sigma(\psi_1) \ \wedge \ \sigma(\psi_2)$$

Nous avons bien établi la confluence locale dans ce cas.

2. Supposons que $(u_2 l'_2, \phi) \in Pref(u_1|\phi)$. Ce cas est symétrique par rapport au précédent et s'établit donc de façon similaire. Nous avons fini de traiter les cas où les redex sont disjoints.

3. Supposons $(l'_1, \phi) \in Sub(l'_2|\phi)$. Montrons que dans ce cas, nous avons une instance d'une paire critique de PC_3. Il existe deux mots w_1 et w'_1 tels que $(w_1, l'_1, w'_1) \in Dec(l'_2|\phi)$. Nous avons $(w_1, w'_1) \in Cont(l'_2|\phi)$.

   Comme $\sigma(l_1) = l'_1$ et $\sigma(l_2) = l'_2$, les deux étapes de réécriture suivantes sont correctes :
   (r1) $u|\phi \xrightarrow[p_1]{l_1 \to r_1|\psi_1} v_1 w_1 \sigma(r_1) w'_1 v_2|\phi \ \wedge \ \sigma(\psi_1)$
   (r2) $u|\phi \xrightarrow[p_2]{l_2 \to r_2|\psi_2} v_1 \sigma(r_2) v_2|\phi \ \wedge \ \sigma(\psi_2)$
   Comme $\sigma(l_2) = w_1 \sigma(l_1) w'_1$, il existe c_1 et c'_1 tels que $(c_1, l_1, c_2, \psi_2) \in Dec(l_2|\psi_2)$ et $\sigma(c_1) = w_1$ et $\sigma(c'_1) = w'_1$. Il suffit de prendre $c_1 = l_2|_{1,|w_1|}$ et $c'_1 = l_2|_{|w_1|+|l_1|+1,|w'_1|}$.
   La paire critique

   $$c_1 r_1 c'_1, r_2, \psi_1 \ \wedge \ \psi_2 \ \wedge \ (c_1 l_1 c'_1 \equiv l_2)$$

   est joignable. Le lemme 13 page 44 permet d'instancier cette paire critique et d'affirmer que

   $$\sigma(c_1 r_1 c'_1), \sigma(r_2), \sigma(\psi_1 \ \wedge \ \psi_2 \ \wedge \ (c_1 l_1 c'_1 \equiv l_2))$$

   est joignable, c'est-à-dire que

   $$w_1 r'_1 w'_1, r'_2, \sigma(\psi_1) \ \wedge \ \sigma(\psi_2) \ \wedge \ (w_1 l'_1 w'_1 \equiv l'_2))$$

   est joignable. La contrainte $w_1 l'_1 w'_1 \equiv l'_2$ est toujours vraie, ce qui nous permet de simplifier et d'obtenir que

   $$w_1 r'_1 w'_1, r'_2, \sigma(\psi_1) \ \wedge \ \sigma(\psi_2)$$

   est joignable. En appliquant les lemmes 11 page 43 et 12 page 44 nous avons

   $$v_1 w_1 r'_1 w'_1 v_2, v_1 r'_2 v_2, \sigma(\psi_1) \ \wedge \ \sigma(\psi_2) \ \wedge \ \phi$$

   joignable, ce qui implique immédiatement que les deux mots $v_1 w_1 r'_1 w'_1 v_2|\sigma(\psi_1) \ \wedge \ \phi$ et $v_1 r'_2 v_2|\sigma(\psi_2) \ \wedge \ \phi$ sont joignables.

4. Supposons $(l'_2, \phi) \in Sub(l'_1|\phi)$. De façon symétrique au cas précédent, nous pouvons conclure.

5. Supposons $(u_1, \phi) \in Pref(u'_1 l_2|\phi)$ et $(u'_2, \phi) \in Suff(l'_1 u_2|\phi)$. De façon analogue au cas précédent, nous pouvons conclure.

6. Le dernier cas, $(u'_1, \phi) \in Pref(u_1 l'_1|\phi)$ et $(u_2, \phi) \in Suff(l'_2 u'_2|\phi)$, est symétrique par rapport au précédent.

Nous avons bien couvert tous les cas possibles et nous pouvons conclure. ◇

Exemple 58 Considérons le système suivant composé de la règle $a_i a_j \longrightarrow a_j a_i | i < j$. Montrons qu'il est localement confluent. On calcule les paires critiques de la règle avec elle-même :

$$\begin{array}{rcl} PC & = & \{(a_l a_j a_i, a_{j'} a_{i'} a_{l'}, (a_l a_i a_j = a_{i'} a_{j'} a_{l'}) \ \wedge \ i < j \ \wedge \ i' < j'); \\ & & (a_j a_i, a_{j'} a_{i'}, (a_i a_j = a_{i'} a_{j'}) \ \wedge \ i < j \ \wedge \ i' < j'); \\ & & (a_j a_i a_l, a_{i'} a_{l'} a_{j'}, (a_i a_j a_l = a_{l'} a_{i'} a_{j'}) \ \wedge \ i < j \ \wedge \ i' < j')\} \end{array}$$

Nous pouvons vérifier que ces paires sont joignables.

Exemple 59 Pour le système $a_{i+2} a_{i+1} \rightarrow a_i | i \geq 0$ lié à la présentation des variétés groupoïdes vérifiant $xxyx = y$ décrite dans la thèse de Pedersen [27], les paires critiques sont

$$\begin{array}{rcl} PC & = & \{(a_l a_i, a_{i'} a_{l'}, (a_l a_{i+2} a_{i+1} = a_{i'+2} a_{i'+1} a_{l'}) \ \wedge \ i \geq 0 \ \wedge \ i' \geq 0); \\ & & (a_i, a_{i'}, (a_{i+2} a_{i+1} = a_{i'+2} a_{i'+1}) \ \wedge \ i \geq 0 \ \wedge \ i' \geq 0); \\ & & (a_i a_l, a_{l'} a_{i'}, (a_{i+2} a_{i+1} a_l = a_{l'} a_{i'+2} a_{i'+1}) i \geq 0 \ \wedge \ i' \geq 0)\} \end{array}$$

La deuxième paire est clairement confluente comme dans l'exemple précédent. La première correspond aux deux façons de réécrire $a_{i+3} a_{i+2} a_{i+1}$. Elle n'est pas joignable. Le système paramétré n'est donc pas localement confluent, ce qui signifie qu'au moins une de ses ν-interprétation n'est pas localement confluente.

Nous pouvons conclure cette section par notre premier résultat fondamental.

Théorème 4 *Soit R un système de réécriture paramétré avec indices. Si toutes ses paires critiques sont confluentes alors toutes les ν-interprétations de R sont localement confluentes.*

Preuve C'est une application immédiate des lemmes 16 page 55 et 17 page 57. ◇

4.3.4 Terminaison

Définition 60 *Un système paramétré R termine si et seulement si tout mot paramétré t ne peut être réduit qu'un nombre fini de fois par R.*

Théorème 5 *R termine si et seulement si pour tout ν, $[\![R]\!]_\nu$ termine.*

Preuve Le sens direct se montre par l'absurde : il suffit de considérer une chaîne infinie dans $[\![R]\!]_\nu$ et de la relever avec le lemme 4.3.1 page 48 en une chaîne infinie dans R. Réciproquement, si R ne termine pas alors nous pouvons construire une chaîne infinie de mots paramétrés que nous pouvons interpréter en une chaîne infinie dans l'interprétation. ◇

Pour montrer la terminaison nous pouvons tenter de construire un ordre bien fondé sur les mots paramétrés qui est stable par ν-interprétation. Par exemple, l'ordre induit par la longueur des mots paramétrés convient. En pratique des compositions lexicographique de la longueur et d'un ordre sur expressions en indice sera souvent utilisable. Ainsi la règle $a_{i+1} \rightarrow a_i | i > 0$ peut être orienté par l'ordre défini par : $a_i > a_j$ ssi $i > j$.

4.3.5 Formes normales

Formes normales standard

Définition 61 *Un mot paramétré est dit en forme normale standard pour un système paramétré si aucune règle du système ne peut le réduire.*

Nous pouvons facilement décider si un mot est en forme normale vis-à-vis d'un système de réécriture en lui appliquant l'algorithme de filtrage.

Proposition 9 *Le mot $u|\psi$ est en forme normale standard vis-à-vis du système R si et seulement si il n'est filtré par aucune règle de R.*

Preuve C'est une conséquence immédiate de la proposition 8 page 48. $\diamond$

Par contre un mot paramétré peut être en forme normale sans que ses interprétations le soient. Ainsi a_P est en forme normale pour $R = \{a_{i+1} \to a_i | i < P\}$. L'interprétation $P = 0$ fait que a_0 est en forme normale. Nous avons donc besoin d'une autre notion de normalisation, plus fine, la *normalisation uniforme*.

Formes normales uniformes

Définition 62 *Un mot est dit en* forme normale uniforme *si et seulement si toutes ses ν-interprétations sont en forme normale.*

La forme normale uniforme, si elle existe, peut être différente de la forme normale. Ainsi a_P est en forme normale pour $a_{i+1} \to a_i | 0 \leq i < P$, mais a pour forme normale uniforme a_0.

On sait décider si un mot est en forme normale uniforme grâce au théorème suivant.

Théorème 6 *Un mot $u|\psi$ est en forme normale uniforme pour la règle $l \to r|\xi$ si et seulement si pour toute position p telle que u a un facteur de longueur $|l|$ à la position p i.e.*

$$\begin{array}{ll} u = & a_{e_1} \cdots a_{e_{p-1}} a_{e_p} \cdots a_{e_{p+|l|-1}} \ a_{e_{p+|l|}} \cdots a_{e_m} \\ l = & \qquad\qquad\quad a_{i_1} \cdots a_{i_{|l|}} \end{array}$$

En notant σ la substitution

$$\begin{array}{rcl} \sigma(i_1) & = & e_p \\ & \vdots & \\ \sigma(i_{|l|}) & = & e_{p+|l|-1} \end{array}$$

et $\{x_1; \cdots; x_n\} = Var(\sigma(\xi)) \setminus Var(\psi)$ alors

$$\psi \ \wedge \ \exists x_1 \cdots \exists x_n \ \sigma(\xi)$$

est insatisfiable.

Preuve Soit σ définie par $\sigma(l) = u|_{p,|l|}$. Notons μ une interprétation des paramètres et des variables telle que $\mu(\psi \wedge \exists x_1 \cdots \exists x_n\ \sigma(\xi))$ est vraie. Si $\mu(\psi)$ est fausse alors $\mu(u|\psi)$ n'est pas défini, l'interprétation est donc bien trivialement en forme normale. Sinon $\mu(\psi)$ est vraie. Prolongeons μ sur les x_i telle que $\mu(\sigma(\xi))$ est vraie. Ainsi la règle a une de ses interprétations qui est $\mu(\sigma(l)) \to \mu(\sigma(r))$ et qui réduit bien $\mu(u)$. Réciproquement, si u n'est pas en forme normale uniforme, une de ses instances se réduit. Soit μ l'instanciation qui rend u réductible, $\mu(\psi \wedge \exists x_1 \cdots \exists x_n\ \sigma(\xi))$ est vraie et donc la formule $\psi \wedge \exists x_1 \cdots \exists x_n\ \sigma(\xi)$ est satisfiable. ◇

Par contre, il n'existe pas d'algorithme pour calculer une forme normale uniforme ni pour vérifier qu'un mot est une forme normale uniforme d'un mot donné.

Proposition 10 *Trouver une forme normale uniforme pour un mot donné est indécidable.*

Preuve Idée de la preuve : le nombre d'étapes de réécriture dépend de la valeur du paramètre. Il faut donc une preuve inductive pour calculer la forme normale uniforme. ◇

Une condition nécessaire pour être une forme normale uniforme est donnée par la proposition suivante.

Proposition 11 *Si u est un mot en forme normale uniforme qui a au moins une interprétation alors u est en forme normale.*

Preuve En effet, si l'on note ϕ la contrainte de u et ψ la contrainte $\exists x_1 \ldots \exists x_n \sigma(\xi)$ définie comme dans le lemme 6 page précédente alors u est en forme normale uniforme, $\phi \wedge \psi$ est insatisfiable. De plus, par hypothèse, ϕ est satisfiable. Donc $\phi \Rightarrow \psi$ n'est pas valide. ◇

C'est le principal problème de la réécriture paramétrée. Cela empêche les raisonnements habituels sur les formes normales d'être utilisables directement.

4.3.6 Algorithmes de complétion

Nous définissons un algorithme de complétion pour les systèmes paramétrés.

Algorithme 2 (Complétion)
Entrée : *E ensemble d'équations paramétrées, $>$ un ordre sur les mots paramétrés*
Résultat : *R ensemble de règles paramétrées*
Algorithme : *On commence avec E et $R = \emptyset$*

1. *On met en forme normale pour R toutes les équations de E en éliminant les équations triviales.*
2. *On oriente toutes les équations de E en utilisant $>$ et on les met dans R. Si certaines ne sont pas orientables, on les laisse dans E.*
3. *On met en forme normale les membres droits des règles par R*
4. *On calcule toutes les paires critiques paramétrées pas encore calculées et on les ajoute comme des équations dans l'ensemble E*

5. *Si $E = \emptyset$ alors on renvoie R sinon on recommence en 1.*

Cet algorithme est très classique et s'adapte parfaitement au cas de la réécriture paramétrée. La recherche d'ordre peut être difficile. Mais elle est guidée par les ordres dans l'interprétation.

Nous pouvons optimiser la complétion au prix de l'automatisation en utilisant un oracle qui met en forme normale uniforme les mots au lieu de faire une mise en forme normale classique : nous obtenons ainsi un algorithme de complétion interactif bien plus puissant que l'algorithme automatique.

Chapitre 5

Réécriture paramétrée avec indices et exposants

5.1 Mots paramétrés avec indices et exposants

Dans ce chapitre, nous allons généraliser la notion de mot paramétré avec indices. En effet, le formalisme de la section précédente ne permet que de paramétrer des mots d'une longueur donnée, ce qui ne suffit pas dans la pratique. Ainsi la description du groupe symétrique par deux générateurs, un n-cycle b et une transposition t, nécessite l'équation $b^n = \epsilon$ dont le membre gauche est d'une longueur paramétrée. Nous allons donc introduire les outils permettant d'exprimer ce type de spécification. En particulier nous allons par exemple montrer que, sur l'alphabet $\{a; b\}$, le mot

$$a(ba)^N | N > 0$$

se réduit par la règle

$$(ab)^i a \to a^N | i > 0 \ \wedge \ N > 0$$

en un mot équivalent (dans un sens que l'on précisera) à

$$a^N (ba)^{N-1}$$

Nous fixons un ensemble infini dénombrable de variables I et une signature paramétrée réduite $\Sigma = \{a_i | \varphi\}_{\mathcal{N}}$ comme définie dans le chapitre 3 page 27, avec $i \in I$.

5.1.1 Définition et sémantique

Définition 63 *Un mot paramétré avec indices et exposants W est défini par la grammaire suivante :*

$$
\begin{array}{lcll}
W & := & M|\psi & \text{avec } \psi \text{ formule} \\
M & := & \epsilon & \\
 & | & NM & \\
 & | & {a_f}^e M & \text{avec } e \text{ et } f \text{ expressions arithmétiques} \\
 & | & (a_f a_{f'} N)^e M & \text{avec } e,\ f \text{ et } f' \text{ expressions arithmétiques} \\
N & := & \epsilon & \\
 & | & a_f N & \text{avec } f \text{ expression arithmétique}
\end{array}
$$

Les expressions e sont appelées exposants du mot.

Les a_f sont les lettres du mot.

Nous dirons que $(N)^e$ est un facteur avec exposant *ou avec puissance.*

Notons bien que notre grammaire interdit l'imbrication des exposants.

Cette définition est volontairement vague quant aux expressions arithmétiques et aux contraintes utilisées. En effet, a priori elles sont arbitraires. Dans la suite de cette section, nous nous restreindrons aux expressions arithmétiques linéaires et aux formules de la logique de Presburger afin de rester dans des logiques décidables. Si nous omettons ces restrictions, nos algorithmes qui utilisent des tests de satisfiabilité et de validité des formules ne sont plus que des semi-algorithmes. Dans la pratique, nous autoriserons donc des expressions arbitraires et des formules arbitraires, la seule restriction étant que le domaine des expressions doit être $\mathbb{Z}$.

Cette définition des mots avec exposants est à rapprocher de la notion de ρ-termes développée par Hubert Comon dans [10].

Dans tout ce chapitre, nous appellerons *mot* un mot paramétré avec indices et exposants.

Convention syntaxique : dans toute cette section, un mot avec exposants et indices sera noté par les lettres u, v, w. Un mot sans exposant sera noté x, y ou z. Les expressions de l'arithmétique de Presburger seront notées i,j,k,n et les paramètres (c'est-à-dire les éléments de $\mathcal{N}$) N,P,Q.

Exemple 64 Dans ce formalisme, il est possible d'écrire les mots suivants :

$$a_i^i a_1^{i-1} a_i | i \geq 2$$

$$a_i^3 a_{i+1}^{4*i+1} | \top$$

$$(a_i a_{i-1} a_{i-2})^k | i \leq k$$

$$(a_i a_{i-1})^j (a_j a_{j-1})^i | i + j \leq N$$

Par contre, le mot $(a^i b^i)^k | i = k$ n'est pas autorisé, car il contient des exposants imbriqués.

Nous étendons la notion de mot justement contraint pour se restreindre aux mots dont la contrainte implique la positivité des exposants.

Définition 65 *Le mot $u|\psi$ sur la signature $\Sigma = \{a_i|\varphi\}_N$ est dit* justement contraint *si et seulement si*
- *en notant $a_{f_1}, \ldots, a_{f_k}$ les lettres de u,*

$$\psi \Rightarrow \varphi[i \mapsto f_1] \ \wedge \ \ldots \ \wedge \ \varphi[i \mapsto f_k]$$

  est une formule valide et
- *en notant $e_1, \ldots, e_{k'}$ les exposants du mot,*

$$\psi \Rightarrow e_1 \geq 0 \ \wedge \ \ldots \ \wedge \ e_{k'} \geq 0$$

  est une formule valide.

Exemple 66 Sur la signature $\{a_i|0 \leq i \leq N\}_N$,
- les mots suivants sont justement contraints :
  1. $a_1|N \geq 1$
  2. $(a_i)^{N-i}|i \geq 0 \ \wedge \ N - i \geq 0$
  3. $(a_{i-1}a_ia_{i+1})^{N-1}|N \geq 1 \ \wedge \ N - 1 \geq i \geq 1$
- et les mots suivant ne sont pas justement contraints :
  1. $a_1|\top$, en effet $\top \Rightarrow 0 \leq 1 \leq N$ n'est pas valide.
  2. $a_i^i|i > 0$, car $i > 0 \Rightarrow 0 \leq i \leq N$ n'est pas valide.
  3. $a_i^{N-i}|i > 0$ car $i > 0 \Rightarrow 0 \leq N - i$ n'est pas valide.

Nous donnons maintenant une sémantique aux mots justement contraints. Il s'agit de donner une valeur entière aux indices et de remplacer les exposants par leur développement. Ainsi le mot $a_{N-1}^{N+1}|N \geq 1$ doit s'interpréter pour $N = 2$ en $a_1a_1a_1$.

Définition 67 *Soit $\nu : \mathcal{N} \rightarrow \mathbb{Z}$. La ν-interprétation du mot justement contraint $u|\psi$ avec l'interprétation $\mu : I \rightarrow \mathbb{Z}$ est définie si $\nu(\mu(\psi))$ est vraie par :*

$$[\![a_iv]\!]_\nu(\mu) = a_{\nu(\mu(i))}[\![v]\!]_\nu(\mu)$$
$$[\![\epsilon]\!]_\nu(\mu) = \epsilon$$
$$[\![u^iv]\!]_\nu(\mu) = \underbrace{[\![u]\!]_\nu(\mu) \cdots [\![u]\!]_\nu(\mu)}_{\nu(\mu(i))fois}[\![v]\!]_\nu(\mu)$$

Elle est indéfinie si $\nu(\mu(\psi))$ est fausse.

Nous montrerons dans la section suivante que la définition de l'interprétation sur les seuls mots justement contraints n'est pas une restriction.

Exemple 68 Nous donnons l'interprétation de quelques mots paramétrés avec indices et exposants sur la signature $\Sigma = \{a_i|0 < i < N\}_N$.

- Pour le mot $a_i^N|0 < i < N$,
  - si $\nu(N) = 0$ ou $\nu(N) = 1$ alors $[\![a_i^N|0 < i < N]\!]_\nu(\mu)$ est indéfini pour tout μ.
  - si $\nu(N) = 2$ et $\mu(i) = 1$ alors $[\![a_i^N|0 < i < N]\!]_\nu(\mu) = a_1a_1$ et est indéfini si $\mu(i) \neq 1$
  - si $\nu(N) = 3$ et $\mu(i) = 1$ alors $[\![a_i^N|0 < i < N]\!]_\nu(\mu) = a_1a_1a_1$
  - si $\nu(N) = 3$ et $\mu(i) = 2$ alors $[\![a_i^N|0 < i < N]\!]_\nu(\mu) = a_2a_2a_2$
- Pour le mot $a_{N-1}^{N-1}|N > 0$, si $\nu(N) = p$ tel que $p > 0$ alors pour tout μ,

$$[\![a_{N-1}^{N-1}|N > 0]\!]_\nu(\mu) = \underbrace{a_p \ldots a_p}_{pfois}$$

Notons que nous pouvons faire les mêmes extensions que dans le cas des mot paramétrés avec indices : constructeurs multiples et indices multiples. Dans les exemples où nous voudrons insister uniquement sur les exposants, nous utiliserons des signatures de la forme $\Sigma = \{a; b; c\}$ en omettant les indices. Ainsi nous parlerons du mot $(ab)^N c$ sur Σ au lieu de la signature $\{a_i|i = 0 \vee i = 1 \vee i = 2\}$ et du mot $(a_0a_1)^N a_2|N > 0$.

Dans ce chapitre nous allons établir que de nombreuses propriétés des mots avec exposants peuvent être vérifiées à l'aide de critères syntaxiques.

5.1.2 Substitutions et renommages des mots paramétrés

Nous étendons la définition des substitutions sur les mots avec indices et exposants.

Définition 69 *Soit σ une substitution des variables et des paramètres de Σ. La substitution du mot paramétré $u|\psi$ par σ est un mot paramétré défini par $\sigma(u)|\sigma(\psi)$ avec*
- $\sigma(a_iv) = a_{\sigma(i)}\sigma(v)$
- $\sigma(\epsilon) = \epsilon$
- $\sigma(v^e w) = \sigma(v)^{\sigma(e)}\sigma(w)$

Le renommage est défini comme au chapitre précédent.

C'est bien une extension de la définition pour les mots avec indices.

Exemple 70 Posons $\sigma(i) = j$, nous avons $\sigma(a_i|i > N) = a_j|j > N$. C'est un renommage de variable.

Lemme 18 *Soit ν une interprétation des paramètres et $u|\phi$ un mot paramétré. L'ensemble $\{[\![u|\phi]\!]_\nu(\mu) \ / \ \mu : I \to \mathbb{Z}\}$ est invariant par renommage des variables de $u|\phi$.*

La preuve est identique à celle du lemme 4 page 35.

5.1.3 Équivalence globale - Forme réduite

Nous généralisons les définitions introduites dans le chapitre sur les mots avec indices.

Définition 71 *Deux mots paramétrés $u|\phi$ et $v|\psi$ sont dits globalement équivalents si et seulement si pour toute interprétation des paramètres ν*

$$\{[\![u|\phi]\!]_\nu(\mu) \;/\; \mu : I \to \mathbb{Z}\} = \{[\![v|\psi]\!]_\nu(\mu) \;/\; \mu : I \to \mathbb{Z}\}$$

La proposition 2 page 35 se généralise en

Proposition 12 *Pour tout mot paramétré $u|\psi$ il existe un mot paramétré $v|\xi$ tel que :*
- *$u|\psi$ et $v|\xi$ sont globalement équivalents*
- *les indices et les exposants de v sont des variables deux à deux distinctes et sont les seules variables libres de ξ.*

Le mot paramétré $v|\xi$ est dit en forme réduite simple*.*

Preuve La preuve se fait en généralisant trivialement le lemme 6 page 36. ◇

Nous pouvons de nouveau combiner la forme réduite simple et la juste contrainte du mot en généralisant le théorème 3 page 38.

5.1.4 Similitude locale

Définition et sémantique

Nous généralisons aussi la notion de similitude locale.

Définition 72 *Deux mots paramétrés $u|\phi$ et $v|\psi$ sont dits* localement similaires *si et seulement si pour toutes interprétations $\nu : \mathcal{N} \to \mathbb{Z}$ et $\mu : I \to \mathbb{Z}$*
- *soit $\nu(\mu(\phi))$ et $\nu(\mu(\psi))$ sont fausses*
- *soit $\nu(\mu(\phi))$ et $\nu(\mu(\psi))$ sont vraies et*

$$[\![u|\phi]\!]_\nu(\mu) = [\![v|\psi]\!]_\nu(\mu)$$

Nous proposons maintenant un algorithme de décision de la locale similitude de deux mots de façon analogue au lemme 8 page 39. Rappelons son énoncé :

Lemme 19 (rappel de 8 page 39) *Soient deux mots paramétrés avec indices $u|\phi$ et $v|\psi$. Les deux mots sont localement similaires si et seulement si*
- *$\phi \;\wedge\; \psi \Rightarrow u \equiv v$ est une formule valide et*
- *$\phi \Leftrightarrow \psi$ est une formule valide.*

La généralisation n'est plus du tout évidente. En effet, la première contrainte contient $u \equiv v$ qui dans le cas des mots avec indices était simplement une notation désignant la conjonction des égalités des indices deux à deux. Cette notion n'a plus un sens immédiat dans le cas des mots contenant des exposants.

Nous allons donc définir la notion de *formule de similitude de deux mots.*

Définition 73 *Une formule de similitude entre les deux mots $u|\phi$ et $v|\psi$ notée*

$$u \equiv v|\phi \ \wedge \ \psi$$

est une formule telle que pour tout $\nu : \mathcal{N} \to \mathbb{Z}$ et tout $\mu : I \to \mathbb{Z}$,

$$\nu(\mu(u \equiv v|\phi \ \wedge \ \psi)) = \top \quad ssi \quad (\nu(\mu(u)) = \nu(\mu(v))) \ et \ \nu(\mu(\phi \ \wedge \ \psi))) = \top$$

Exemple 74 Une simple preuve par récurrence sur l'entier N montre que la formule de similitude de $a(ba)^N|N \geq 0$ et $(ab)^N a|N \geq 0$ notée

$$a(ba)^N \equiv (ab)^N a|N \geq 0$$

est équivalente à $N \geq 0$.

Les ν-interprétations sont des structures infinies difficiles à manipuler directement. Ainsi la définition de la formule de similitude de deux mots n'est pas calculatoire, ce qui ne peut pas nous satisfaire. Nous allons maintenant donner des algorithmes qui permettent de calculer cette formule syntaxiquement, c'est-à-dire sans énumérer les interprétations.

Algorithme de résolution de la formule de similitude

Une première approche peut être de chercher à trouver une forme normale pour les mots avec exposants, mais il n'existe pas à notre connaissance de système canonique qui normalise les mots avec exposants et nos efforts dans ce sens n'ont pas été concluants.

Nous donnons un algorithme qui, étant donnée une formule de similitude $u \equiv v|\phi$, calcule une formule *arithmétique* ψ qui lui est équivalente.

Définition 75 *Une formule de similitude est dite en forme résolue si c'est une formule de l'arithmétique.*

Donnons d'abord un exemple d'exécution de l'algorithme. À chaque étape nous transformons la formule de similitude en une formule équivalente en lui ajoutant des contraintes arithmétiques pour remplacer les contraintes entre les mots. Le résultat est une formule de similitude résolue.

Exemple 76
- Formule initiale :

  $$a(ba)^n \equiv (ab)^n a|n \geq 0$$

  Nous devons distinguer deux cas suivant la nullité de n.
- Une disjonction d'équations :

  $$(a \equiv a|n = 0 \ \wedge \ n \geq 0) \vee (((ba)^n \equiv (ba)^{n-1}ba)|n > 0 \ \wedge \ n \geq 0)$$

  Dans le second membre de la disjonction, on remarque que nous avons effectué une permutation circulaire à l'intérieur de la puissance.

– Factorisation :

$$(n = 0 \ \wedge \ n \geq 0) \vee (((ba)^n \equiv (ba)^{n-1+1})|n > 0 \ \wedge \ n \geq 0)$$

Le premier membre n'a plus de $\equiv$ et est donc en forme résolue ; dans le second nous avons factorisé le contenu de la puissance le plus possible.

– Élimination de puissances :

$$(n = 0 \ \wedge \ n \geq 0) \vee (n = n - 1 + 1 \ \wedge \ n > 0 \ \wedge \ n \geq 0)$$

– Simplification : $n = 0 \vee n > 0$

– Nous avons une formule de similitude résolue : $n \geq 0$

Cet exemple explicite les différents sous-problèmes à résoudre. La permutation sous les puissances, la factorisation d'un mot sans puissance dans un mot avec puissances, et l'élimination des puissances. Toutes ces notions seront détaillées dans la suite de cette section.

Nous présentons l'algorithme de résolution de la formule de similitude sous forme de règles d'inférence. Toute cette section est consacrée à son explication et à sa preuve.

Algorithme 3 (Résolution de la formule de similitude) Entrée : *une formule de similitude* $u \equiv v|\psi$.

Résultat : *une formule de similitude résolue équivalente à l'entrée.*

Algorithme :

$$\frac{\epsilon \equiv \epsilon|\phi}{\phi}$$

$$\frac{\epsilon \equiv a_i u|\phi}{\perp}$$

$$\frac{\epsilon \equiv x^n u|\phi}{\epsilon \equiv u|\phi \ \wedge \ n = 0}$$

$$\frac{a_i u \equiv a_j v|\phi}{u \equiv v|\phi \ \wedge \ i = j}$$

$$\frac{a_i u \equiv (a_j x)^n v|\phi}{(a_i u \equiv v|\phi \ \wedge \ n = 0) \vee (u \equiv (x a_j)^{n-1} x v|\phi \ \wedge \ n > 0 \ \wedge \ i = j)}$$

$$\frac{x^n u \equiv y^p v|\phi \qquad si \ \left\{ \begin{array}{l} |x| = |y| \ et \\ \quad \neg \mathit{IsPrefix}(x, u, \phi) \\ \quad\quad \wedge \neg \mathit{IsPrefix}(y, v, \phi) \end{array} \right\} \ valide}{\begin{array}{l} (u \equiv v|\phi \ \wedge \ n = p \ \wedge \ \neg n = 0 \ \wedge \ x \equiv y) \vee \\ (x^n u \equiv v|\phi \ \wedge \ p = 0) \vee \\ (u \equiv y^p v|\phi \ \wedge \ n = 0) \end{array}}$$

$$\frac{x^n u \equiv y^p v|\phi \quad si \begin{cases} |x| < |y| \ ou \\ \begin{cases} |x| = |y| \ et \\ \neg \mathit{IsPrefix}(x,u,\phi) \wedge \\ \neg \mathit{IsPrefix}(y,v,\phi) \\ \ non\ valide \end{cases} \end{cases}}{\displaystyle\bigvee_{\substack{(n',u',\phi'_u) \in \mathit{LFact}(x,u,\phi) \\ (p',v',\phi'_v) \in \mathit{LFact}(x,y^p v,\phi)}} (x^{n+n'}u' \equiv x^{p'}v'|\phi \ \wedge \ \phi'_u \ \wedge \ \phi'_v)}$$

Notons que cet algorithme est déterministe puisque les deux dernières règles s'excluent mutuellement à cause de la condition d'application et les autres règles ne simplifient pas des équations de même forme.

Cet algorithme dépend de deux notions pour lesquelles nous allons donner des définitions formelles et des algorithmes de construction :

- *IsPrefix*(x, u, ϕ) (où x est un mot sans puissance et $u|\phi$ un mot contraint avec exposants) est une formule telle que si $(x_0, u_0) \in [\![x, u|\phi]\!]_\nu$ alors x_0 est un préfixe de u_0. L'algorithme 4 permet de calculer cette formule.
- *LFact*(x, u, ϕ) (acronyme désignant *Left Factorizations* c'est-à-dire factorisations à gauche) est l'ensemble des triplets (n, u', ϕ') où n est une expression arithmétique et $u'|\phi'$ un mot contraint avec exposants tels que l'équation paramétrée $u \equiv x^{n'}u'|\phi'$ est valide et *IsPrefix*(x, u', ϕ') n'est pas satisfiable. Il s'agit informellement de l'ensemble des écritures possibles distinctes de $u|\phi$ comme une puissance de x concaténée à un mot paramétré dont aucune instance n'a x pour préfixe. Cet ensemble est calculé par l'algorithme 6 page 75.

Donnons maintenant un algorithme de calcul de *IsPrefix*.

Algorithme 4 (*IsPrefix*) Entrée : *x un mot sans exposant et $u|\psi$ un mot paramétré avec exposants. L'algorithme débute sur le triplet (x, u, ψ).*

Résultat : *une formule ϕ vérifiant la proposition 13.*

Algorithme :

$$\frac{\epsilon, w, \psi}{\psi} \tag{5.1}$$

$$\frac{a_i x, \epsilon, \psi}{\perp} \tag{5.2}$$

$$\frac{a_i x, a_j u, \psi}{x, u, i = j \ \wedge \ \psi} \tag{5.3}$$

$$\frac{a_i x, (a_j y)^n u, \psi}{(a_i x, u, \psi \ \wedge \ n = 0) \vee (x, (y a_j)^{n-1} y u, \psi \ \wedge \ n > 0 \ \wedge \ i = j)} \tag{5.4}$$

Exemple 77 Nous calculons *IsPrefix*$(a_i a_{i+1}, a_i (a_j a_i)^n a_{i+1}, n \geq 0)$ et nous obtenons successivement :

- $a_i a_{i+1}, a_i (a_j a_i)^n a_{i+1}, n \geq 0$

- $a_{i+1}, (a_j a_i)^n a_{i+1}, n \geq 0 \ \wedge \ i = i$
- $(a_{i+1}, a_{i+1}, n \geq 0 \ \wedge \ i = i \ \wedge \ n = 0) \vee (\epsilon, (a_i a_j)^{n-1} a_i a_{i+1}, n \geq 0 \ \wedge \ i = i \ \wedge \ n > 0 \ \wedge \ i+1 = j)$
- $(\epsilon, \epsilon, n \geq 0 \ \wedge \ i = i \ \wedge \ n = 0 \ \wedge \ i+1 = i+1) \vee (n \geq 0 \ \wedge \ i = i \ \wedge \ n > 0 \ \wedge \ i+1 = j)$
- $(n \geq 0 \ \wedge \ i = i \ \wedge \ n = 0 \ \wedge \ i+1 = i+1) \vee (n \geq 0 \ \wedge \ i = i \ \wedge \ n > 0 \ \wedge \ i+1 = j)$

En simplifiant nous obtenons la formule $n = 0 \vee (n > 0 \ \wedge \ i+1 = j)$, qui n'est pas valide. Si nous remplaçons j par $i+1$ nous avons un formule valide.

Proposition 13 *Soit x un mot sans exposant et $u|\psi$ un mots paramétré avec indices et exposants. Le calcul de $\mathit{IsPrefix}(x, u, \psi)$ termine et donne une formule logique telle que pour tout $\nu : \mathcal{N} \to \mathbb{Z}$ et tout $\mu : I \to N$,*

$$\nu(\mu(\mathit{IsPrefix}(x, u, \psi))) = \top$$

si et seulement si
- *$\nu(\mu(\psi)) = \top$ et*
- *$\nu(\mu(x))$ est un préfixe de $\nu(\mu(u))$.*

Preuve La terminaison est évidente puisque chaque règle diminue
- soit la taille du préfixe que l'on cherche (le x)
- soit le nombre d'exposants du mot que l'on analyse (le u).

Montrons la correction. Nous fixons $\nu : \mathcal{N} \to \mathbb{Z}$ et $\mu : I \to \mathbb{Z}$.
- Règle 5.1 page ci-contre :
  - Si $\nu(\mu(\mathit{IsPrefix}(\epsilon, w, \psi)$ est vraie alors en particulier $\nu(\mu(\psi))$ est vraie.
  - Si $\nu(\mu(\psi))$ est vraie, comme $\nu(\mu(\epsilon))$ est trivialement préfixe de tout mot $\nu(\mu(w))$, on a bien $\nu(\mu(\mathit{IsPrefix}(\epsilon, w, \psi)$ vraie.
- Règle 5.2 page précédente : $\nu(\mu(\mathit{IsPrefix}(a_i x, \epsilon, \psi)$ est toujours fausse puisque $\nu(\mu(a_i x))$ est un mot non vide qui n'est donc jamais préfixe de ϵ.
- Règle 5.3 page ci-contre :
  - Si $\nu(\mu(\mathit{IsPrefix}(a_i x, a_j u, \psi)$ est vraie alors
    - $\nu(\mu(\psi))$ est vraie,
    - $\nu(\mu(a_i x))$ est préfixe de $\nu(\mu(a_j u))$ donc $\nu(\mu(i)) = \nu(\mu(j))$.

    Ainsi $\nu(\mu(\psi \ \wedge \ i = j))$ est vraie et $\nu(\mu(x))$ est préfixe de $\nu(\mu(u))$.
  - Si $\nu(\mu(\mathit{IsPrefix}(x, u, \psi \ \wedge \ i = j)$ est vraie alors $\nu(\mu(i)) = \nu(\mu(j))$. Donc $\nu(\mu(a_i)) = \nu(\mu(a_j))$ ce qui implique que $\nu(\mu(a_i)).\nu(\mu(x))$ est un préfixe de $\nu(\mu(a_j)).\nu(\mu(u))$.
- Règle 5.4 page précédente :
  - Si $\nu(\mu(\mathit{IsPrefix}(a_i x, (a_j y)^n u, \psi)))$ est vraie alors $\nu(\mu(\psi))$ est vraie et donc comme le mot est justement contraint, $\nu(\mu(n)) \geq 0$.
    - Si $\nu(\mu(n)) = 0$ alors $\nu(\mu(a_i x))$ est un préfixe de $\nu(\mu(u))$ et on a bien $\nu(\mu(\mathit{IsPrefix}(a_i x, u, \psi \ \wedge \ n = 0)))$.
    - Si $\nu(\mu(n)) > 0$ alors $\nu(\mu(a_i x))$ est un préfixe de $\nu(\mu(a_j y (a_j y)^{n-1} u))$. On en déduit que $\nu(\mu(i)) = \nu(\mu(j))$ et $\nu(\mu(x))$ préfixe du mot $\nu(\mu(y(a_j y)^{n-1} u))$ qui est égal à $\nu(\mu((y a_j)^{n-1} y u))$. Nous avons bien montré que

$$\nu(\mu(\mathit{IsPrefix}(x, y(a_j y)^{n-1} u, n > 0 \ \wedge \ i = j \ \wedge \ \psi))$$

est vraie.

– Quant à la réciproque, montrons que chaque élément de la disjonction implique que $\nu(\mu(\mathit{IsPrefix}(a_i x, (a_j y)^n u, \psi)))$ est vraie.
  – Si $\nu(\mu(\mathit{IsPrefix}(a_i x, u, \psi \ \wedge \ n = 0)))$ est vraie alors $\nu(\mu(\psi))$ est vraie et $\nu(\mu(n)) = 0$. Comme $\nu(\mu(a_i x))$ est un préfixe de $\nu(\mu(u))$, on a bien $\nu(\mu(\mathit{IsPrefix}(a_i x, (a_j y)^n u, \psi)))$ vraie.
  – Si $\mathit{IsPrefix}(x, (ya_j)^{n-1} yu, \psi \ \wedge \ n > 0 \ \wedge \ i = j)$ est vraie alors $\nu(\mu(\psi))$ est vraie, $\nu(\mu(n)) > 0$ et $\nu(\mu(i)) = \nu(\mu(j))$. Comme $\nu(\mu(x))$ est un préfixe de $\nu(\mu((ya_j)^{n-1} yu))$, on a $\nu(\mu(a_i x))$ préfixe de $\nu(\mu(a_j (ya_j)^{n-1} yu))$. Ce dernier est égal à $\nu(\mu((a_j y)^n u))$ ce qui permet de conclure que $\nu(\mu(\mathit{IsPrefix}(a_i x, (a_j y)^n u, \psi)))$ est vraie.

$\diamond$

Nous modifions cet algorithme afin de pouvoir décrire de façon finie toutes les instances de $u|\psi$ de la forme $xu'|\psi'$. Pour cela, il suffit de changer la première règle en

$$\frac{\epsilon, w, \psi}{w|\psi}$$

et de remplacer les $\vee$ par des unions ensemblistes. On obtient l'algorithme suivant :

Algorithme 5 (*RemovePrefix***) Entrée :** *x un mot sans exposant et $u|\psi$ un mots paramétré avec exposants. L'algorithme commence avec le triplet (x, u, ψ).*

Résultat : *un ensemble de mots $u'|\phi'$ vérifiant la proposition 14.*

Algorithme :

$$\frac{\epsilon, w, \psi}{w|\psi}$$

$$\frac{a_i x, \epsilon, \psi}{\emptyset}$$

$$\frac{a_i x, a_j u, \psi}{x, u, i = j \ \wedge \ \psi}$$

$$\frac{a_i x, (a_j y)^n u, \psi}{\{a_i x, u, \psi \ \wedge \ n = 0\} \cup \{x, (ya_j)^{n-1} yu, \psi \ \wedge \ n > 0 \ \wedge \ i = j\}}$$

Proposition 14 *$\mathit{RemovePrefix}(x, u|\phi)$ est un ensemble dont tout élément $u'|\phi'$ est tel que pour tout $\nu : \mathcal{N} \to \mathbb{Z}$ et tout $\mu : I \to \mathbb{Z}$*

– *si $\nu(\mu(\phi')) = \top$ alors*

$$[\![xu'|\phi']\!]_\nu(\mu) = [\![u|\phi]\!]_\nu(\mu)$$

– *si $[\![x, u|\phi]\!]_\nu(\mu) = (x_0, u_0)$ et il existe u'_0 tel que $u_0 = x_0 u'_0$ alors il existe un élément $u'|\phi' \in \mathit{RemovePrefix}(x, u|\phi)$ tel que $u'_0 = [\![u'|\phi']\!]_\nu(\mu)$*

– *si $x \neq \epsilon$, alors*
  – *soit u' a strictement moins de facteurs avec des puissances que u*

– *soit la longueur du plus long préfixe sans puissance de u' est strictement plus petite que la longueur du plus long préfixe sans puissance u.*

Preuve La preuve est similaire à celle de la proposition 13 page 73. ◇

Nous pouvons maintenant donner les règles pour *LFact*. Cet algorithme permet de factoriser un préfixe sans exposant autant de fois que possible dans un mot quelconque. Le nombre d'occurrence d'un préfixe dépend de la valeur des paramètres, mais nous allons montrer que l'on peut énumérer toutes les possibilités de façon finie.

Algorithme 6 (*LFact*) **Entrée :** *x un mot sans exposant et $u|\phi$ un mots paramétré avec exposants. L'algorithme commence avec le triplet $0, u, \phi$.*

Résultat : *un ensemble de triplets (n', u', ϕ') vérifiant la proposition 15 page suivante*

Algorithme :

$$\frac{n, yu, \phi \qquad\qquad si\ |x| = |y|}{(n+1, u, \phi\ \wedge\ x \equiv y); Result(n, yu, \phi\ \wedge\ \neg x \equiv y)}$$

$$\frac{n, y^p u, \phi \qquad\qquad si\ |x| = |y|}{(n+p, u, \phi\ \wedge\ x \equiv y); (n, u, \phi\ \wedge\ p = 0); Result(n, y^p u, \phi\ \wedge\ \neg x \equiv y\ \wedge\ \neg p = 0)}$$

$$\frac{n, y, \phi \qquad\qquad si\ |x| > |y|}{Result(n, y, \phi)}$$

$$\frac{n, yz^p v, \phi \qquad\qquad si \begin{cases} |x| > |y| \\ R = RemovePrefix(x, yz^p v, \phi) \end{cases}}{\bigcup_{v'|\psi' \in R} \{(n+1, v', \phi\ \wedge\ \psi')\}; Result(n, yz^p v, \phi\ \wedge\ (\ \wedge_{\ u'|\psi' \in R} \neg\psi'))\}}$$

$$\frac{n, y^p v, \phi \qquad\qquad si \begin{cases} |x| \neq |y| \\ d_x = ppcm(|x|, |y|)/|x| \\ d_y = ppcm(|x|, |y|)/|y| \end{cases}}{\begin{array}{l} \bigcup_{r=0}^{d_y-1} \{n; y^r u; \phi\ \wedge\ p = r\} \\ \bigcup_{r=0}^{d_x-1} \left\{ Result \begin{pmatrix} n+r; \\ \tilde{y}^{p-d_y} Suff_{|x^r|}(y^{d_y}) u; \\ \phi\ \wedge\ p \geq d_y\ \wedge\ x^r \equiv Pref_{|x^r|}(y^{d_y})\ \wedge\ \neg(x^{r+1} \equiv Pref_{|x^{r+1}|}(y^{d_y})) \end{pmatrix} \right\} \\ \bigcup_{r=0}^{d_y-1} \{n + \frac{d_x}{d_y}(p-r); y^r u; \phi\ \wedge\ x^{d_x} \equiv y^{d_y}\ \wedge\ p \geq d_y\ \wedge\ (p = r \mod d_y)\} \end{array}}$$

Dans la dernière règle, $\tilde{y}$ désigne une permutation des lettres de y telle que

$$\tilde{y}^{p-d_y} Suff_{|x^r|}(y^{d_y}) = Suff_{|x^r|}(y^{d_y}) y^{p-d_y}$$

Il en existe évidemment une.

Rappelons que

$$Pref_j(a_{i_1}\dots a_{i_j}\dots a_{i_k}) = a_{i_1}\dots a_{i_j}$$

avec j en entier naturel et

$$Suff_j(a_{i_1}\dots a_{i_j}\dots a_{i_k}) = a_{i_j}\dots a_{i_k}$$

Exemple 78 Donnons par exemple le début du calcul de $LFact(a_ia_j,(a_ka_la_m)^pu, p \geq 0)$. Appliquons la dernière règle sur $0,(a_ka_la_m)^p, p \geq 0$. Dans cet exemple $d_x = 3$ et $d_y = 2$.

$$\begin{array}{ll}
0, u & p = 0 \\
0, a_ka_la_mu & p = 1 \\
Result(0,(a_ka_la_m)^pu & p \geq 2 \wedge \neg(i = k \wedge j = l) \qquad\qquad\qquad\qquad\qquad\qquad) \\
Result(1,(a_ma_ka_l)^{p-1}a_mu & p \geq 2 \wedge \quad (i = k \wedge j = l) \wedge \neg(i = m \wedge j = k) \qquad\qquad) \\
Result(2,(a_la_ma_k)^{p-2}a_la_mu & p \geq 2 \wedge \quad (i = k \wedge j = l) \wedge \quad (i = m \wedge j = k) \wedge \neg(i = l \wedge j = m)) \\
3p/2, u & p \geq 2 \wedge \quad (i = k \wedge j = l) \wedge \quad (i = m \wedge j = k) \wedge \quad (i = l \wedge j = m) \wedge p \equiv 0[2] \\
3(p-1)/2, a_ka_la_mu & p \geq 2 \wedge \quad (i = k \wedge j = l) \wedge \quad (i = m \wedge j = k) \wedge \quad (i = l \wedge j = m) \wedge p \equiv 1[2]
\end{array}$$

Exemple 79 Donnons un autre cas intéressant pour le calcul de $LFact(a_ia_ja_k,(a_la_m)^pu, p \geq 0)$. Nous appliquons la dernière règle sur $0,(a_la_m)^p, p \geq 0$.

$$\begin{array}{rlll}
 & 0 & u & p = 0 \\
 & 0 & a_la_mu & p = 1 \\
 & 0 & a_la_ma_la_mu & p = 2 \\
Result(& 0 & (a_la_m)^pu & p \geq 3 \wedge \neg(i = l \wedge j = m \wedge k = l) \qquad\qquad\qquad\qquad\qquad\qquad) \\
Result(& 1 & (a_ma_l)^{p-1}a_mu) & p \geq 3 \wedge \quad (i = l \wedge j = m \wedge k = l) \wedge \neg(i = m \wedge j = l \wedge k = m) \\
 & 2\frac{p}{3} & u & p \geq 3 \wedge \quad (i = l \wedge j = m \wedge k = l) \wedge \quad (i = m \wedge j = l \wedge k = m) \wedge p \equiv 0[3] \\
 & 2\frac{p-1}{3} & a_la_mu & p \geq 3 \wedge \quad (i = l \wedge j = m \wedge k = l) \wedge \quad (i = m \wedge j = l \wedge k = m) \wedge p \equiv 1[3] \\
 & 2\frac{p-2}{3} & a_la_ma_la_mu & p \geq 3 \wedge \quad (i = l \wedge j = m \wedge k = l) \wedge \quad (i = m \wedge j = l \wedge k = m) \wedge p \equiv 2[3]
\end{array}$$

Lemme 20 (Décroissance de $LFact$) *Pour tout $(n, v, \phi) \in LFact(x, u, \psi)$,*

- *soit le nombre d'exposants dans v est strictement plus petit que celui dans u*
- *soit u et v ont le même nombre d'exposants et la taille du plus long préfixe sans exposant de v est strictement plus petite que celle du plus long préfixe sans exposants de u.*

Preuve Chaque règle respecte cet ordre y compris la dernière grâce à l'utilisation de $\tilde{y}$. $\diamond$

Proposition 15 (Sémantique de $LFact$) *Pour tout $(n, v, \phi) \in LFact(x, u, \psi)$*

1. *pour toute interprétation $\nu : \mathcal{N} \to \mathbb{Z}$ et $\mu : I \to \mathbb{Z}$*
   - *si $\nu(\mu(\phi))$ est vraie alors $\nu(\mu(\psi))$ est vraie et*

     $$[\![x^nv|\phi]\!]_\nu(\mu) = [\![u|\psi]\!]_\nu(\mu)$$

   - *si $\nu(\mu(\psi))$ est vraie alors il existe $(n, v, \phi) \in LFact(x, u, \psi)$ tel que $\nu(\mu(\phi))$ est vraie et*

     $$[\![u|\psi]\!]_\nu(\mu) = [\![x^nv|\phi]\!]_\nu(\mu)$$

2. $\neg IsPrefix(x, v, \phi)$ *est valide*

Preuve Par cas la preuve est claire. $\diamond$

Nous pouvons maintenant prouver la correction de l'algorithme de résolution d'une formule de similitude.

Lemme 21 *Soit* $u \equiv v|\phi$ *une formule de similitude. L'algorithme de résolution termine sur* $u \equiv v|\phi$ *et donne une formule* χ *telle que pour toute interprétation des paramètres* ν *et des variables* μ*,*

- *si* $\nu(\mu(\chi))$ *est vraie alors*
$$\nu(\mu(u)) = \nu(\mu(v))$$
est vraie si et seulement si
$$\nu(\mu(\phi)) = \top$$
- *si* $\nu(\mu(\chi))$ *est fausse alors*
$$\nu(\mu(\phi)) = \bot$$
ou
$$\nu(\mu(u)) \neq \nu(\mu(v))$$

Preuve

- Terminaison : en omettant la dernière, chaque règle diminue strictement soit le nombre d'exposants dans l'équation, soit la somme des tailles des plus grands préfixes sans exposant des membres de la formule de similitude. Cela nous donne un ordre lexicographique qui assure la terminaison de l'algorithme sans la dernière règle. La dernière règle fait décroître cette mesure, mais seulement au sens large d'après le théorème 20 page précédente. Le théorème 15 page ci-contre montre que cette règle ne peut pas s'appliquer deux fois de suite. La terminaison de l'algorithme est ainsi établie.
- Correction : montrons que chaque règle transforme le problème en un problème équivalent. Nous énumérons toutes les règles une par une dans l'ordre de l'algorithme. On se fixe ν (resp. μ) une interprétation des paramètres (resp. des variables).
  - $\epsilon \equiv \epsilon|\phi$:
    - si $\nu(\mu(\phi)) = \top$ alors $\nu(\mu(\epsilon)) = \nu(\mu(\epsilon))$ est vraie ssi $\nu(\mu(\phi))$.
    - si $\nu(\mu(\phi)) = \bot$ alors $\chi = \bot$ satisfait bien les hypothèses du théorème.
  - $\epsilon \equiv a_i u|\phi$:
    - si $\chi = \bot$ est vraie alors $\nu(\mu(\epsilon)) = \nu(\mu(a_i u))$ ssi $\nu(\mu(\phi))$ est vraie, ce qui établit le premier cas de figure.
    - si $\chi = \bot$ est fausse alors soit $\nu(\mu(\phi))$ est fausse et nous concluons, soit elle est vraie et dans ce cas, $\nu(\mu(\epsilon)) = \epsilon$, $[\![a_i u]\!]_\nu(\mu) =$ est définie et est de longueur au moins 1, et est donc distincte de ϵ. Ce qui établit le deuxième cas.

  Pour les autre règles, de la forme $\frac{u \equiv v|\psi}{u' \equiv v'|\psi'}$, il suffit de montrer que
$$\nu(\mu(u)) = \nu(\mu(v)) \ \wedge \ \nu(\mu(\psi)) \Leftrightarrow \nu(\mu(u')) = \nu(\mu(v')) \ \wedge \ \nu(\mu(\psi'))$$

- $\epsilon \equiv x^n u|\phi$: la formule
$$\nu(\mu(\epsilon)) = \nu(\mu(x^n u))$$
est équivalente à
$$\nu(\mu(\epsilon)) = \nu(\mu(u)) \ \wedge \ n = 0$$
pour des raisons de longueur.
- $a_i u \equiv a_j v|\phi$: la formule
$$\nu(\mu(a_i u)) = \nu(\mu(a_j v))$$
est équivalente à
$$\nu(\mu(u)) = \nu(\mu(v)) \ \wedge \ \nu(\mu(i)) = \nu(\mu(j))$$
- $a_i u \equiv (a_j x)^n v|\phi$: la formule
$$\nu(\mu(a_i u)) = \nu(\mu((a_j x)^n v)) \ \wedge \ \nu(\mu(\phi))$$
est équivalente à la disjonction
$$\begin{array}{l}((\nu(\mu(a_i u)) = \nu(\mu(v)) \ \wedge \ n = 0) \vee \\ (\nu(\mu(a_i u)) = \nu(\mu(a_j x(a_j x)^{n-1} v)) \ \wedge \ n > 0)) \ \wedge \ \nu(\mu(\phi))\end{array}$$
En effet, comme l'équation est justement contrainte, $\phi \Rightarrow n \geq 0$ est valide. La seconde partie de la disjonction est équivalente à
$$\nu(\mu(i)) = \nu(\mu(j)) \ \wedge \ (\nu(\mu(u)) = \nu(\mu(x(a_j x)^{n-1} v)) \ \wedge \ n > 0)$$
Puis, par récurrence sur n, on peut montrer que
$$\nu(\mu(x(a_j x)^{n-1})) = \nu(\mu((x a_j)^{n-1} x))$$
pour tout $n > 0$. Nous obtenons bien l'équivalence cherchée.
- $x^n u \equiv y^p v|\phi$: supposons que les conditions d'application de la règle sont vérifiées i.e. $|x| = |y|$ et $\neg \mathit{IsPrefix}(x, u, \phi) \ \wedge \ \neg \mathit{IsPrefix}(y, v, \phi)$ valide. On note
$$E_1 = (\nu(\mu(x^n u)) = \nu(\mu(y^p v)) \ \wedge \ \nu(\mu(\phi)))$$
$$E_2^1 = (\nu(\mu(u = v \ \wedge \ \phi \ \wedge \ n = p \ \wedge \ \neg n = 0 \ \wedge \ x = y)))$$
$$E_2^2 = (\nu(\mu(x^n u = v \ \wedge \ \phi \ \wedge \ p = 0)))$$
et
$$E_2^3 = (\nu(\mu(u = y^p v \ \wedge \ \phi \ \wedge \ n = 0)))$$
Nous devons montrer que $E_1 = E_2^1 \vee E_2^2 \vee E_2^3$. Raisonnons suivant les valeurs de n et p.
  - si $\nu(\mu(n)) = 0$ alors E_1 est équivalent à E_2^3
  - si $\nu(\mu(p)) = 0$ alors E_1 est équivalent à E_2^2

– sinon, les deux exposants sont non nuls. E_1 contient la formule

$$\underbrace{\nu(\mu(x))\dots\nu(\mu(x))}_{\nu(\mu(n))}\nu(\mu(u)) = \underbrace{\nu(\mu(y))\dots\nu(\mu(y))}_{\nu(\mu(p))}\nu(\mu(v))$$

Comme $\neg \mathit{IsPrefix}(x, u, \phi) \wedge \neg \mathit{IsPrefix}(y, v, \phi)$ est valide, on sait d'après le théorème 13 page 73 que $\nu(\mu(x))$ (resp. $\nu(\mu(y))$) n'est pas un préfixe de $\nu(\mu(u))$ (resp. $\nu(\mu(v))$). Comme les exposants sont non nuls et que $|x| = |y|$, nous déduisons que $\nu(\mu(x)) = \nu(\mu(y))$ est vraie. Puis, pour des raison de longueur, nous obtenons $\nu(\mu(n)) = \nu(\mu(p))$. On peut donc simplifier l'équation en $\nu(\mu(u)) = \nu(\mu(v))$. Nous avons montré que E_1 est équivalent à E_2^1

Nous avons bien établi la correction de la règle.

– Il reste la dernière règle. Sa correction découle immédiatement du théorème 15 page 76

La correction de l'algorithme est bien établie.

◇

Théorème 7 *L'algorithme de résolution de formule de similitude est correct.*

Preuve C'est une conséquence immédiate du théorème précédent. ◇

Exemple 80 Donnons une exécution complète de l'algorithme pour résoudre $a_i(a_j a_i)^n \equiv (a_i a_j)^n a_i | i, j, n \geq 0$.

Règle 5 : $(a_i(a_i a_j)^n \equiv a_i | i, j, n \geq 0 \wedge n = 0) \vee ((a_j a_i)^n \equiv (a_j a_i)^{n-1} a_j a_i | i, j, n \geq 0 \wedge n > 0 \;\wedge\; i = i)$

Règle 4,règle 7 : – La première équation donne $((a_i a_j)^n \equiv \epsilon | i, j, n \geq 0 \wedge n = 0 \wedge i = i)$.

– Pour la seconde, la règle 7 s'applique puisque $|x| = |y| = 2$ et

$$\begin{aligned}
&\neg \mathit{IsPrefix}(a_j a_i, a_j a_i, (i, j, n \geq 0 \;\wedge\; n > 0 \;\wedge\; i = i)) \\
&\quad = \neg((i, j, n) \geq 0 \;\wedge\; n > 0 \;\wedge\; i = i \;\wedge\; i = i \;\wedge\; j = j) \\
&\quad = (n \leq 0 \vee i < 0 \vee j < 0)
\end{aligned}$$

n'est pas valide. On calcule donc

$$\begin{aligned}
F_1 &= \mathit{LFact}(a_j a_i, \epsilon, (i, j, n) \geq 0 \;\wedge\; n > 0 \;\wedge\; i = i) \\
&= \{\mathit{Result}(0, \epsilon, (i, j) \geq 0 \;\wedge\; n > 0)\}
\end{aligned}$$

et

$$\begin{aligned}
F_2 &= \mathit{LFact}(a_j a_i, (a_j a_i)^{n-1} a_j a_i, (i, j, n) \geq 0 \;\wedge\; n > 0 \;\wedge\; i = i) \\
&= \{\mathit{Result}(n, \epsilon, (i, j) \geq 0 \;\wedge\; n > 0); \\
&\qquad \mathit{Result}(1, \epsilon, (i, j) \geq 0 \;\wedge\; n = 1); \\
&\qquad \mathit{Result}(_, _, \bot)\}
\end{aligned}$$

Il y a trois éléments de la forme $\mathit{Result}(\_, \_, \bot)$ que nous pouvons clairement omettre dans la suite l'algorithme puisqu'ils se réduiront à $\bot$. Il nous reste donc deux équations : $(a_ja_i)^n \equiv (a_ja_i)^n, (i,j \geq 0 \ \wedge \ n > 0)$ et $(a_ja_i)^n \equiv (a_ja_i)^1, (i,j \geq 0 \ \wedge \ n = 1)$.

Il nous reste donc

$$\begin{array}{l} ((a_ia_j)^n \equiv \epsilon | i,j \geq 0 \ \wedge \ n = 0) \\ (a_ja_i)^n \equiv (a_ja_i)^n | i,j \geq 0 \ \wedge \ n > 0 \\ (a_ja_i)^n \equiv (a_ja_i)^1 | i,j \geq 0 \ \wedge \ n = 1 \end{array}$$

Règle 3, règle 6, règle 6 :

$$\begin{array}{l} (\epsilon \equiv \epsilon | i,j \geq 0 \ \wedge \ n = 0 \ \wedge \ n = 0) \\ \epsilon \equiv \epsilon | i,j \geq 0 \ \wedge \ n > 0 \ \wedge \ n = n \\ \epsilon \equiv \epsilon | i,j \geq 0 \ \wedge \ n = 1 \ \wedge \ n = 1 \end{array}$$

La règle 6 s'applique puisque $\neg \mathit{IsPrefix}(a_ja_i, \epsilon, \phi) = \neg\bot = \top$ est bien valide.

Règle 1, règle 1, règle 1

$$\begin{array}{l} (i,j \geq 0 \ \wedge \ n = 0) \\ i,j \geq 0 \ \wedge \ n > 0 \\ i,j \geq 0 \ \wedge \ n = 1 \end{array}$$

Cet exemple donne un aperçu des difficultés du problème résolu. La factorisation en présence d'indice et de puissances est une grande source de complexité et engendre de grosses contraintes. Dans l'implantation nous simplifierons les contraintes au cours de l'algorithme afin de couper très tôt des branches contenant des contraintes non satisfiables. Remarquons de plus que la règle 2 de *LFact* pourrait être renforcée ainsi

$$\frac{n, y^pu, \phi \qquad\qquad \text{si } |x| = |y|}{(n+p, u, \phi \ \wedge \ x \equiv y \ \wedge \ \neg p = 0); (n, u, \phi \ \wedge \ p = 0); \mathit{Result}(n, y^pu, \phi \ \wedge \ \neg x \equiv y \ \wedge \ \neg p = 0)}$$

ce qui se traduirait par une augmentation de la taille de la contrainte. Dans un souci d'efficacité, nous veillerons toujours à construire des règles qui n'engendrent pas des contraintes inutilement grosses.

Nous avons maintenant les outils nécessaires pour énoncer le pendant du théorème 8 page 39.

Lemme 22 *Soient deux mots paramétrés avec indices et exposants $u|\phi$ et $v|\psi$. Soit χ une forme résolue de la formule de similitude $u \equiv v|\phi \ \wedge \ \psi$. Les deux mots sont localement similaires si et seulement si*

- *$\phi \ \wedge \ \psi \Rightarrow \chi$ est une formule valide et*
- *$\phi \Leftrightarrow \psi$ est une formule valide.*

Preuve

- Supposons que les mots $u|\phi$ et $v|\psi$ sont localement similaires. Fixons ν une interprétation des paramètres et μ une interprétation des variables.
  - Si $\nu(\mu(\phi)) \ \wedge \ \nu(\mu(\psi))$ est fausse, la prémisse de la première formule est fausse donc l'implication est vraie. Sinon, en appliquant la définition de la similitude locale, $[\![u|\phi]\!]_\nu(\mu) = [\![v|\psi]\!]_\nu(\mu)$ c'est-à-dire $\nu(\mu(\chi))$ est vraie.

  - $\phi \Leftrightarrow \psi$ est une formule valide par définition de la similitude locale.
- Réciproquement supposons que $\phi \wedge \psi \Rightarrow \chi$ et $\phi \Leftrightarrow \psi$ sont des formules valides. Si $\nu(\mu(\phi \wedge \psi))$ est vraie alors $\nu(\mu(\chi))$ est vraie. Donc d'après le théorème 21 page 77

$$[\![u|\phi]\!]_\nu(\mu) = [\![v|\psi]\!]_\nu(\mu)$$

Nous avons bien établi la réciproque.

◇

Notons que χ contient la contrainte $\phi \wedge \psi$. Nous aurions pu résoudre simplement la formule de similitude $u \equiv v|\top$ et énoncer le même théorème. Nous choisissons de garder $\phi \wedge \psi$ afin de simplifier la résolution de la formule de similitude. En effet, cela permet de limiter le nombre de fois que *IsPrefix* sera satisfiable et donc d'empêcher un trop grand nombre d'applications de *LFact*.

5.2 Équations - Règles - Réécriture

5.2.1 Équations de mots

Définition et sémantique des équations

Définition 81 *Une* équation *entre mots paramétrés avec indices et exposants est un couple* $(\{u; v\}, \psi)$ *noté*

$$(u = v)|\psi$$

où
- ψ *est une formule de la logique de Presburger et*
- $u|\psi$ *et* $v|\psi$ *sont deux mots paramétrés avec indices et exposants tels que le mot paramétré avec indices et exposants* $uv|\psi$ *est réduit.*

Exemple 82 $a(ba)^n = (ab)^n a|n > 0$ est une équation que nous étudierons plus en détails dans cette section.

Nous définissons aussi une notion technique qui sera importante dans la suite.

Définition 83 *Un* couple contraint *est un couple* $((u, v), \psi)$ *noté*

$$(u, v)|\psi \text{ ou } u, v|\psi$$

avec
- ψ *est une formule de la logique de Presburger et*
- $u|\psi$ *et* $v|\psi$ *sont deux mots paramétrés tels que le mot paramétré* $uv|\psi$ *est réduit.*

Nous définissons maintenant l'interprétation d'une équation et d'un couple contraint.

Définition 84 *Soient $\nu : \mathcal{N} \to \mathbb{Z}$ une interprétation des paramètres. La ν-interprétation du couple contraint $u, v|\phi$ est définie comme l'ensemble des couples formés des instanciations simultanées de u et v de façon compatible avec la contrainte, soit :*

$$[\![u, v|\phi]\!]_\nu = \bigcup_{\substack{\mu : I \to \mathbb{Z} \\ \nu(\mu(\phi)) \text{ vraie}}} \{(\nu(\mu(u)), \nu(\mu(v)))\}$$

Soit $u = v|\phi$ une équation paramétrée avec exposants, la ν-interprétation de $u = v|\phi$ est définie par :

$$[\![u = v|\phi]\!]_\nu = \bigcup_{(u',v') \in [\![u,v|\phi]\!]_\nu} \{u' = v'\}$$

La ν-interprétation d'une équation est donc un ensemble d'équations sur la ν-interprétation de la signature. Cet ensemble peut être vide, fini ou infini comme le montrent les exemples suivants.

Exemple 85
1. $[\![a_1^n = a_2^n | n < 0 \ \wedge \ n > 0]\!]_\nu = \emptyset$ pour tout ν.
2. $[\![a_i^i = a_i^{i+1} | 0 < i < n]\!]_{n \mapsto 3} = \{a_1 = a_1 a_1; a_2 a_2 = a_2 a_2 a_2\}$
3. $$[\![a_1^i = a_i^n | 0 < n \ \wedge \ i > 0]\!]_{n \mapsto 2} = \{a_1 = a_1 a_1; a_1 a_1 = a_2 a_2; a_1 a_1 a_1 = a_3 a_3; \cdots\}$$

Décomposition d'un mot paramétré avec exposants

Dans cette section, nous allons définir les notions de préfixes, de facteurs et de suffixes d'un mot paramétré avec exposants. En effet, contrairement au cas sans exposant, la notion de position dans un mot avec exposants dépend fortement de la valeur des paramètres et des variables. Par exemple, dans le mot $a^N b a^N$, la position du b est N. Par contre, la position du a à droite du b est $N + 1$ si $N > 0$ et n'est pas définie sinon. Nous allons voir que plutôt que de continuer à représenter les positions avec des expressions arithmétiques, nous allons directement énumérer les facteurs, à toutes les positions, mais de façon finie. Les contraintes vont permettre d'exprimer les différents découpages possibles dans un mot avec des exposants.

Définition 86 *Soit $u|\phi$ un mot contraint avec exposant. On peut toujours l'écrire $U_1^{N_1} \ldots U_k^{N_k} | \psi$ où pour tout j, N_j est soit une expression soit l'entier* 1*. La* décomposition *de $u|\phi$ est l'ensemble* fini *$Dec(u|\phi)$ des triplets de mots contraints $(pre, sub, suf)|\psi'$*

défini par :

$$
\begin{aligned}
&Dec(u \mid \phi) = \\
&\bigcup_{\substack{0 \le i < j \le k+1 \\ v_1.v_2 = U_i \\ w_1.w_2 = U_j}}
\left\{
\begin{array}{l}
\left(U_1^{N_1} \dots U_i^{p}.v_1, v_2.U_i^{q} \dots U_j^{p'}.w_1, w_2.U_j^{q'} \dots U_k^{N_k}\right) \mid \phi \ \wedge \ 1+p+q = N_i \ \wedge \ 1+p'+q' = N_j; \\
\left(U_1^{N_1} \dots U_i^{p}, U_i^{q} \dots U_j^{p'}.w_1, w_2.U_j^{q'} \dots U_k^{N_k}\right) \mid \phi \ \wedge \ p+q = N_i \ \wedge \ 1+p'+q' = N_j; \\
\left(U_1^{N_1} \dots U_i^{p}, U_i^{q} \dots U_j^{p'}, U_j^{q'} \dots U_k^{N_k}\right) \mid \phi \ \wedge \ p+q = N_i \ \wedge \ p'+q' = N_j; \\
\left(U_1^{N_1} \dots U_i^{p}.v_1, v_2.U_i^{q} \dots U_j^{p'}, U_j^{q'} \dots U_k^{N_k}\right) \mid \phi \ \wedge \ 1+p+q = N_i \ \wedge \ p'+q' = N_j;
\end{array}
\right\} \\
&\bigcup_{\substack{0 \le i \le k+1 \\ v_1.v_2 = U_i \\ w_1.w_2 = U_i}}
\left\{
\begin{array}{l}
\left(U_1^{N_1} \dots U_i^{p}.v_1, v_2 U_i^{q}.w_1, w_2.U_i^{r} \dots U_k^{N_k} \mid \phi \ \wedge \ 2+p+q+r = N_i\right); \\
\left(U_1^{N_1} \dots U_i^{p}, U_i^{q}.w_1, w_2.U_i^{r} \dots U_k^{N_k} \mid \phi \ \wedge \ 1+p+q+r = N_i\right); \\
\left(U_1^{N_1} \dots U_i^{p}.v_1, v_2.U_i^{q}, U_i^{r} \dots U_k^{N_k} \mid \phi \ \wedge \ 1+p+q+r = N_i\right); \\
\left(U_1^{N_1} \dots U_i^{p}, U_i^{q}, U_i^{r} \dots U_k^{N_k} \mid \phi \ \wedge \ p+q+r = N_i\right);
\end{array}
\right\} \\
&\bigcup_{\substack{0 \le i \le k+1 \\ v_1.v_2.v_3 = U_i}}
\left\{ \left(U_1^{N_1} \dots U_i^{p} v_1, v_2, v_3 U_i^{q} \dots U_k^{N_k} \mid \phi \ \wedge \ p+q = N_i\right); \right\}
\end{aligned}
$$

Dans cette définition, on considère que U_0 et U_{k+1} désignent le mot vide ϵ et que p, q, r, p', q' sont des nouvelles variables. De plus nous avons omis les contraintes $p, q, r, p', q' \geq 0$ afin de ne pas alourdir encore la notation.

La décomposition d'un mot désigne toutes les différentes façons de le découper en un préfixe, un facteur et un suffixe. Le premier groupe de triplets désigne les découpages obtenus en coupant dans deux parties disjointes du mot ($U_i^{N_i}$ et $U_j^{N_j}$ avec $i \neq j$), alors que les deux secondes permettent de couvrir le cas ou l'on découpe une même partie du mot ($U_i^{N_i}$) en deux endroits. On note que certains découpages renforcent la contrainte du mot. En effet, il est clair que suivant la valeur des exposants certains facteurs peuvent ou non exister. Ainsi ba^nb admet bab comme facteur seulement si n vaut 1. Nous verrons un peu plus loin que nous ne nous intéresserons qu'aux découpages qui ne renforcent pas la contrainte afin de pouvoir raisonner uniformément sur les facteurs d'un mot. Notons que le rôle des variables p, q, p', q' introduites dans la décomposition est un peu particulier. En effet, pour distinguer une décomposition particulière, on peut instancier ces variables. La définition du filtrage exploitera cette possibilité.

Exemple 87 Étudions le mot $b^n | n \geq 1$. La définition s'applique avec $U_1 = b$ et $N_1 = n$.

Notons que les v_1, v_2, w_1, w_2, v_3 sont tous soit ϵ soit b.

$$
\begin{array}{ll}
Dec(b^n \mid n \geq 1) = \{ & (\epsilon, \epsilon, b^n) \mid n \geq 1; \\
& (\epsilon, b^n, \epsilon) \mid n \geq 1; \\
& (b^n, \epsilon, \epsilon) \mid n \geq 1; \\
& (\epsilon, b^p.b, b^q) \mid 1+p+q = n \ \wedge \ n \geq 1; \\
& (\epsilon, b^p, b.b^q) \mid 1+p+q = n \ \wedge \ n \geq 1; \\
& (b^p.b, b^q, \epsilon) \mid 1+p+q = n \ \wedge \ n \geq 1; \\
& (b^p, b.b^q, \epsilon) \mid 1+p+q = n \ \wedge \ n \geq 1; \\
& (b^p, bb^q, bb^r) \mid 2+p+q+r = n \ \wedge \ n \geq 1; \\
& (b^pb, b^q, bb^r) \mid 2+p+q+r = n \ \wedge \ n \geq 1; \\
& (b^p, bb^qb, b^r) \mid 2+p+q+r = n \ \wedge \ n \geq 1; \\
& (b^pb, b^qb, b^r) \mid 2+p+q+r = n \ \wedge \ n \geq 1; \\
& (b^p, b^q, bb^r) \mid 1+p+q+r = n \ \wedge \ n \geq 1; \\
& (b^p, b^qb, b^r) \mid 1+p+q+r = n \ \wedge \ n \geq 1; \\
& (b^p, bb^q, b^r) \mid 1+p+q+r = n \ \wedge \ n \geq 1; \\
& (b^pb, b^q, b^r) \mid 1+p+q+r = n \ \wedge \ n \geq 1; \\
\} &
\end{array}
$$

Remarquons encore que les contraintes de positivité des nouvelles variables ne sont pas écrites pour simplifier la notation, mais doivent être vraiment présentes.

Notons que le cas $n = 0$ est insatisfiable.

Proposition 16 *Soit $u|\psi$ un mot contraint avec indices et exposants et soit D sa décomposition. Pour*

- *toute interprétation $\nu : \mathcal{N} \to N$,*
- *toute interprétation $\mu : Var(u|\phi) \to \mathbb{Z}$,*
- *tous mots v_1, v_2 et v_3 tels que $v_1.v_2.v_3 \in [\![u|\psi]\!]_\nu(\mu)$,*

il existe

- *un triplet contraint $(pre, sub, suf)|\zeta \in D$ et*
- *une extension de μ, encore notée $\mu : I \to \mathbb{Z}$*

tels que

- *$v_1 = [\![pre|\zeta]\!]_\nu(\mu)$,*
- *$v_2 = [\![sub|\zeta]\!]_\nu(\mu)$ et*
- *$v_3 = [\![suf|\zeta]\!]_\nu(\mu)$.*

Preuve Par cas, la preuve est claire. $\diamond$

Proposition 17 *Soit $u|\psi$ un mot paramétré avec indices et exposants. Soit $(pre, sub, suf, \zeta) \in Dec(u|\psi)$. Pour toute interprétation $\nu : \mathcal{N} \to \mathbb{Z}$ et $\mu : I \to \mathbb{Z}$,*

si $\nu(\mu(\zeta)) = \top$ alors $\nu(\mu(\psi)) = \top$ et

$$\nu(\mu(pre)).\nu(\mu(sub)).\nu(\mu(suf)) = \nu(\mu(u))$$

Preuve Par cas sur la définition de *Dec*, la preuve est évidente. $\diamond$

Nous avons donc un algorithme capable d'énumérer de façon finie les facteurs d'un mot contraint avec exposants. Dans la suite, nous allons nous restreindre à une partie de ces décompositions, celles qui ne renforcent pas la contrainte.

Définition 88 *Nous notons $UDec(u|\psi)$ l'ensemble des* décompositions uniformes *défini par*

$$UDec(u|\psi) = \{(pre, sub, suf, \zeta) \in Dec(u|\psi) \ / \ \psi \Leftrightarrow (\exists (Var(\zeta) \setminus Var(u|\psi)) \ \zeta)\}$$

Un élément de $UDec(u|\phi)$ sera appelé position *dans le mot $u|\psi$*

Exemple 89 Reprenons l'exemple 87 page précédente $(b^n|n \geq 1)$ en supprimant les décompositions non uniformes et on obtient toutes les façons de partager notre mot indépendantes de la valeurs des variables et des paramètres.

$$\begin{array}{ll} UDec(b^n \mid n \geq 1) = \{ & (\epsilon, \epsilon, b^n) \mid n \geq 1; \\ & (\epsilon, b^n, \epsilon) \mid n \geq 1; \\ & (b^n, \epsilon, \epsilon) \mid n \geq 1; \\ & (\epsilon, b^p.b, b^q) \mid 1+p+q = n \ \wedge \ n \geq 1; \\ & (\epsilon, b^p, b.b^q) \mid 1+p+q = n \ \wedge \ n \geq 1; \\ & (b^p.b, b^q, \epsilon) \mid 1+p+q = n \ \wedge \ n \geq 1; \\ & (b^p, b.b^q, \epsilon) \mid 1+p+q = n \ \wedge \ n \geq 1; \\ & (b^p, b^q, bb^r) \mid 1+q+r = n \ \wedge \ n \geq 1; \\ & (b^p, b^q b, b^r) \mid 1+p+q+r = n \ \wedge \ n \geq 1; \\ & (b^p, bb^q, b^r) \mid 1+p+q+r = n \ \wedge \ n \geq 1; \\ & (b^p b, b^q, b^r) \mid 1+p+q+r = n \ \wedge \ n \geq 1; \\ \} & \end{array}$$

Nous avons défini dans cette section la notion de décomposition uniforme d'un mot avec exposants et indices. On nommera respectivement préfixe, facteur, et suffixe uniforme les trois éléments d'un triplet de la décomposition.

Étape équationelle

Nous allons définir d'abord la notion de filtrage.

Définition 90 *Le mot $u|\psi$ est dit* convertible *par l'équation $l = r|\xi$ s'il existe*
- *une substitution $\sigma : Var(l = r|\xi) \rightarrow Expr(I, \mathcal{N}, \mathbb{Z})$ et*
- *un triplet $(p, sub, s, \zeta) \in UDec(u|\psi)$*

tels qu'avec les notations :
- *$\aleph$ une forme résolue de $sub \equiv \sigma(l)|\zeta \ \wedge \ \sigma(\xi)$,*
- *$\{x_1, \ldots, x_k\} = Var(\sigma(\xi)) \setminus Var(\zeta)$ et*
- *$\{y_1, \ldots, y_{k'}\} = Var(\zeta) \setminus Var(\psi)$*

la formule

$$\exists x_1 \ldots \exists x_k \exists y_1 \ldots \exists y_{k'}\ (\zeta \Rightarrow \aleph)$$

est valide.

Dans ce cas, nous dirons que (p, sub, s, ζ) *est une* position de filtrage *du mot pour l'équation, que* σ *est une* substitution de filtrage *et que* $\aleph$ *est une contrainte de filtrage. Nous dirons aussi que le mot* $u|\psi$ *est* filtré *par l'équation* $l = r|\xi$.

Nous pouvons remarquer que dans le cas où il n'y a pas d'exposant, la définition coïncide avec celle donnée pour les mots avec indices sans exposants.

Proposition 18 *Si* $u|\psi$ *est* convertible *par l'équation* $l = r|\xi$ *alors pour toute instanciation* $\nu : \mathcal{N} \to \mathbb{Z}$ *et* $\mu : I \to \mathbb{Z}$, *telles que* $\nu(\mu(\psi)) = \top$, *le mot de* $[\![u|\psi]\!]_\nu(\mu)$ *est filtré par une des équation de* $[\![l = r|\psi]\!]_\nu$.

Preuve En combinant le théorème sur la décomposition et sur la formule de similitude et en passant à l'interprétation, la preuve est claire. ◇

Définition 91 *Le mot paramétré* $u|\psi$ *est convertible en un mot paramétré* $v|\chi$ *par l'équation* $l = r|\xi$ *à la position* $(p, l', s, \zeta) \in UDec(u|\psi)$ *et on note*

$$u|\psi \xleftrightarrow[(p,l',s,\zeta)]{l=r|\xi} v|\chi$$

si et seulement si

il existe une substitution σ *telle que*

1. $u|\psi$ *est filtré par* $l = r|\xi$ *à la position* (p, l', s, ζ) *avec la substitution de filtrage* σ *et la formule de filtrage* $\aleph$,
2. $p\sigma(r)s|\aleph$ *et* $v|\chi$ *sont localement similaires.*

Exemple 92 Montrons que

$$a_k a_l^l | k = l \wedge l \geq 1 \xleftrightarrow[\epsilon, a_k a_l^p, a_l a_l^q, p+q+1=l \ \wedge\ k=l \ \wedge\ l\geq 1]{a_i^i = a_j | j=i-1 \ \wedge\ i \geq 1} a_{j'} a_l a_l^q \left|\ \begin{array}{l} l = i' \ \wedge\ q+1 = i' \ \wedge\ j' = i'-1 \\ \wedge\ i' \geq 1 \ \wedge\ p+q+1 = l \ \wedge\ k = l \\ \wedge\ k = i' \ \wedge\ l \geq 1 \ \wedge\ p, q \geq 0 \end{array}\right.$$

Nous fixons la substitution σ telle que

$$\begin{aligned} \sigma(i) &= i' \\ \sigma(j) &= j' \end{aligned}$$

- En résolvant la formule de similitude

  $$a_k a_l^q \equiv a_{i'}^{i'} | j' = i' - 1 \ \wedge\ i' \geq 1 \ \wedge\ p+q+1 = l \ \wedge\ k = l \ \wedge\ l \geq 1$$

  nous trouvons la formule

  $$\aleph \ = \ ((k = i' \wedge l = i' \wedge q{+}1 = i') \wedge j' = i'{-}1 \wedge i' \geq 1 \wedge p{+}q{+}1 = l \wedge k = l \wedge l \geq 1)$$

Comme
$$\exists p \exists q \exists i' \exists j' p+q+1=l \ \wedge\ k=l \ \wedge\ l \geq 1 \Rightarrow \aleph$$
est valide, nous avons bien établi la convertibilité de notre mot.

– Le mot
$$\epsilon.a_{j'}.a_l a_l^q \left| \begin{array}{l} \ \ l=i' \ \wedge\ q+1=i' \ \wedge\ j'=i'-1 \\ \wedge\ i' \geq 1 \ \wedge\ p+q+1=l \ \wedge\ k=l \\ \wedge\ k=i' \ \wedge\ l \geq 1 \ \wedge\ p,q \geq 0 \end{array} \right.$$
est bien localement similaire au mot produit puisqu'il lui est exactement égal syntaxiquement.
Notons que le mot produit peut aussi se noter
$$a_{l-1} a_l^{q+1} \left| \begin{array}{l} \ \ l=i' \ \wedge\ q+1=i' \ \wedge\ j'=i'-1 \\ \wedge\ i' \geq 1 \ \wedge\ p+q+1=l \ \wedge\ k=l \\ \wedge\ k=i' \ \wedge\ l \geq 1 \ \wedge\ p,q \geq 0 \end{array} \right.$$
qui vérifie bien la condition de production.

Proposition 19 (Sémantique d'une étape équationelle) *Si*
$$u|\psi \xleftrightarrow[p,l',s,\zeta]{l=r|\xi} v|\chi$$
est une étape équationelle alors pour tout $\nu : \mathcal{N} \to \mathbb{Z}$ *et tout* $\mu : I \to \mathbb{Z}$ *tels que* $[\![u|\psi]\!]_\nu(\mu)$ *et* $[\![v|\chi]\!]_\nu(\mu)$ *sont définis,*
$$[\![u|\psi]\!]_\nu(\mu) \xleftrightarrow[|p|+1]{[\![l=r|\xi]\!]_\nu} [\![v|\chi]\!]_\nu(\mu)$$

Preuve Fixons $\nu : \mathcal{N} \to \mathbb{Z}$ et $\mu : I \to \mathbb{Z}$ tels que $u_0 = [\![u|\psi]\!]_\nu(\mu)$ et $v_0 = [\![v|\chi]\!]_\nu(\mu)$ sont définis, c'est-à-dire tels que $\nu(\mu(\psi))$ et $\nu(\mu(\chi))$ sont vraies.

On note σ la substitution de vérifiant les conditions 1) et 2).

Nous avons $\chi \Leftrightarrow \aleph$ et $\aleph \Rightarrow \sigma(\xi)$ par définition de $\aleph$. Donc comme $\nu(\mu(\chi))$ est vraie et $\nu(\mu(\sigma(\xi)))$ est aussi vraie. Nous avons donc
$$(\nu(\mu(\sigma(l))) = \nu(\mu(\sigma(r)))) \in [\![l=r|\xi]\!]_\nu$$
De plus, on a bien $\nu(\mu(\sigma(l))) = \nu(\mu(l'))$ facteur de $\nu(\mu(u))$ à la position $|p|+1$ et $\nu(\mu(\sigma(r)))$ facteur de $\nu(\mu(v))$ à cette même position. En ajoutant le contexte on a bien la propriété voulue. ◇

Comme cette proposition est une généralisation de la proposition 5 page 41, sa réciproque est fausse pour les mêmes raisons.

Relation équationelle

Définition 93 *Un* système équationel paramétré *est un ensemble fini d'équations paramétrées. Deux mots paramétrés sont dits convertibles par ce système si et seulement si il existe une séquence finie d'étapes équationelles entre eux.*

Nous remarquons que, contrairement à la réécriture avec indices et sans exposant, la relation de convertibilité n'est pas symétrique. Nous donnons un exemple de ce comportement dans la signature finie $\Sigma = \{a; b; c\}$, sans indice afin de renforcer la lisibilité du problème.

Exemple 94 Soient $u = ba(cab)^n|n \geq 0$ un mot, $ba = ab$ une équation. Clairement, nous avons l'étape équationelle

$$ba(cab)^n|n \geq 0 \xleftrightarrow[\epsilon,ba,(cab)^n,n\geq 0]{ba=ab} ab(cab)^n|n \geq 0$$

Or le mot $ab(cab)^n|n \geq 0$ est localement similaire à $a(bca)^n b|n \geq 0$. Nous pouvons donc écrire :

$$ba(cab)^n|n \geq 0 \xleftrightarrow[\epsilon,ba,(cab)^n,n\geq 0]{ba=ab} a(bca)^n b|n \geq 0$$

Or $a(bca)^n b|n \geq 0$ n'a pas de décomposition uniforme qui permet d'appliquer $ab = ba$. Donc la relation équationelle n'est pas réflexive.

Suite à ce handicap de la relation équationelle, nous pouvons essayer de restreindre la définition de la réécriture. Ainsi si nous essayons d'empêcher le raisonnement modulo la similitude locale, nous pouvons choisir que le mot produit est syntaxiquement égal à $p.\sigma(r).s$ (en reprenant les notations de la définition. Pour autant, cela ne permet pas d'obtenir une relation réflexive comme le montre l'exemple suivant :

Exemple 95 Posons $u = ab(cab)^n|n \geq 1$. Considérons cette fois l'équation $abc = bc$. Nous avons clairement l'étape équationelle suivante :

$$ab(cab)^n|n \geq 1 \xleftrightarrow[\epsilon,abc,ab(cab)^{n-1},n\geq 1]{abc=bc} \epsilon.bc.ba(cab)^{n-1}|n \geq 1$$

Nous avons "dans l'autre sens" :

$$bcba(cab)^{n-1}|n \geq 1 \xleftrightarrow[\epsilon,bc,ba(cab)^{n-1},n\geq 1]{bc=abc} \epsilon.abc.ba(cab)^{n-1}|n \geq 1$$

Or $abc.ba(cab)^{n-1}|n \geq 1$ n'est pas égal syntaxiquement à u. Nous n'avons donc toujours pas la réflexivité.

Nous montrons maintenant que la convertibilité est stable par mise sous contexte, par renforcement de la contrainte et par instanciation. Cela nous permet d'affirmer que la clôture réflexive de la relation de convertibilité est une congruence.

Lemme 23 *Si*

$$u|\psi \xleftrightarrow[p,l',s,\zeta]{l=r|\xi} v|\chi$$

est une étape équationelle alors pour toute formule ϕ*,*

$$u|\psi \ \wedge \ \phi \xleftrightarrow[p,l',s,\zeta \ \wedge \ \phi]{l=r|\xi} v|\chi \ \wedge \ \phi$$

est une étape équationelle.

Preuve $(p, l', s, \zeta \ \wedge \ \phi)$ est bien un élément de $UDec(u|\psi \ \wedge \ \phi)$ quitte à renommer les nouvelles variables de ζ. En notant $\aleph$ la forme résolue de $l' \equiv \sigma(l)|\zeta \ \wedge \ \sigma(\xi)$, nous avons clairement $\aleph \ \wedge \ \phi$ est une forme résolue de $l' \equiv \sigma(l)|\zeta \ \wedge \ \phi \ \wedge \ \sigma(\xi)$. En utilisant les notations de la définition, nous posons

$$\{x_1, \ldots, x_k\} = Var(\sigma(\xi)) \setminus Var(\zeta)$$

et

$$\{y_1, \ldots, y_{k'}\} = Var(\zeta) \setminus Var(\psi)$$

et nous avons

$$\exists x_1 \ldots \exists x_k \exists y_1 \ldots \exists y_{k'} \ (\zeta \Rightarrow \aleph)$$

valide. Nous pouvons en déduire que

$$\exists x_1 \ldots \exists x_k \exists y_1 \ldots \exists y_{k'} \ (\zeta \ \wedge \ \phi \Rightarrow \aleph \ \wedge \ \phi)$$

est valide. Notons bien que grâce au renommage les variables de ϕ ne sont pas capturée dans la formule précédente.

De plus comme $p\sigma(r)s|\aleph$ est localement similaire à $v|\chi$, nous avons aussi $p\sigma(r)s|\aleph \wedge \phi$ est localement similaire à $v|\chi \wedge \phi$ puisque la similitude locale est stable par renforcement de la contrainte.

Nous avons bien démontré le lemme. ◇

Lemme 24 *Si*

$$u|\psi \xleftrightarrow[p,l',s,\zeta]{l=r|\xi} v|\chi$$

est une étape équationelle alors pour tout contexte justement contraint $C[\diamond]|\phi = (p', s', \phi)$*,*

$$C[u]|\psi \ \wedge \ \phi \xleftrightarrow[p'p,l',ss',\zeta \ \wedge \ \phi]{l=r|\xi} C[v]|\chi \ \wedge \ \phi$$

est une étape équationelle.

Preuve En appliquant le lemme 23 nous avons déjà

$$u|\psi \ \wedge \ \phi \xleftrightarrow[p,l',s,\zeta \ \wedge \ \phi]{l=r|\xi} v|\chi \ \wedge \ \phi$$

avec la formule de filtrage $\aleph$. De plus nous constatons que $p'p, l', ss', \zeta \wedge \phi$ est une position uniforme de $C[u]|\psi \wedge \phi$. Comme $p\sigma(r)s|\aleph$ est localement similaire à $v|\chi \wedge \phi$, nous avons aussi $C[p\sigma(r)s]|\aleph$ localement similaire à $C[v]|\chi \wedge \phi$. ◇

Lemme 25 *Si*

$$u|\psi \xleftrightarrow[p,l',s,\zeta]{l=r|\xi} v|\chi$$

est une étape équationelle alors pour toute instanciation ρ telle que $\rho(u|\psi)$ et $\rho(v|\chi)$ sont justement contraints,

$$\rho(u|\psi) \xleftrightarrow[\rho(p),\rho(l'),\rho(s),\rho(\zeta)]{l=r|\xi} \rho(v|\chi)$$

est une étape équationelle

Preuve Comme la validité d'une formule est stable par instanciation, une application immédiate de la définition établi le lemme. $\diamond$

5.2.2 Règles de réécriture

Définition

Nous introduisons ensuite la notion de *règle de réécriture sur les mots paramétrés avec indices et exposants.*

Définition 96 *Une* règle sur les mots paramétrés avec indices et exposants *est un couple $((u, v), \psi)$ dont la première composante est un couple tel que*

- *ψ est une formule de la logique de Presburger et*
- *$u|\psi$ et $v|\psi$ sont deux mots tels que $uv|\psi$ est réduit.*

u est appelé membre gauche *de la règle, v est appelé* membre droit *de la règle et ψ est appelé* contrainte *de la règle. Cette règle sera notée*

$$u \to v|\psi$$

Un système de réécriture sur les mots paramétrés *est défini comme un ensemble fini de règles et est noté :*

$$R = \left\{ \begin{array}{ccc} u_1 \to v_1 & | & \psi_1; \\ & \vdots & \\ u_k \to v_k & | & \psi_k \end{array} \right\}$$

La sémantique des règles est très similaire à celle des équations.

Définition 97 *Pour une interprétation des paramètres fixée ν, la ν-*interprétation *d'une règle est définie par*

$$[\![u \to v|\phi]\!]_\nu = \bigcup_{(u',v') \in [\![u,v|\phi]\!]_\nu} \{u' \to v'\}$$

La ν-interprétation d'une règle paramétrée est donc un ensemble de règles entre des mots sur la ν-interprétation de la signature.

La ν-interprétation d'un système de réécriture est l'union des ν-interprétations de chaque règle du système.

Ainsi l'interprétation d'une règle paramétrée est exactement un système de réécriture de mots classique (éventuellement infini).

Exemple 98 Dans $\Sigma_2 = \{a_i | 1 \leq i \leq N-1\}_{\{N\}}$,

$$[\![a_i^i \to a_j | 0 \leq i < j \leq P]\!]_{P \mapsto 2} = \{\epsilon \to a_1; \epsilon \to a_2; a_1 \to a_2\}$$

$$[\![(u_i u_j)^N \to u_j u_i | i - j \geq 2]\!]_{N \to 5} = \left\{ \begin{array}{lcl} u_3u_1u_3u_1u_3u_1u_3u_1u_3u_1 & \to & u_1u_3; \\ u_4u_1u_4u_1u_4u_1u_4u_1u_4u_1 & \to & u_1u_4; \\ u_4u_2u_4u_2u_4u_2u_4u_2u_4u_2 & \to & u_2u_4 \end{array} \right\}$$

Nous allons maintenant définir l'exécution de la réécriture de mots paramétrés.

Réduction

Définition 99 *Le mot paramétré $u|\psi$ se réduit en un mot paramétré $v|\chi$ par la règle $l \to r|\xi$ à la position p, l', s, ζ et on note*

$$u|\psi \xrightarrow[p,l',s,\zeta]{l \to r|\xi} v|\chi$$

si et seulement si

$$u|\psi \xleftrightarrow[p,l',s,\zeta]{l = r|\xi} v|\chi$$

Sémantique

Notons qu'une étape de réduction est une étape équationelle par définition. Ainsi nous pouvons appliquer le théorème 19 page 87 directement pour énoncer :

Proposition 20 *Si*

$$u|\psi \xrightarrow[p,l',s,\zeta]{l \to r|\xi} v|\chi$$

est une étape de réécriture alors pour tout $\nu : \mathcal{N} \to \mathbb{Z}$ et tout $\mu : I \to \mathbb{Z}$ tels que $[\![u|\psi]\!]_\nu(\mu)$ et $[\![v|\chi]\!]_\nu(\mu)$ sont définis,

$$[\![u|\psi]\!]_\nu(\mu) \xrightarrow[p]{[\![l \to r|\xi]\!]_\nu} [\![v|\chi]\!]_\nu(\mu)$$

5.2.3 Algorithme de filtrage

L'algorithme de filtrage permet de décider si une règle paramétrée peut s'appliquer sur un mot paramétré avec indices et exposants.

Algorithme 7 (Filtrage)

Entrée : *$u|\psi$ et $l \to r|\xi$*

Résultat : *($\aleph$ une contrainte, $\pi \in UDec(u|\psi)$) ou $\perp$*

Algorithme : *Posons $S = UDec(u|\psi)$*

– *Si $S = \emptyset$ renvoyer $\perp$*

– *Soit* $(p, l', s, \zeta) \in S$
  1. *Soit* $\aleph$ *une forme résolue de* $l' \equiv l|\zeta \ \wedge \ \xi$
  2. *Posons* $\{x_1; \ldots; x_r\} = Var(\xi) \setminus Var(\psi)$. *Si* $\exists x_1 \ldots x_r(\psi \Rightarrow \sigma(\xi))$ *est valide alors on renvoie* $(\aleph, (p, l', s, \zeta))$ *sinon on recommence avec* $S \setminus \{(p, l', s, \zeta)\}$.

Notons que nous renvoyons une contrainte au lieu d'une substitution dans cet algorithme. En effet, la contrainte, comme elle est résolue, définie implicitement une substitution pour les variables.

Lemme 26 *Un mot* $u|\psi$ *est réductible par une règle* $l \to r|\xi$ *si et seulement si* $u|\psi$ *est filtrée par* $l \to r|\xi$.

Preuve C'est une application immédiate de la définition de la réductibilité en notant que la forme résolue de la contrainte permet de construire une substitution des variables. ◇

5.2.4 Exemples de pas de réécriture

Exemple 100
1. $U = a_i|i \geq 0$, $R = a_j \to a_{j+1}|j \geq 0$. Nous montrons avec cet exemple que l'on retrouve le même résultat que dans le cas sans exposants. À la position $(\epsilon, a_i, \epsilon)|i \geq 0$, nous posons $\sigma(j) = j$. On résout la formule de similitude locale $a_i \equiv a_j|i \geq 0 \ \wedge \ j \geq 0$ et on trouve $i = j \ \wedge \ i \geq 0 \ \wedge \ j \geq 0$ Comme

$$\exists j(i \geq 0 \Rightarrow i = j \ \wedge \ i \geq 0 \ \wedge \ j \geq 0)$$

est valide, nous avons bien établi la réductibilité de U par R. Nous avons donc

$$a_i|i \geq 0 \xrightarrow[(\epsilon, a_i, \epsilon)|i \geq 0]{a_j \to a_{j+1}|j \geq 0} \epsilon.a_{j+1}.\epsilon|i = j \ \wedge \ i \geq 0 \ \wedge \ j \geq 0$$

On peut simplifier le mot produit en $a_{j+1}|j \geq 0 \ \wedge \ i = j$

On retrouve bien ce qu'on obtient en appliquant l'algorithme du chapitre précédent dans le cas où il n'y a pas d'exposants.

2. $U = a(ba)^N|N > 0$ et $R = (ab)^i a \to a^N|i > 0 \ \wedge \ N > 0$. On considère la décomposition uniforme de U, $(\epsilon, a(ba)^p, (ba)^q)|N > 0 \ \wedge \ p + q = N$. Posons $\sigma(i) = i$. En résolvant la formule de similitude

$$a(ba)^p \equiv (ab)^i a|N > 0 \ \wedge \ p + q = N \ \wedge \ (p, q \geq 0) \ \wedge \ i > 0 \ \wedge \ N > 0$$

nous obtenons

$$\aleph \ = \ ((i = 0 \wedge p = 0) \vee (i > 0 \wedge i = p) \wedge N > 0 \wedge p + q = N \wedge (p, q \geq 0) \wedge i > 0)$$

que l'on simplifie en

$$\aleph \ = \ (i = p \ \wedge \ N > 0 \ \wedge \ p + q = N \ \wedge \ (p, q \geq 0) \ \wedge \ i > 0)$$

Comme

$$\exists p \exists q \exists i (N > 0 \ \wedge \ p + q = N) \Rightarrow \aleph$$

est valide, nous pouvons écrire la réduction

$$a(ba)^N | N > 0 \xrightarrow[(\epsilon, a(ba)^p, (ba)^q) | N>0 \ \wedge \ p+q=N]{(ab)^i a \to a^N | i>0 \ \wedge \ N>0} a^N.(ba)^q \left| \begin{array}{l} i > 0 \ \wedge \ N > 0 \ \wedge \ p + q = N \\ \wedge \ i = p \ \wedge \ (p, q \geq 0) \end{array} \right.$$

3. $U = a_i^j | i > 0 \ \wedge \ j > 0$, $R = a_k \to a_0 | k \geq 0$. On considère la décomposition uniforme de U,

$$(\epsilon, a_i^p a_i, a_i^q) | i > 0 \ \wedge \ j > 0 \ \wedge \ 1 + p + q = j$$

Posons $\sigma(k) = k$. La formule de similitude

$$a_k = a_i^p a_i | i, j > 0 \ \wedge \ 1 + p + q = j \ \wedge \ k \geq 0$$

se résout en

$$\aleph \ = \ i, j > 0 \ \wedge \ 1 + p + q = j \ \wedge \ k \geq 0 \ \wedge \ p = 0 \ \wedge \ i = k$$

Nous avons bien

$$\exists p \exists q \exists k (i > 0 \ \wedge \ j > 0 \ \wedge \ 1 + p + q = j) \Rightarrow \aleph$$

valide. Nous pouvons donc écrire

$$a_i^j | i > 0 \ \wedge \ j > 0 \xrightarrow[(\epsilon, a_i^p a_i, a_i^q) | i>0 \ \wedge \ j>0 \ \wedge \ 1+p+q=j]{a_k \to a_0 | k \geq 0} a_0 a_i^q | \aleph$$

5.3 Propriétés fondamentales

5.3.1 Relèvement

Nous pouvons noter que la notion de relèvement présentée dans la section 4.3.1 page 48 se généralise directement au cas des mots avec exposants et indices. Nous prenons donc la même définition pour le relèvement.

Lemme 27 *Soit $u|\psi$ un mot avec indice et exposant. Soient ν et μ deux substitutions respectivement des paramètres et des variables. Tout mot de $[\![u|\psi]\!]_\nu(\mu)$ a un relèvement $\overline{u}$ qui est un mot avec indices sans exposants.*

Preuve En appliquant le lemme 14 page 48 un relèvement est un mot avec indices ne contenant pas d'exposants. $\diamond$

Lemme 28 *Soit $l \to r|\xi$ une règle de réécriture paramétrée. Soit ν une interprétation des paramètres. Soient u et v deux mots de $[\![\Sigma]\!]_\nu^*$. Si $u \xrightarrow[p]{[\![l \to r|\xi]\!]_\nu} v$ alors il existe $\overline{u}$ un ν-relèvement de u et $\overline{v}$ un ν-relèvement de v tels que*

$$\overline{u} \xrightarrow{l \to r|\xi} \overline{v}$$

Preuve Donnons quelques notations :
- Soit $\nu : \{N_1; \cdots; N_r\} \to \mathbb{Z}$,
- $u = C_1 a_{n_1} \dots a_{n_q} C_2$ avec $n_j \in \mathbb{Z}$ et C_1, C_2 deux mots tels que $|C_1| = p$.
- $\overline{u} = \overline{C_1} a_{n_1} \dots a_{n_q} \overline{C_2} \mid \ \wedge\ _{1 \leq k \leq r} N_k = \nu(N_k)$ un ν-relèvement de u,
- $v = C_1 a_{m_1} \dots a_{m_{q'}} C_2$ avec $m_j \in \mathbb{Z}$
- $\mu : I \to \mathbb{Z}$, définie sur toutes les variables de la règle $l \to r|\xi$ telle que
$$(a_{n_1} \dots a_{n_q} \to a_{m_1} \dots a_{m_{q'}}) = \nu(\mu(l)) \to \nu(\mu(r))$$
et
$$\nu(\mu(\xi)) = \top$$
Une telle substitution existe puisque dans l'interprétation, $C_1 a_{n_1} \dots a_{n_q} C_2$ est réductible en $C_1 a_{m_1} \dots a_{m_{q'}} C_2$.
- $\overline{v} = \overline{C_1} a_{m_1} \dots a_{m_q} \overline{C_2} | \ \wedge\ _{1 \leq k \leq r} N_k = \nu(N_k) \ \wedge\ \mu(\xi)$. $\overline{v}$ est bien un ν-relèvement de v puisque la formule $\wedge\ _{1 \leq k \leq r} N_k = \nu(N_k) \Rightarrow \mu(\xi)$, dont les seuls symboles non instanciés sont les paramètres, est valide.

Montrons que $\overline{u} \xrightarrow{l \to r|\xi} \overline{v}$.

1. Montrons d'abord que $\overline{u}$ est filtré par $l \to r|\xi$:
   (a) Constatons que $(\overline{C_1}, a_{n_1} \dots a_{n_q}, \overline{C_2}, \ \wedge\ _{1 \leq k \leq r} N_k = \nu(N_k)$ est une décomposition uniforme du $\overline{u}$.
   (b) La formule de similitude à calculer est
   $$\aleph = \left(a_{n_1} \dots a_{n_q} \equiv l | \bigwedge_{1 \leq k \leq r} N_k = \nu(N_k) \ \wedge\ \mu(\xi) \right)$$
   Puis en considérant μ comme une substitution des variables de la règle, il suffit de montrer que
   $$(\ \wedge\ _{1 \leq k \leq r} N_k = \nu(N_k)) \Rightarrow \aleph$$
   est valide puisque toutes les variables sont instanciées de façon close. Soit $\nu' : \mathcal{N} \to \mathbb{Z}$. Si $\nu' \neq \nu$ alors la prémisse de l'implication est fausse et on peut conclure. Sinon $\nu' = \nu$, et il faut montrer que la formule close $\nu(\mu(\aleph))$ est vraie.
   - $\nu(\mu(\xi))$ est vraie par hypothèse sur μ
   - $\nu(\mu(\bigwedge_{1 \leq k \leq r} N_k = \nu(N_k)))$ est une tautologie puisque μ ne porte pas sur les paramètres et peut donc être omise de cette formule.
   - $\nu(\mu(a_{n_1} \dots a_{n_q})) = \nu(\mu(l))$ par hypothèse sur ν et μ.

   Nous avons donc établi la validité de
   $$(\ \wedge\ _{1 \leq k \leq r} N_k = \nu(N_k)) \Rightarrow \aleph$$
   (c) Montrons maintenant la condition 2 de la définition 91 page 86. Les deux mots
   $$\overline{C_1} \mu(r) \overline{C_2} | \aleph$$

et

$$\overline{C_1}a_{m_1}\dots a_{m_q}\overline{C_2}| \ \wedge \ _{1\le k\le r}N_k = \nu(N_k) \ \wedge \ \mu(\xi)$$

sont-ils localement similaires ? Nous avons déjà montré que

$$(\wedge \ _{1\le k\le r}N_k = \nu(N_k)) \Rightarrow \aleph$$

Et clairement

$$\aleph \Rightarrow (\wedge \ _{1\le k\le r}N_k = \nu(N_k) \ \wedge \ \mu(\xi))$$

Donc

$$\aleph \Leftrightarrow \ \wedge \ _{1\le k\le r}N_k = \nu(N_k) \ \wedge \ \mu(\xi)$$

est valide. Reste à montrer que

$$\begin{aligned}(\aleph \ \wedge \ \textstyle\bigwedge_{1\le k\le r} N_k = \nu(N_k) \ \wedge \ \mu(\xi)) \Rightarrow \ (\ & \overline{C_1}\mu(r)\overline{C_2} \equiv \overline{C_1}a_{m_1}\dots a_{m_q}\overline{C_2}|\aleph \\ & \wedge \ \textstyle\bigwedge_{1\le k\le r} N_k = \nu(N_k) \ \wedge \ \mu(\xi) \ )\end{aligned}$$

est valide. C'est évident. En appliquant le lemme 22, nous avons bien montré la locale similitude des deux mots.

$\diamond$

5.3.2 Interprétation des preuves paramétrées - Sémantique

Soit E un système équationel paramétré avec indices et exposants. Notons $\leftrightarrow^*_E$ la clôture *symétrique* et transitive de la relation de convertibilité paramétrée. L'ensemble des théorèmes de E est défini par $Th(E) = \{(u, v) \ / \ u \leftrightarrow^*_E v\}$.

Théorème 8 *Soit E un système équationel paramétré avec indices et exposants et $E' = Th(E)$ et ν une interprétation des paramètres. Le diagramme*

$$\begin{array}{ccc} E & \xrightarrow{Th} & E' \\ \nu\downarrow & & \downarrow\nu \\ [\![E]\!]_\nu & \xrightarrow{Th} & [\![E']\!]_\nu \end{array}$$

commute.

Preuve La preuve est identique à celle du théorème 4.3.2 en remarquant bien que nous avons utilisé la clôture symétrique de la relation. $\diamond$

Ce théorème justifie l'intérêt d'étudier les preuves des systèmes paramétrés avec exposants.

Nous pouvons aussi généraliser immédiatement la notion de modèles de la réécriture avec indices ainsi que le théorème de correction et de complétude qui lui est associé.

5.3.3 Confluence locale

Nous généralisons les définitions et théorèmes de la section 4.3.3 page 54.

Définition 101 *Un système paramétré est dit* localement confluent *si pour tous* $u|\phi$, $v_1|\psi_1$ *et* $v_2|\psi_2$ *tels que*

- $u|\phi \xrightarrow{R} v_1|\psi_1$,
- $u|\phi \xrightarrow{R} v_2|\psi_2$ *et*
- $Var(v_1|\psi_1) \cap Var(v_2|\psi_2) \subset Var(u|\phi)$

il existe deux mots $v'_1|\psi'_1$ *et* $v'_2|\psi'_2$ *tels que*

- $Var(v'_1|\psi'_1) \cap Var(v'_2|\psi'_2) \subset Var(u|\phi)$
- $v_1|\psi_1 \overset{R}{\longrightarrow}^* v'_1|\psi'_1$,
- $v_2|\psi_2 \overset{R}{\longrightarrow}^* v'_2|\psi'_2$ *et*
- $\psi'_1 \ \wedge \ \psi'_2 \Rightarrow v'_1 \equiv v'_2|\psi'_1 \ \wedge \ \psi'_2$

Théorème 9 *Soit R un système paramétré localement confluent. Toute ν-interprétation de R est localement confluente.*

Preuve La preuve est identique à la preuve du théorème 16 page 55 puisque la notion de relèvement est la même. $\diamond$

Notion de paire critique

Définition 102 (Paire critique) *Soient deux règles réduites paramétrées avec indices et exposants* $l_1 \to r_1 \mid \phi_1$, $l_2 \to r_2 \mid \phi_2$. *Construisons les deux ensembles finis*

$$D_1 = \{((x_1, y_1)|\phi'_1)|(x_1, \epsilon, y_1)|\phi'_1 \in Dec(l_1|\phi_1)\}$$

et

$$D_2 = \{((x_2, y_2)|\phi'_2)|(x_2, \epsilon, y_2)|\phi'_2 \in Dec(l_2|\phi_2)\}$$

Nous définissons PC_1 *l'ensemble des couple contraints* $(r_1.y_2, x_1.r_2) \mid \aleph$ *tels que*

- $((x_1, y_1)|\phi'_1), ((x_2, y_2)|\phi'_2)) \in D_1 \times D_2$ *et*
- $\aleph$ *est une forme résolue de la formule de similitude* $y_1 \equiv x_2|\phi'_1 \ \wedge \ \phi'_2$.

Nous définissons aussi PC_2 *l'ensemble des couple contraints* $(r_1, x_1.r_2.y_1) \mid \aleph$ *tels qu'il existe* x_2 *vérifiant*

- $((x_1, x_2, y_1)|\phi'_1) \in Dec(l_1|\phi_1)$ *et*
- $\aleph$ *est une forme résolue de la formule de similitude* $l_2 \equiv x_2|\phi'_1 \ \wedge \ \phi'_2$.

Nous définissons PC l''ensemble des paires critique comme

$$PC = \{((a, b)|\aleph) \ / \ ((a, b)|\aleph) \in PC_1 \cup PC_2 \text{ et } \aleph \text{ satisfiable}\}$$

Les éléments de PC sont appelés paire critique *de la règle* $l_1 \to r_1 \mid \phi_1$ *avec la règle* $l_2 \to r_2 \mid \phi_2$.

Une telle paire critique est dite confluente *ou joignable si le mot* $a|\aleph$ *a un réduit localement similaire à un réduit de* $b|\aleph$.

Lemme des paires critiques

Proposition 21 *Si un système paramétré est localement confluent alors toutes ses paires critiques sont confluentes.*

Preuve Une paire critique est bien-sûr un cas particulier d'un pic critique. ◇

Contrairement au théorème 17, la réciproque est fausse. L'exemple suivant montre que le problème vient du manque de stabilité de la notion de décomposition. Cette notion est "trop" syntaxique pour pouvoir capturer la confluence.

Ce résultat négatif ne sonne pas le glas de la réécritures paramétrée avec exposants. En effet, la joignabilité des paires critiques permet tout de même de prouver la confluence locale de toutes les interprétations du système. C'est bien la propriété qui nous intéresse.

Théorème 10 *Soit R un système de réécriture paramétré avec indices et exposants. Si toutes ses paires critiques sont confluentes alors toutes les ν-interprétations de R sont localement confluentes.*

Preuve Soit $\nu : \mathcal{N} \to \mathbb{Z}$ une interprétation des paramètres. Soit R un système de réécriture paramétré. Donnons-nous $(r_1 l'_2, l'_1 r_2)$ une paire critique entre les règles $l_1 \to r_1$ et $l_2 \to r_2$ de $[\![R]\!]_\nu$.

Commençons pas traiter le cas d'une paire critique telle que l'_2 est un suffixe de l_2 et l'_1 est un préfixe de l_1. Notons $u = l_1 l'_2 = l'_1 l_2$. Nous avons $u \to r_1 l'_2$ et $u \to l'_1 r_2$. En appliquant le lemme 28, nous relevons ces étapes de réécriture dans le système paramétré en

$$\overline{u} \xrightarrow{u_1 \to v_1 | \psi_1} \overline{r_1 l'_2}$$

et

$$\overline{u} \xrightarrow{u_2 \to v_2 | \psi_2} \overline{l'_1 r_2}$$

Notons Φ la contrainte de $\overline{u}$. Comme $\overline{u}$ est un relèvement, il ne contient pas de facteurs avec exposants. Ainsi $(\epsilon, \overline{l_1}, \overline{l'_2}, \Phi)$ et $(\overline{l'_1}, \overline{l_2}, \epsilon, \Phi)$ sont des décompositions uniformes de $\overline{u}$. Nous pouvons donc préciser les étapes de réécriture :

$$\overline{u} \xrightarrow[(\epsilon, \overline{l_1}, \overline{l'_2}, \Phi)]{u_1 \to v_1 | \psi_1} \overline{r_1}\overline{l'_2}$$

et

$$\overline{u} \xrightarrow[(\overline{l'_1}, \overline{l_2}, \epsilon, \Phi)]{u_2 \to v_2 | \psi_2} \overline{l'_1}\overline{r_2}$$

Les deux règles $u_1 \to v_1 | \psi_1$ et $u_2 \to v_2 | \psi_2$ (dont on peut supposer les variables distinctes) sont des règles paramétrées de R. Il existe une interprétation μ telle que

- $\nu(\mu(u_1)) = l_1$,
- $\nu(\mu(u_2)) = l_2$,
- $\nu(\mu(v_1)) = r_1$,

- $\nu(\mu(v_2)) = r_2$,
- $\nu(\mu(\psi_1))$ et $\nu(\mu(\psi_2))$ sont vraies.

En appliquant la proposition 16, il existe une extension μ' de μ et des triplets contraints $(x_1, \epsilon, y_1)|\psi'_1$ et $(x_2, \epsilon, y_2)|\psi'_2$ tels que :
- $(x_1, \epsilon, y_1)|\psi'_1 \in Dec(u_1|\psi_1)$ et $[\![x_1|\psi'_1]\!]_\nu(\mu') = l'_1$ puisque $\nu(\mu(u_1))$ a pour préfixe l'_1.
- $(x_2, \epsilon, y_2)|\psi'_1 \in Dec(u_2|\psi_2)$ et $[\![y_2|\psi'_2]\!]_\nu(\mu') = l'_2$ puisque $\nu(\mu(u_2))$ a pour préfixe l'_2.
-
-

Notons $\aleph$ la formule de similitude $y_1 \equiv x_2 \mid \psi'_1 \ \wedge \ \psi'_2$. Elle est satisfiable, car :
- Nous avons :
$$\begin{aligned}\nu(\mu'(x_1y_1y_2)) &= \nu(\mu'(x_1y_1))\nu(\mu'(y_2))\\ &= \nu(\mu'(u_1))\nu(\mu'(y_2))\\ &= l_1l'_2\\ &= l'_1l_2\\ &= \nu(\mu'(x_1))\nu(\mu'(x_2y_2))\\ &= \nu(\mu'(x_1x_2y_2))\end{aligned}$$
Nous en déduisons que $l'_1\nu(\mu'(y_1))l'_2 = l'_1\nu(\mu'(x_2))l'_2$ et donc que $\nu(\mu'(y_1)) = \nu(\mu'(x_2))$.
- De plus $\nu(\mu'(\psi'_1 \ \wedge \ \psi'_2))$ est vraie puisque les interprétation $[\![x_1|\psi'_1]\!]_\nu(\mu') = l'_1$ et $[\![y_2|\psi'_2]\!]_\nu(\mu') = l'_2$ sont bien définies.

Nous en déduisons que le couple contraint
$$(v_1y_2, x_1v_2)|\aleph$$
est une paire critique du système de réécriture paramétré qui provient de PC_1. Par hypothèse, une telle paire critique est joignable. Il existe donc J_1 un réduit de $v_1y_2|\aleph$ localement similaire à un réduit J_2 de $x_1v_2|\aleph$. En appliquant la proposition 19, nous interprétons les étapes de réécriture
$$v_1y_2|\aleph \to^* J_1$$
et
$$x_1v_2|\aleph \to^* J_2$$
par l'interprétation $\nu \circ \mu'$ qui vérifie les conditions d'application de la proposition. Nous obtenons :
$$[\![v_1y_2|\aleph]\!]_\nu(\mu') \to^* [\![J_1]\!]_\nu(\mu')$$
et
$$[\![x_1v_2|\aleph]\!]_\nu(\mu') \to^* [\![J_2]\!]_\nu(\mu')$$
Par définition de la similitude locale, $[\![J_1]\!]_\nu(\mu') = [\![J_2]\!]_\nu(\mu')$. De plus $[\![v_1y_2|\aleph]\!]_\nu(\mu') = r_1l'_2$ et $[\![x_1v_2|\aleph]\!]_\nu(\mu') = l'_1r_2$. Nous avons donc prouvé que la paire critique dans l'interprétation est joignable.

Un raisonnement analogue permet de traiter le cas où la paire critique de l'interprétation provient de PC_2, c'est-à-dire est de la forme $(c_1r_1c_2, r_2)$.

En appliquant le théorème des paires critiques dans l'interprétation, nous pouvons conclure que l'interprétation est localement confluente. $\diamond$

5.3.4 Terminaison

Les propriétés et définitions sont les mêmes que dans le cas avec indices sans exposants.

Définition 103 *Un système paramétré R termine si et seulement si tout mot paramétré t ne peut être réduit qu'un nombre fini de fois par R.*

Proposition 22 *R termine si et seulement si pour tout ν, $[\![R]\!]_\nu$ termine.*

Preuve Le sens direct se montre par l'absurde : il suffit de considérer une chaîne infinie dans $[\![R]\!]_\nu$ et de la relever avec le lemme 28 page 93 en une chaîne infinie dans R. Notons que les relèvements doivent être choisis plus contraints que dans la preuve du lemme 28 page 93. Réciproquement, si R ne termine pas alors nous pouvons construire une chaîne infinie de mots paramétrés que nous pouvons interpréter en une chaîne infinie dans l'interprétation. $\diamond$

Pour montrer la terminaison nous pouvons tenter de construire un ordre bien fondé sur les mots paramétrés qui est stable par ν-interprétation. À cause des exposants, la longueur n'est plus un critère simple d'orientation des règles mais peut toujours être utilisée en comparant des expressions arithmétiques qui représentent la longueur. Des tels ordres devront être stables par interprétation. Par exemple la règle $a_i^i \to \epsilon | i > 0$ peut être orientée en interprétant a_i^i par l'expression arithmétique i et ϵ en 0. Sous la contrainte $i > 0$ nous avons bien ordonné la règle.

5.3.5 Formes normales

Formes normales standard

Définition 104 *Un mot est dit en forme normale pour un système R s'il n'est pas réductible par R.*

Un algorithme très simple de décision de la propriété "être en forme normale" consiste à tester la réductibilité. Clairement la notion de forme normale n'est pas stable par interprétation comme dans le cas des mots avec indices sans exposants.

Formes normales uniformes

Nous pouvons raffiner la notion de forme normale afin de pouvoir décrire plus finement les formes normales de l'interprétation.

Définition 105 *Un mot est dit en* forme normale uniforme *si et seulement si toutes ses ν-interprétations sont en forme normale.*

Proposition 23 *Tout mot en forme normale uniforme est en forme normale.*

Preuve Claire. ◇

La réciproque est fausse comme dans le cas des mots avec indices sans exposants.

Nous pouvons décider la propriété de forme normale uniforme à l'aide de l'algorithme suivant.

Algorithme 8 (Forme normale uniformes) **Entrée :** *$U = u|\psi$ et $R = l \rightarrow r|\xi$ sans variables communes.*

Sortie : *$\top$ si U est en forme normale uniforme et $\bot$ sinon.*

Algorithme :

- *Soit $D = Dec(u|\psi)$ l'ensemble des décompositions non nécessairement uniformes de $u|\psi$.*
- *S'il existe $(pre, s, suf, \zeta) \in D$ tel qu'une forme résolue $\aleph$ de $l \equiv s|\zeta \ \wedge \ \xi$ vérifie*

$$\psi \ \wedge \ \aleph$$

satisfiable alors renvoyer $\bot$ sinon renvoyer $\top$.

Proposition 24 *L'algorithme de test de forme normale uniforme est correct.*

Preuve En passant à l'interprétation c'est clair en utilisant la complétude de la décomposition établie dans le lemme 17. ◇

Intuitivement la restriction que l'on impose pour la réduction par rapport à la réduction uniforme est que le facteur que l'on réduit est toujours à la même position. Dans le cas des mots sans exposants, les facteurs existent indépendamment de la valeur des paramètres et la notion de position dans un mot n'est pas paramétrée. Avec des exposants, une position ne peut plus être un entier, mais cela doit être une expression arithmétique.

5.3.6 Algorithmes de complétion

Algorithme 9 (Complétion)

Entrée : *E ensemble d'équations paramétrées, $>$ un ordre sur les mots paramétrés*

Résultat : *R ensemble de règles paramétrées*

Algorithme : *On commence avec E et $R = \emptyset$*

1. *On met en forme normale pour R toutes les équations de E en éliminant les équation triviales.*
2. *On oriente toutes les équations de E en utilisant $>$ et on les met dans R. Si certaines ne sont pas orientables, on les laisse dans E.*
3. *On met en forme normale les membres droits des règles par R.*
4. *On calcule toutes les paires critiques paramétrées pas encore calculées et on les ajoute comme des équations dans l'ensemble E.*
5. *Si $E = \emptyset$ alors on renvoie R sinon on recommence en 1.*

Cet algorithme est très classique et s'adapte parfaitement au cas de la réécriture paramétrée avec indices et exposants. Nous pouvons l'optimiser au prix de l'automatisation en utilisant un oracle qui met en forme normale uniforme les mots au lieu de faire une mise en forme normale classique.

Chapitre 6

Réécriture paramétrée avec indices et produits finis

Dans ce chapitre nous donnons une extension de la réécriture qui permet d'exprimer des systèmes équationnels paramétrés pour lesquels il n'existe pas à notre connaissance de formalisme établi. Ces systèmes sont très courants en mathématique et apparaissent souvent lors des analyses de divergence des procédures de complétion classique [20]. Il s'agit de ce que les mathématiciens nomment couramment les *produits finis*. Par exemple, l'ensemble des règles de la forme :

$$a_i a_{i-1} \dots a_j a_i \to a_{i-1} a_i a_{i-1} \dots a_j$$

avec i et j des entiers entre 1 et $N-1$ et $j < i$ intervient naturellement dans la description d'un système équationel complet ayant pour modèle le groupe symétrique $\mathcal{S}_N$. Cet ensemble sera noté dans notre formalisme :

$$(\prod_{k=0}^{i-j} a_{i-k}) a_i \to a_{i-1} (\prod_{k=0}^{i-j} a_{i-k}) | 1 \leq j < i \leq N-1$$

Nous définissons les outils nécessaires pour manipuler formellement de tels systèmes de réécriture et montrons un lemme des paires critiques.

Dans ce chapitre, nous ne présenterons que les différences que les produits finis introduisent par rapport aux exposants en suivant le même plan que dans le chapitre précédent.

Nous fixons un ensemble infini dénombrable de variables I et une signature paramétrée réduite $\Sigma = \{a_i | \varphi\}_{\mathcal{N}}$ comme définie dans le chapitre 3 page 27, avec $i \in I$.

6.1 Mots paramétrés avec indices et produits finis

6.1.1 Définition et sémantique

Définition 106 *Un mot paramétré avec indices et produits finis W est défini par la grammaire suivante :*

$$
\begin{array}{lcll}
W & := & M|\psi & \text{où } \psi \text{ est une formule arithmétique} \\
M & := & \epsilon & \\
 & | & NM & \\
 & | & \left(\prod_{k=e_1}^{e_2} a_f N\right) M & \text{où } k \text{ est une variable et} \\
 & & & \text{où } e_1,\ e_2 \text{ et } f \text{ sont des expressions arithmétiques linéaires} \\
N & := & \epsilon & \\
 & | & a_f N & \text{où } f \text{ est une expression arithmétique linéaire}
\end{array}
$$

Les a_f sont les lettres du mot*.*

Nous dirons que $\left(\prod_{k=e_1}^{e_2} N\right)$ *est un* facteur avec produit fini*.*

Notons bien que la grammaire interdit l'imbrication des produits finis. Souvent les facteurs avec produits finis seront notés

$$\left(\prod_{k=e_1}^{e_2} a_{f_1(k)} \dots a_{f_n(k)}\right)$$

pour insister sur la liaison de la variable k par le produit. Cela n'implique bien sûr pas la présence systématique de k dans les expressions en indice.

La linéarité est fondamentale dans l'algorithme de résolution de la similitude locale, comme nous le verrons dans la section 6.1.4 page 107. En fait il suffit que pour toute lettre $a_{f(k)}$ d'un facteur avec produit fini indexé par k, la formule $f(k)$ soit linéaire en k.

Exemple 107 Nous pouvons écrire le mot

$$a_p \left(\prod_{k=1}^{p-1} a_k a_{2k}\right) a_j | p \geq 1 \ \wedge \ j \geq 0$$

dans notre syntaxe.

Remarquons tout de suite que nous présentons ici une extension de la réécriture avec exposants avec contraintes linéaires. En effet, dans un mot paramétré avec indices et exposants, tout facteur avec exposant peut s'écrire

$$N^e = \left(\prod_{k=1}^{e} N\right)$$

si k n'est pas une variable de N. Les algorithmes de cette section peuvent donc s'appliquer aux mots avec exposants, mais sont moins adaptés à ce cas. Il est possible de les adapter afin de retrouver des règles raisonnables pour les mots avec exposants. En particulier, nous verrons que la décision de la similitude locale exige en pratique de traiter de nombreux cas inutiles si les mots n'ont que des exposants. Forts de cette remarque, la plupart des notions ne seront pas redéfinies si elles coïncident avec les notions concernant les mots avec exposants. Par exemple nous utiliserons la même définition pour la réécriture.

Définissons la notion de mot justement contraint.

Définition 108 *Le mot $u|\psi$ sur la signature $\Sigma = \{a_i|\varphi\}_N$ est dit* justement contraint *si et seulement si*

- *En notant $a_{f_1}, \ldots, a_{f_k}$ les lettres de u qui n'apparaissent pas dans un facteur avec produit fini,*

$$\psi \Rightarrow \varphi[i \mapsto f_1] \ \wedge \ \ldots \ \wedge \ \varphi[i \mapsto f_k]$$

  est une formule valide et
- *Pour tout facteur de la forme $\left(\prod_{k=e_1}^{e_2} a_{f_1(k)} \ldots a_{f_l(k)}\right)$, la formule*

$$\forall k \ (e_1 \leq k \leq e_2 \ \wedge \ \psi \Rightarrow \varphi[i \mapsto f_1(k)] \ \wedge \ \ldots \ \wedge \ \varphi[i \mapsto f_l(k)])$$

  est valide.

Notons que cette notion coïncide avec celle concernant les mots avec exposants en utilisant la transformation des exposants en des produits finis. Le cas $e_2 = e_1$ correspond à l'exposant 1 et le cas $e_2 - e_1 = -1$ à l'exposant 0.

Définissons la notion d'interprétation.

Définition 109 *Soit $\nu : \mathcal{N} \to \mathbb{Z}$. La ν-interprétation du mot justement contraint $u|\psi$ avec l'interprétation $\mu : I \to \mathbb{Z}$ est définie si $\nu(\mu(\psi))$ est vraie par :*

$[\![a_i v]\!]_\nu(\mu) = a_{\nu(\mu(i))} [\![v]\!]_\nu(\mu)$

$[\![\epsilon]\!]_\nu(\mu) = \epsilon$

Si $\nu(\mu(e_2 - e_1)) \geq 0$ alors

$$[\![\left(\prod_{i=e_1}^{e_2} a_{f_1(i)} \ldots a_{f_l(i)}\right) v]\!]_\nu(\mu) = \underbrace{[\![a_{f_1(e_1)} \ldots a_{f_l(e_1)}]\!]_\nu(\mu) \cdots [\![a_{f_1(e_2)} \ldots a_{f_l(e_2)}]\!]_\nu(\mu)}_{\nu(\mu(e_2 - e_1))+1 \text{ facteurs}} [\![v]\!]_\nu(\mu)$$

Si $\nu(\mu(e_2 - e_1)) = -1$ alors

$$[\![\left(\prod_{i=e_1}^{e_2} a_{f_1(i)} \ldots a_{f_l(i)}\right) v]\!]_\nu(\mu) = [\![v]\!]_\nu(\mu)$$

Elle est indéfinie si $\nu(\mu(\psi))$ est fausse.

Exemple 110 Choisissons $\nu : N \mapsto 3$ et $\mu : j \mapsto 5$. Comme $\nu(\mu(N-1)) + 1 = 3 > -1$, nous avons

$$[\![(\prod_{i=1}^{N} a_i a_{i+j})]\!]_\nu(\mu) = [\![a_1 a_{1+j}]\!]_\nu(\mu)[\![a_2 a_{2+j}]\!]_\nu(\mu)[\![a_3 a_{3+j}]\!]_\nu(\mu) = a_1 a_6 a_2 a_7 a_3 a_8$$

6.1.2 Substitutions et renommages des mots paramétrés

Nous étendons la définition des substitutions sur les mots avec produits finis.

Définition 111 *Soit σ une substitution des variables et des paramètres de Σ. La substitution du mot paramétré $u|\psi$ par σ est un mot paramétré défini par $\sigma(u)|\sigma(\psi)$ avec*
- $\sigma(a_i v) = a_{\sigma(i)}\sigma(v)$
- $\sigma(\epsilon) = \epsilon$
- $\sigma((\prod_{i=e_1}^{e_2} v)w) = (\prod_{i=\sigma(e_1)}^{\sigma(e_2)} \sigma(v))\sigma(w)$ *si i n'est pas dans le support de σ.*

L'application de la substitution $\sigma : i \mapsto f$ à un mot ou à une expression x sera parfois notée $x_{[i \leftarrow f]}$.

La notion de renommage est la même que dans les chapitres précédents.

Remarquons qu'il s'agit bien d'une extension de la notion concernant les mots avec exposants.

Lemme 29 *Soit ν une interprétation des paramètres et $u|\phi$ un mot paramétré. L'ensemble $\{[\![u|\phi]\!]_\nu(\mu)\ /\ \mu : I \to \mathbb{Z}\}$ est invariant par renommage des variables de $u|\phi$.*

La preuve est la même que celle du lemme 4

Dans toute la suite nous nous autoriserons à renommer les variables liées par des produits en des nouvelles variables. Ce renommage ne change bien sûr pas ni la sémantique des mots ni leur classe le similitude locale.

6.1.3 Équivalence globale - Forme réduite

L'équivalence globale est toujours la même mais par contre la notion de forme réduite n'est plus pertinente.

Définition 112 *Deux mots paramétrés $u|\phi$ et $v|\psi$ sont dits globalement équivalents si et seulement si pour toute interprétation des paramètres ν*

$$\{[\![u|\phi]\!]_\nu(\mu)\ /\ \mu : I \to \mathbb{Z}\} = \{[\![v|\psi]\!]_\nu(\mu)\ /\ \mu : I \to \mathbb{Z}\}$$

Contrairement au cas des mots avec indices et exposants, il n'est pas possible de se ramener au cas où les indices des lettres d'un mot sont des variables. Cela poserait des problèmes de captures de variables très difficiles à gérer du fait de la liaison de la variable du produit au niveau du mot lui-même. Ainsi dans le mot

$$\left(\prod_{i=0}^{N} a_{i+j}a_{2i-j}\right)|j > 0$$

la variable i est liée par le produit et ne peut apparaître dans la contrainte.

6.1.4 Similitude locale

Définition et sémantique

Nous généralisons la notion de similitude locale.

Définition 113 *Deux mots paramétrés $u|\phi$ et $v|\psi$ sont dits* localement similaires *si et seulement si pour toutes interprétations $\nu : \mathcal{N} \to \mathbb{Z}$ et $\mu : I \to \mathbb{Z}$*
- *Soit $\nu(\mu(\phi))$ et $\nu(\mu(\psi))$ sont fausses*
- *Soit $\nu(\mu(\phi))$ et $\nu(\mu(\psi))$ sont vraies et*

$$[\![u|\phi]\!]_\nu(\mu) = [\![v|\psi]\!]_\nu(\mu)$$

Dans le paragraphe suivant nous donnons un algorithme de décision de la similitude locale de deux mots en suivant le même schéma que pour les mots avec exposants.

Définition 114 *Une formule de similitude entre les deux mots $u|\phi$ et $v|\psi$ notée*

$$u \equiv v|\phi \ \wedge \ \psi$$

est une formule telle que pour tout $\nu : \mathcal{N} \to \mathbb{Z}$ et tout $\mu : I \to \mathbb{Z}$,

$$\nu(\mu(u \equiv v|\phi \ \wedge \ \psi)) = \top \quad ssi \quad (\nu(\mu(u)) = \nu(\mu(v))) \ et \ \nu(\mu(\phi \ \wedge \ \psi))) = \top$$

Algorithme de résolution de la formule de similitude

Comme pour les mots avec exposants, nous donnons un algorithme qui, étant donnée une formule de similitude $u \equiv v|\phi$, calcule une formule *arithmétique* ψ qui lui est équivalente.

Définition 115 *Une formule de similitude est dite en forme résolue si c'est une formule de l'arithmétique.*

Modifions l'algorithme de résolution de la formule de similitude pour prendre en compte les produits finis.

Algorithme 10 (Résolution de la formule de similitude avec produits finis)
Entrée : *une formule de similitude* $u \equiv v|\psi$.
Résultat : *une formule de similitude résolue équivalente à l'entrée.*
Algorithme :

$$\frac{\epsilon \equiv \epsilon|\phi}{\phi}$$

$$\frac{\epsilon \equiv a_i u|\phi}{\bot}$$

$$\frac{\epsilon \equiv \left(\prod_{k=e_1}^{e_2} x\right) u|\phi}{\epsilon \equiv u|\phi \ \wedge \ e_1 - 1 = e_2}$$

$$\frac{a_i u \equiv a_j v|\phi}{u \equiv v|\phi \ \wedge \ i = j}$$

$$\frac{a_i u \equiv \left(\prod_{k=e_1}^{e_2} a_{f(k)} x\right) v|\phi}{\begin{array}{l}(a_i u \equiv v|\phi \ \wedge \ e_2 - e_1 = -1)\\ \bigvee(u \equiv \left(\prod_{k=e_1}^{e_2-1} x a_{f(k+1)}\right) x_{[k \leftarrow e_2]} v|\phi \ \wedge \ e_2 \geq e_1 \ \wedge \ i = f(e_1))\end{array}}$$

$$\frac{\left(\prod_{k=e_1}^{e_2} x\right) u \equiv \left(\prod_{l=f_1}^{f_2} y\right) v|\phi \qquad si \left\{\begin{array}{l} |x| = |y| \ et \\ \neg \mathit{IsPrefix}(x_{[k \leftarrow e_2+1]}, u, \phi) \\ \quad \wedge \ \neg \mathit{IsPrefix}(y_{[l \leftarrow f_2+1]}, v, \phi) \end{array}\right\} \ valide}{\begin{array}{l}\big(u \equiv v|\phi \ \wedge \ (e_2 - e_1 = f_2 - f_1) \ \wedge \ (e_2 - e_1 > 0) \ \wedge \ x_{[k \leftarrow e_1]} \equiv y_{[l \leftarrow f_1]} \ \wedge \ x_{[k \leftarrow e_2]} \equiv y_{[l \leftarrow f_2]}\big)\\ \bigvee \left(\left(\prod_{k=e_1}^{e_2} x\right) u \equiv y_{[l \leftarrow f_1]} v|\phi \ \wedge \ f_2 - f_1 = 0\right)\\ \bigvee \left(x_{[k \leftarrow e_2]} u \equiv \left(\prod_{l=f_1}^{f_2} y\right) v|\phi \ \wedge \ e_2 - e_1 = 0\right)\\ \bigvee \left(\left(\prod_{k=e_1}^{e_2} x\right) u \equiv v|\phi \ \wedge \ f_2 - f_1 = -1\right)\\ \bigvee \left(u \equiv \left(\prod_{l=f_1}^{f_2} y\right) v|\phi \ \wedge \ e_2 - e_1 = -1\right)\end{array}}$$

$$\dfrac{\left(\prod_{k=e_1}^{e_2} x\right) u \equiv \left(\prod_{l=f_1}^{f_2} y\right) v|\phi \quad si \begin{cases} |x| < |y| \ ou \\ \begin{cases} |x| = |y| \ et \\ (\neg IsPrefix(x_{[k\leftarrow e_2+1]}, u, \phi) \wedge \\ \neg IsPrefix(y_{[l\leftarrow f_2+1]}, v, \phi)) \\ \ non\ valide \end{cases} \end{cases}}{\bigvee\limits_{\substack{(e_2', u', \phi_u') \in LFact(x,k,e_2,u,\phi) \\ (f_2', v', \phi_v') \in LFact\left(x,k,e_1-1,\left(\prod_{l=f_1}^{f_2} y\right) v, \phi\right)}} \left(\left(\prod_{k=e_1}^{e_2'} x\right) u' \equiv \left(\prod_{k=e_1}^{f_2'} x\right) v'|\phi \ \wedge\ \phi_u' \ \wedge\ \phi_v'\right)}$$

La correction de la sixième règle repose fondamentalement sur l'hypothèse de linéarité des indices : elle permet d'affirmer qu'il suffit que deux expressions coïncident en deux points distincts pour qu'elles soient égales partout. Pour les cas où il n'y a pas au moins deux points distincts dans les produits, nous expansons le produit explicitement. Cela permet d'appliquer l'avant-dernière règle sur la formule de similitude suivante :

$$\left(\prod_{k=2}^{2} a_k\right) a_1 a_0 \equiv \left(\prod_{l=1}^{3} a_{3-l}\right)$$

et d'obtenir la formule

$$a_2 a_1 a_0 \equiv \left(\prod_{l=1}^{3} a_{3-l}\right)$$

Notons que nous pourrions relâcher la contrainte de linéarité en demandant seulement la linéarité des indices de chaque facteur avec produit fini en la variable liée du ce produit produit.

Cet algorithme dépend de deux notions, similaires à celles développées dans le cas des exposants, pour lesquelles nous allons donner des définitions formelles et des algorithmes de construction :

- $IsPrefix(x, u, \phi)$ (où x est un mot sans produits finis et $u|\phi$ un mot contraint avec produits finis) est une formule telle que si $(x_0, u_0) \in [\![x, u|\phi]\!]_\nu$ alors x_0 est un préfixe de u_0. L'algorithme 11 page suivante permet de calculer cette formule. Nous avons donc exactement la même sémantique que dans le cas des mots avec exposants.
- $LFact(x, k, e, u, \phi)$ (acronyme désignant *Left Factorizations* c'est-à-dire factorisations à gauche) est l'ensemble des triplets (e', u', ϕ') où e' est une expression arithmétique et $u'|\phi'$ un mot contraint avec produits finis tels que l'équation paramétrée

$$u \equiv \left(\prod_{k=e}^{e'} x\right) u'|\phi'$$

est valide et $\mathit{IsPrefix}(x_{[k\leftarrow e'+1]}, u', \phi')$ n'est pas satisfiable. Il s'agit informellement de l'ensemble des écritures possibles distinctes de $u|\phi$ comme un produit fini de x concaténé à un mot paramétré dont aucune instance ne permet de prolonger le produit fini. Cet ensemble est calculé par l'algorithme 13 page 113.

Donnons maintenant un algorithme de calcul de *IsPrefix* :

Algorithme 11 (*IsPrefix*) Entrée : *x un mot sans produits finis et $u|\psi$ un mot paramétré avec produits finis. L'algorithme débute sur le triplet (x, u, ψ).*

Résultat : *une formule ϕ vérifiant la proposition 25.*

Algorithme :

$$\frac{\epsilon, w, \psi}{\psi}$$

$$\frac{a_i x, \epsilon, \psi}{\perp}$$

$$\frac{a_i x, a_j u, \psi}{x, u, i = j \ \wedge \ \psi}$$

$$\frac{a_i x, \left(\prod_{k=e_1}^{e_2} a_{f(k)} y\right) u, \psi}{(a_i x, u, \psi \ \wedge \ e_2 - e_1 = -1) \vee (x, \left(\prod_{k=e_1}^{e_2-1} y a_{f(k+1)}\right) y_{[k\leftarrow e_2]} u, \psi \ \wedge \ e_1 \leq e_2 \ \wedge \ i = f(e_1))}$$

Proposition 25 *Soit x un mot sans produits finis et $u|\psi$ un mot paramétré avec indices et produits finis. Le calcul de IsPrefix(x, u, ψ) termine et donne une formule logique telle que pour tout $\nu : \mathcal{N} \to \mathbb{Z}$ et tout $\mu : I \to N$,*

$$\nu(\mu(\mathit{IsPrefix}(x, u, \psi))) = \top$$

si et seulement si

- $\nu(\mu(\psi)) = \top$ *et*
- $\nu(\mu(x))$ *est un préfixe de* $\nu(\mu(u))$.

Preuve La preuve est semblable à celle de la proposition 13 page 73. Nous précisons seulement la correction de la dernière règle qui est différente. Nous fixons $\nu : \mathcal{N} \to \mathbb{Z}$ et $\mu : I \to \mathbb{Z}$.

- Si
$$\nu(\mu(\mathit{IsPrefix}(a_i x, \left(\prod_{k=e_1}^{e_2} a_{f(k)} y\right) u, \psi)))$$
est vraie alors $\nu(\mu(\psi))$ est vraie et, puisque le mot est justement contraint, nous avons $\nu(\mu(e_2 - e_1)) \geq -1$.

  - Si $\nu(\mu(e_2 - e_1)) = -1$ alors $\nu(\mu(a_i x))$ est un préfixe de $\nu(\mu(u))$ et on a bien $\nu(\mu(\mathit{IsPrefix}(a_i x, u, \psi \ \wedge \ e_2 - e_1 = -1)))$.
  - Si $\nu(\mu(e_2 - e_1)) \geq 0$ alors $\nu(\mu(a_i x))$ est un préfixe de
  $$\nu(\mu(a_{f(e_1)} y_{[k \leftarrow e_1]} \left(\prod_{k=e_1+1}^{e_2} a_{f(k)} y \right) u))$$
  On en déduit que $\nu(\mu(i)) = \nu(\mu(f(e_1)))$ et $\nu(\mu(x))$ est un préfixe du mot
  $$\nu(\mu(y_{[k \leftarrow e_1]} \left(\prod_{k=e_1+1}^{e_2} a_{f(k)} y \right) u))$$
  qui est égal à
  $$\nu(\mu(\left(\prod_{k=e_1}^{e_2-1} y a_{f(k+1)} \right) y_{[k \leftarrow e_2]} u))$$
  Nous avons bien montré que
  $$\nu(\mu(\mathit{IsPrefix}(x, \left(\prod_{k=e_1}^{e_2-1} y a_{f(k+1)} \right) y_{[k \leftarrow e_2]} u, e_2 - e_1 \geq 0 \ \wedge \ i = f(e_1) \ \wedge \ \psi)))$$
  est vraie.
- Réciproquement, montrons que chaque élément de la disjonction implique que
$$\nu(\mu(\mathit{IsPrefix}(a_i x, \left(\prod_{k=e_1}^{e_2} a_{f(k)y} \right) u, \psi)))$$
est vraie.
  - Si $\nu(\mu(\mathit{IsPrefix}(a_i x, u, \psi \ \wedge \ e_2 - e_1 = -1)))$ est vraie alors $\nu(\mu(\psi))$ est vraie et $\nu(\mu(e_2 - e_1)) = -1$. Comme $\nu(\mu(a_i x))$ est un préfixe de $\nu(\mu(u))$, on a bien
  $$\nu(\mu(\mathit{IsPrefix}(a_i x, \left(\prod_{k=e_1}^{e_2} a_{f(k)y} \right) u, \psi)))$$
  vraie.
  - Si
  $$\mathit{IsPrefix}(x, \left(\prod_{k=e_1}^{e_2-1} y a_{f(k+1)} \right) y_{[k \leftarrow e_2]} u, \psi \ \wedge \ e_1 \leq e_2 \ \wedge \ i = f(e_1))$$
  est vraie alors $\nu(\mu(\psi))$ est vraie, $\nu(\mu(e_2 - e_1)) \geq 0$ et $\nu(\mu(i)) = \nu(\mu(f(e_1)))$. Comme $\nu(\mu(x))$ est un préfixe de
  $$\nu(\mu(\left(\prod_{k=e_1}^{e_2-1} y a_{f(k+1)} \right) y_{[k \leftarrow e_2]} u))$$

nous avons $\nu(\mu(a_i x))$ préfixe de

$$\nu(\mu(a_i \left(\prod_{k=e_1}^{e_2-1} y a_{f(k+1)}\right) y_{[k\leftarrow e_2]} u))$$

Ce dernier est égal à

$$\nu(\mu(a_{f(e_1)} \left(\prod_{k=e_1}^{e_2-1} y a_{f(k+1)}\right) y_{[k\leftarrow e_2]} u))$$

qui peut s'écrire après décalage :

$$\nu(\mu(\left(\prod_{k=e_1}^{e_2} a_{f(k)} y\right) u))$$

Ce qui permet de conclure que

$$\nu(\mu(\mathit{IsPrefix}(a_i x, \left(\prod_{k=e_1}^{e_2} a_{f(k)y}\right) u, \psi)))$$

est vraie.

$\diamond$

Comme pour dans le cas des mots avec exposants, nous modifions cet algorithme afin de pouvoir décrire de façon finie toutes les instances de $u|\psi$ de la forme $xu'|\psi'$. Pour cela il suffit de changer la première règle en

$$\frac{\epsilon, w, \psi}{w|\psi}$$

et de remplacer les $\vee$ par des unions ensemblistes. On obtient l'algorithme suivant :

Algorithme 12 (*RemovePrefix***) Entrée :** *x un mot sans produits finis et $u|\psi$ un mot paramétré avec produits finis. L'algorithme commence avec le triplet (x, u, ψ).*

Résultat : *un ensemble de mots $u'|\phi'$ vérifiant la proposition 26 page suivante.*

Algorithme :

$$\frac{\epsilon, w, \psi}{w|\psi}$$

$$\frac{a_i x, \epsilon, \psi}{\emptyset}$$

$$\frac{a_i x, a_j u, \psi}{x, u, i = j \ \wedge \ \psi}$$

$$\frac{a_i x, \left(\prod_{k=e_1}^{e_2} a_{f(k)} y\right) u, \psi}{\{(a_i x, u, \psi \ \wedge \ e_2 - e_1 = -1)\} \cup \{(x, \left(\prod_{k=e_1}^{e_2-1} y a_{f(k+1)}\right) y_{[k \leftarrow e_2]} u, \psi \ \wedge \ e_1 \leq e_2 \ \wedge \ i = f(e_1))\}}$$

Proposition 26 *RemovePrefix*$(x, u|\phi)$ *est un ensemble dont tout élément* $u'|\phi'$ *est tel que pour tout* $\nu : \mathcal{N} \to \mathbb{Z}$ *et tout* $\mu : I \to \mathbb{Z}$

- *Si* $\nu(\mu(\phi')) = \top$ *alors*
$$[\![xu'|\phi']\!]_\nu(\mu) = [\![u|\phi]\!]_\nu(\mu)$$
- *Si* $[\![x, u|\phi]\!]_\nu(\mu) = (x_0, u_0)$ *et s'il existe* u'_0 *tel que* $u_0 = x_0 u'_0$ *alors il existe un élément* $u'|\phi' \in$ *RemovePrefix*$(x, u|\phi)$ *tel que* $u'_0 = [\![u'|\phi']\!]_\nu(\mu)$
- *Si* $x \neq \epsilon$, *alors*
  - *Soit* u' *a strictement moins de facteurs avec produits finis que* u
  - *Soit la longueur du plus long préfixe sans produits finis de* u' *est strictement plus petite que la longueur du plus long préfixe sans produit finis de* u.

Preuve La preuve est similaire à celle de la proposition 25 page 110. $\diamond$

Nous donnons maintenant les règles de calcul de *LFact*. Cet algorithme énumère toutes les façons d'écrire un mot comme le produit fini d'un mot donné concaténé avec un mot dont aucune instance ne permet de prolonger le produit fini.

Algorithme 13 (*LFact*) **Entrée :** *x un mot sans produits finis, k une variable, une expression de départ* e_0 *et* $u_0|\phi_0$ *un mot paramétré avec produits finis. L'algorithme commence avec le triplet* (e_0, u_0, ϕ_0).

Résultat : *un ensemble de triplets* (e', u', ϕ') *vérifiant la proposition 27 page 118*

Algorithme :

$$\frac{e, yu, \phi \qquad\qquad si\ |x| = |y|}{(e+1, u, \phi \ \wedge \ x_{[k \leftarrow e+1]} \equiv y); Result(e, yu, \phi \ \wedge \ \neg x_{[k \leftarrow e+1]} \equiv y)}$$

$$\frac{e, \left(\prod_{l=e_1}^{e_2} y\right) u, \phi \qquad\qquad si\ |x| = |y|}{\begin{array}{l}(e+1+e_2-e_1, u, \phi \ \wedge \ e_1 \leq e_2 \ \wedge \ x_{[k \leftarrow e+1]} \equiv y_{[l \leftarrow e_1]} \ \wedge \ x_{[k \leftarrow e+e_2-e_1]} \equiv y_{[l \leftarrow e_2]}); \\ (e, u, \phi \ \wedge \ e_2 - e_1 = -1); \\ Result(e+1, \left(\prod_{l=e_1+1}^{e_2} y\right) u, \phi \ \wedge \ e_1 < e_2 \ \wedge \ x_{[k \leftarrow e+1]} \equiv y_{[l \leftarrow e_1]} \ \wedge \ \neg(x_{[k \leftarrow e+e_2-e_1]} \equiv y_{[l \leftarrow e_2]})); \\ Result(e, \left(\prod_{l=e_1}^{e_2} y\right) u, \phi \ \wedge \ e_1 \leq e_2 \ \wedge \ \neg(x_{[k \leftarrow e+1]} \equiv y_{[l \leftarrow e_1]}));\end{array}}$$

$$\frac{e, y, \phi \qquad\qquad si\ |x| > |y|}{Result(e, y, \phi)}$$

$$\frac{e, y\left(\prod_{l=e_1}^{e_2} z\right) v, \phi \qquad\qquad si \begin{cases} |x| > |y| \\ R = RemovePrefix(x_{[k\leftarrow e+1]}, y\left(\prod_{l=e_1}^{e_2} z\right) v, \phi) \end{cases}}{\bigcup_{v'|\psi'\in R}\{(e+1, v', \phi\ \wedge\ \psi')\}; Result(e, y\left(\prod_{l=e_1}^{e_2} z\right) v, \phi\ \wedge\ (\ \wedge_{\ u'|\psi'\in R} \neg\psi'))\}}$$

$$\frac{e, \left(\prod_{l=e_1}^{e_2} y\right) v, \phi \qquad\qquad si \begin{cases} |x| \neq |y| \\ d_x = ppcm(|x|, |y|)/|x| \\ d_y = ppcm(|x|, |y|)/|y| \end{cases}}{\begin{array}{l} \bigcup_{r=0}^{2d_y-1}\{e, \left(\prod_{l=1}^{r} y_{[l\leftarrow e_1-1+l]}\right) u, \phi\ \wedge\ e_2 - e_1 = r - 1\} \cup \\ \bigcup_{r=0}^{2d_x-1}\left\{ Result\left(\begin{array}{l} e+r; \\ \tilde{u}; \\ \ \phi\ \wedge\ (e_2 - e_1 \geq 2d_y - 1) \\ \quad \wedge\ \left(\prod_{k=e+1}^{e+r} x\right) \equiv Pref_{|x|.r}\left(\prod_{l=e_1}^{e_1+2d_y-1} y\right) \\ \quad \wedge \neg\left(\left(\prod_{k=e+1}^{e+r+1} x\right) \equiv Pref_{|x|.(r+1)}\left(\prod_{l=e_1}^{e_1+2d_y-1} y\right)\right) \end{array} \right)\right\} \cup \\ \bigcup_{r=0}^{d_y-1}\left\{ \begin{array}{l} e + \frac{d_x(e_2-e_1-r+1)}{d_y}; \\ \left(\prod_{l=e_2-r+1}^{e_2} y\right) u; \\ \phi\ \wedge\ \left(\prod_{k=e+1}^{e+2d_x} x\right) \equiv \left(\prod_{l=e_1}^{e_1+2d_y-1} y\right)\ \wedge\ e_2 - e_1 \geq 2d_y - 1\ \wedge\ (e_2 - e_1 + 1 = r \mod d_y) \end{array} \right\} \end{array}}$$

Dans la dernière règle, $\tilde{u}$ désigne une écriture du mot

$$\left(Suff_{|x|.r}\left(\prod_{l=e_1}^{e_1+d_y-1} y\right)\right)\left(\prod_{l=e_1+d_y}^{e_2} y\right)$$

qui commence par un facteur avec un produit fini. Nous pouvons construire $\tilde{u}$ en développant explicitement $Suff_{|x|.r}\left(\prod_{l=e_1}^{e_1+d_y-1} y\right)$ et en introduisant ses lettres dans le produit $\left(\prod_{l=e_1+d_y}^{e_2} y\right)$. C'est une manipulation symbolique analogue à celle pratiquée pour les exposants.

Rappelons que

$$Pref_j(a_{i_1} \dots a_{i_j} \dots a_{i_k}) = a_{i_1} \dots a_{i_j}$$

avec j *un entier naturel et*

$$Suff_j(a_{i_1} \dots a_{i_j} \dots a_{i_k}) = a_{i_j} \dots a_{i_k}$$

Notons de plus que la dernière règle introduit des facteurs avec des produits finis portant sur un intervalle de taille entière donnée. Ces derniers doivent être développés avant de continuer l'algorithme.

Par exemple

$$\left(\prod_{l=1}^{r} y\right)$$

où r *est un entier doit être développé en*

$$y_{[l \leftarrow 1]} \cdots y_{[l \leftarrow r]}$$

Les facteurs 2 *dans cette dernière règle permettent de vérifier que expressions en indice sont bien égales en deux points distincts.*

Exemple 116 Étudions une partie du calcul de

$$LFact(a_{k+1}a_{k+n}, k, 0, \left(\prod_{l=2}^{p} a_l a_{l+1} a_{l+2}\right), \top)$$

La dernière règle s'applique avec $x = a_{k+1}a_{k+n}$ et $y = a_l a_{l+1} a_{l+2}$. Nous avons $d_x = 3$ et $d_y = 2$.

1. Premier ensemble de solutions : il s'agit de développer le produit pour les petites valeurs de ses variables et paramètres.
   - Pour $r = 0$ nous avons :

     $$(0, \epsilon, p - 2 = -1)$$

     C'est le cas où $p = 1$, et donc le produit des y dégénère en ϵ. En appliquant la troisième règle nous obtenons :

     $$Result(0, \epsilon, p = 1)$$

   - Pour $r = 1$ nous avons :

     $$(0, a_2a_3a_4, p - 2 = 1 - 1)$$

     Cette fois le produit des y dégénère en un seul élément. La règle 1 permet de continuer en

     $$(1, a_4, p = 2 \ \wedge \ (a_2a_{1+n} \equiv a_2a_3)); Result(0, a_2a_3a_4, p = 2 \ \wedge \ \neg(a_2a_{n+1} \equiv a_2a_3))$$

     Ce qui se simplifie en

     $$Result(1, a_4, p = 2 \ \wedge \ n = 2); Result(0, a_2a_3a_4, p = 2 \ \wedge \ n \neq 2)$$

– Pour $r = 2$ nous avons :

$$(0, a_2a_3a_4a_3a_4a_5, p-2 = 2-1)$$

– Pour $r = 3$ nous avons :

$$(0, a_2a_3a_4a_3a_4a_5a_4a_5a_6, p-2 = 3-1)$$

2. Deuxième ensemble de solutions : les cas d'arrêt de l'algorithme. Dans cet ensemble nous trouvons les cas où les valeurs des bornes sont assez grandes mais les indices ne coïncident pas tous en plus d'un point.
   – Pour $r = 0$ avant décalage en $\tilde{u}$:

$$\mathit{Result}\left(\begin{array}{l} 0+0; \\ \mathrm{Suff}_0\left(\prod_{l=2}^{2+2-1} a_l a_{l+1} a_{l+2}\right)\left(\prod_{l=2+2}^{p} a_l a_{l+1} a_{l+2}\right); \\ \quad (p-2 \geq 4-1) \\ \quad \wedge \left(\prod_{k=0+1}^{0+0} a_{k+1}a_{k+n}\right) \equiv \mathrm{Pref}_0\left(\prod_{l=2}^{2+2-1} a_l a_{l+1} a_{l+2}\right) \\ \quad \wedge \neg\left(\left(\prod_{k=0+1}^{0+0+1} a_{k+1}a_{k+n}\right) \equiv \mathrm{Pref}_{2\times(0+1)}\left(\prod_{l=2}^{2+2-1} a_l a_{l+1} a_{l+2}\right)\right) \end{array}\right)$$

   se simplifie en

$$\mathit{Result}\left(\begin{array}{l} 0; \\ \left(\prod_{l=2}^{3} a_l a_{l+1} a_{l+2}\right)\left(\prod_{l=4}^{p} a_l a_{l+1} a_{l+2}\right); \\ (p \geq 5) \\ \ \wedge\ \epsilon \equiv \epsilon \\ \ \wedge\ \neg\,(a_2a_{1+n} \equiv a_2a_3) \end{array}\right)$$

   En appliquant le décalage défini pour $\tilde{u}$ et en simplifiant encore nous obtenons :

$$\mathit{Result}(0; \left(\prod_{l=2}^{p-2} a_l a_{l+1} a_{l+2}\right) a_{p-1}a_pa_{p+1}a_pa_{p+1}a_{p+2}; p \geq 5 \ \wedge\ n \neq 2$$

   – Pour $r = 1$ avant décalage en $\tilde{u}$:

$$\mathit{Result}\left(\begin{array}{l} 0+1; \\ \mathrm{Suff}_2\left(\prod_{l=2}^{3} a_l a_{l+1} a_{l+2}\right)\left(\prod_{l=4}^{p} a_l a_{l+1} a_{l+2}\right); \\ \quad p \geq 5 \\ \quad \wedge \left(\prod_{k=1}^{0+1} a_{k+1}a_{k+n}\right) \equiv \mathrm{Pref}_2\left(\prod_{l=2}^{3} a_l a_{l+1} a_{l+2}\right) \\ \quad \wedge \neg\left(\left(\prod_{k=1}^{0+1+1} a_{k+1}a_{k+n}\right) \equiv \mathrm{Pref}_{2\times(1+1)}\left(\prod_{l=2}^{2+2-1} a_l a_{l+1} a_{l+2}\right)\right) \end{array}\right)$$

Nous simplifions en :

$$\mathit{Result}\left(\begin{array}{l} 1; \\ a_4a_3a_4a_5\left(\prod_{l=4}^{p} a_la_{l+1}a_{l+2}\right); \\ \quad p \geq 5 \\ \quad\ \wedge\ a_2a_{1+n} \equiv a_2a_3 \\ \quad\ \wedge\ \neg\left(a_2a_{1+n}a_3a_{2+n} \equiv a_2a_3a_4a_3\right) \end{array}\right)$$

Nous avons donc après décalage et simplifications :

$$\mathit{Result}(1; \left(\prod_{l=2}^{p-2} a_{l+2}a_{l+1}a_{l+2}\right) a_{p+1}a_pa_{p+1}a_{p+2}; p \geq 5\ \wedge\ n = 2)$$

Nous pouvons interpréter ce résultat comme le fait que si $p \geq 5$ et $n = 2$ alors $x_{[k\leftarrow 1]}$ est un préfixe de $\left(\prod_{l=2}^{p} a_la_{l+1}a_{l+2}\right)$, mais $x_{[k\leftarrow 1]}x_{[k\leftarrow 2]}$ n'en est pas un.

– Les cas $r = 2, 3, 4, 5$ sont similaires au précédent.

3. Troisième ensemble de solutions : il s'agit des cas pour lesquels le produit paramétré en y peut prolonger le produit des x totalement sauf peut-être pour les dernières valeurs.

– $r = 0$:

$$\left(\begin{array}{l} 3(p-2-0+1)/2; \\ \epsilon; \\ a_2a_{1+n}.a_3a_{2+n}.a_4a_{3+n}.a_5a_{4+n}.a_6a_{5+n}a_7a_{6+n} \equiv a_2a_3a_4.a_3a_4a_4.a_4a_5a_6.a_5a_6a_7 \\ \ \wedge\ p-2 \geq 3\ \wedge\ (p-2+1 = 0 \mod 2) \end{array}\right)$$

À cause des indices, ce triplet se simplifie en

$$\left(\begin{array}{l} 3(p-1)/2; \\ \epsilon; \\ \perp \end{array}\right)$$

– $r = 1$:

$$\left(\begin{array}{l} 3(p-2-1+1)/2; \\ a_pa_{p+1}a_{p+2}; \\ a_2a_{1+n}.a_3a_{2+n}.a_4a_{3+n}.a_5a_{4+n}.a_6a_{5+n}a_7a_{6+n} \equiv a_2a_3a_4.a_3a_4a_4.a_4a_5a_6.a_5a_6a_7 \\ \ \wedge\ p-2 \geq 3\ \wedge\ (p-2+1 = 1 \mod 2) \end{array}\right)$$

À cause des indices, ce triplet se simplifie en

$$\left(\begin{array}{l} 3(p-2)/2; \\ a_pa_{p+1}a_{p+2}; \\ \perp \end{array}\right)$$

Lemme 30 (Décroissance de *LFact***)** *Pour tout* $(e', u', \psi') \in LFact(x, k, e, u, \psi)$,

- *Soit le nombre de facteurs avec produits finis dans* u' *est strictement plus petit que celui dans* u *;*
- *Soit* u *et* u' *ont le même nombre de facteurs avec produits finis et la taille du plus long préfixe sans produits finis de* u' *est strictement plus petite que celle du plus long préfixe sans produits finis de* u*.*

Preuve En remarquant que les produits finis introduits sont tous de la forme

$$\left(\prod_{k=e}^{e+j} z\right)$$

où j est un entier fixé, il est clair qu'il est possible de les développer explicitement pour les remplacer par des facteurs sans produits finis. Chaque règle respecte bien l'ordre donné, y compris la dernière grâce à l'utilisation de $\tilde{y}$. $\diamond$

Proposition 27 (Sémantique de *LFact***)** *Soit* $L = LFact(x, k, e, u, \psi)$.

1. *Pour toute interprétation* $\nu : \mathcal{N} \to \mathbb{Z}$ *et* $\mu : I \to \mathbb{Z}$
   - *Pour tout* $(e', u', \psi') \in L$, *si* $\nu(\mu(\psi'))$ *est vraie alors* $\nu(\mu(\psi))$ *est vraie et*

$$[\![\left(\prod_{k=e+1}^{e'} x\right) u'|\phi]\!]_\nu(\mu) = [\![u|\psi]\!]_\nu(\mu)$$

   - *Si* $\nu(\mu(\psi))$ *est vraie alors il existe* $(e', u', \psi') \in L$ *tel que* $\nu(\mu(\psi'))$ *est vraie et*

$$[\![u|\psi]\!]_\nu(\mu) = [\![\left(\prod_{k=e+1}^{e'} x\right) u'|\psi']\!]_\nu(\mu)$$

2. *Pour tout* $(e', u', \psi') \in L$, *la formule* $\neg IsPrefix(x_{[k \leftarrow e'+1]}, u', \psi')$ *est valide*

Preuve Intuitivement, en oubliant les contraintes, chaque règle de la forme

$$\frac{e, v}{e', v'}$$

vérifie que

$$u_0 = \left(\prod_{k=e_0+1}^{e} x\right) v$$

et

$$u_0 = \left(\prod_{k=e_0+1}^{e'} x\right) v'$$

À l'aide de cette remarque, nous pouvons facilement établir la proposition en passant à l'interprétation et en vérifiant l'invariant sur chaque règle. $\diamond$

Nous pouvons maintenant prouver la correction de l'algorithme de résolution d'une formule de similitude afin d'obtenir l'analogue du lemme 21 page 77.

Lemme 31 *Soit $u \equiv v|\phi$ une formule de similitude. L'algorithme de résolution termine sur $u \equiv v|\phi$ et donne une formule χ telle que pour toute interprétation des paramètres ν et des variables μ,*

- *Si $\nu(\mu(\chi))$ est vraie alors*
$$\nu(\mu(u)) = \nu(\mu(v))$$
est vraie si et seulement si
$$\nu(\mu(\phi)) = \top$$
- *Si $\nu(\mu(\chi))$ est fausse alors*
$$\nu(\mu(\phi)) = \bot$$
ou bien
$$\nu(\mu(u)) \neq \nu(\mu(v))$$

Preuve La preuve est très similaire à celle pour les mots avec exposants.

- Terminaison : en omettant la dernière, chaque règle diminue strictement soit le nombre de facteurs avec produits finis dans l'équation, soit la somme des tailles des plus grands préfixes sans produits finis des membres de la formule de similitude. Cela nous donne une combinaison lexicographique d'ordres qui assure la terminaison de l'algorithme sans la dernière règle. La dernière règle fait décroître cette mesure mais seulement au sens large d'après le lemme 30 page précédente. La proposition 27 page ci-contre montre que cette règle ne peut pas s'appliquer deux fois de suite. La terminaison de l'algorithme est ainsi établie.
- Correction : montrons que chaque règle transforme le problème en un problème équivalent. Nous énumérons toutes les règles qui sont changées par rapport au cas des exposants. On se fixe ν (resp. μ) une interprétation des paramètres (resp. des variables).
Pour chaque règle de la forme $\frac{u \equiv v|\psi}{u' \equiv v'|\psi'}$, il suffit de montrer que
$$\nu(\mu(u)) = \nu(\mu(v)) \ \wedge \ \nu(\mu(\psi)) \Leftrightarrow \nu(\mu(u')) = \nu(\mu(v')) \ \wedge \ \nu(\mu(\psi'))$$
  - Règle $\epsilon \equiv \left(\prod_{k=e_1}^{e_2} x\right) u|\phi$: la formule
$$\nu(\mu(\epsilon)) = \nu(\mu(\left(\prod_{k=e_1}^{e_2} x\right) u))$$
est équivalente à
$$\nu(\mu(\epsilon)) = \nu(\mu(u)) \ \wedge \ e_2 - e_1 = -1$$
pour des raisons de longueur.

– Règle $a_i u \equiv \left(\prod_{k=e_1}^{e_2} a_{f(k)}x\right) v|\phi$: la formule

$$\nu(\mu(a_i u)) = \nu(\mu(\left(\prod_{k=e_1}^{e_2} a_{f(k)}x\right) v)) \ \wedge \ \nu(\mu(\phi))$$

est équivalente à la disjonction

$$\begin{array}{l}((\nu(\mu(a_i u)) = \nu(\mu(v)) \ \wedge \ e_2 - e_1 = -1 \ \wedge \ \nu(\mu(\phi)))\vee \\ (\nu(\mu(a_i u)) = \nu(\mu(a_{f(e_1)}x_{[k\leftarrow e_1]}\left(\prod_{k=e_1+1}^{e_2} a_{f(k)}x\right) v)) \ \wedge \ e_2 - e_1 \geq 0) \ \wedge \ \nu(\mu(\phi)))\end{array}$$

En effet comme l'équation est justement contrainte, $\phi \Rightarrow e_2 - e_1 \geq -1$ est valide. La seconde partie de la disjonction est équivalente à

$$\nu(\mu(i)) = \nu(\mu(f(e_1))) \wedge (\nu(\mu(u)) = \nu(\mu(x_{[k\leftarrow e_1]}\left(\prod_{k=e_1+1}^{e_2} a_{f(k)}x\right) v)) \wedge e_2 - e_1 \geq 0)$$

Puis par récurrence sur $e_2 - e_1$, on peut montrer que

$$\nu(\mu(x_{[k\leftarrow e_1]}\left(\prod_{k=e_1+1}^{e_2} a_{f(k)}x\right) v)) = \nu(\mu(\left(\prod_{k=e_1}^{e_2-1} xa_{f(k+1)}\right) x_{[k\leftarrow e_2]}))$$

pour tout valeur de positive ou nulle de $e_2 - e_1$. Nous obtenons bien l'équivalence cherchée.

– Règle $\left(\prod_{k=e_1}^{e_2} x\right) u \equiv \left(\prod_{l=f_1}^{f_2} y\right) v|\phi$: supposons que les conditions d'application de la règle sont vérifiées i.e.

$$|x| = |y|$$

et

$$\neg \mathit{IsPrefix}(x_{[k\leftarrow e_2+1]}, u, \phi) \ \wedge \ \neg \mathit{IsPrefix}(y_{[l\leftarrow f_2+1]}, v, \phi)$$

est valide. Notons

$$E_1 = (\nu(\mu(\left(\prod_{k=e_1}^{e_2} x\right) u)) = \nu(\mu(\left(\prod_{l=f_1}^{f_2} y\right) v)) \ \wedge \ \nu(\mu(\phi)))$$

$$E_2^1 = (\nu(\mu\left(\begin{array}{l} u = v \ \wedge \ \phi \wedge \\ (e_2 - e_1 = f_2 - f_1) \ \wedge \ (e_2 - e_1 > 0) \wedge \\ x_{[k\leftarrow e_1]} = y_{[l\leftarrow f_1]} \wedge \\ x_{[k\leftarrow e_2]} = y_{[l\leftarrow f_2]} \end{array}\right)))$$

$$E_2^2 = \left(\nu(\mu(\left(\prod_{k=e_1}^{e_2} x\right) u = y_{[l\leftarrow f_1]}v \ \wedge\ \phi \ \wedge\ (f_2 - f_1) = 0))\right)$$

$$E_2^3 = \left(\nu(\mu((x_{[k\leftarrow e_2]}u = \left(\prod_{l=f_1}^{f_2} y\right) v) \ \wedge\ \phi \ \wedge\ (e_2 - e_1) = 0))\right)$$

$$E_2^4 = \left(\nu(\mu(\left(\prod_{k=e_1}^{e_2} x\right) u = v \ \wedge\ \phi \ \wedge\ (f_2 - f_1) = -1))\right)$$

$$E_2^5 = \left(\nu(\mu(u = \left(\prod_{l=f_1}^{f_2} y\right) v|\phi \ \wedge\ (e_2 - e_1) = -1))\right)$$

Nous devons montrer que $E_1 = E_2^1 \vee E_2^2 \vee E_2^3 \vee E_2^4 \vee E_2^5$. Raisonnons suivant les valeurs de $e_2 - e_1$ et $f_2 - f_1$.
- si $\nu(\mu(e_2 - e_1)) = -1$ alors E_1 est équivalent à E_2^5
- si $\nu(\mu(f_2 - f_1)) = -1$ alors E_1 est équivalent à E_2^4
- si $\nu(\mu(e_2 - e_1)) = 0$ alors E_1 est équivalent à E_2^3
- si $\nu(\mu(f_2 - f_1)) = 0$ alors E_1 est équivalent à E_2^2
- sinon, les deux intervalles contiennent au moins deux éléments distincts. E_1 contient la formule

$$\underbrace{\nu(\mu(x_{[k\leftarrow e_1]}))\ldots\nu(\mu(x_{[k\leftarrow e_2]}))}_{\nu(\mu(e_2-e_1))+1}\nu(\mu(u)) = \underbrace{\nu(\mu(y_{[l\leftarrow f_1]}))\ldots\nu(\mu(y_{[l\leftarrow f_2]}))}_{\nu(\mu(f_2-f_1))+1}\nu(\mu(v))$$

  Comme $\neg \mathit{IsPrefix}(x_{[k\leftarrow e_2+1]}, u, \phi) \ \wedge\ \neg \mathit{IsPrefix}(y_{[l\leftarrow f_2+1]}, v, \phi)$ est valide on sait d'après le théorème 25 page 110 que $\nu(\mu(x_{[k\leftarrow e_2+1]}))$ (resp. $\nu(\mu(y_{[l\leftarrow f_2+1]})))$ n'est pas un préfixe de $\nu(\mu(u))$ (resp. $\nu(\mu(v)))$. Comme les intervalles contiennent au moins deux points distincts, et que $|x| = |y|$ nous déduisons que $\nu(\mu(x_{[k\leftarrow e_1]})) = \nu(\mu(y_{[l\leftarrow e_2]}))$ et $\nu(\mu(x_{[k\leftarrow f_1]})) = \nu(\mu(y_{[l\leftarrow f_2]}))$ sont vraies. Les expressions en indice de x étant des fonctions linéaires de k qui coïncident en deux points distincts avec les expressions en indice de y, elles mêmes linéaires en l, nous pouvons conclure qu'elle coïncident partout. On peut donc simplifier l'équation en $\nu(\mu(u)) = \nu(\mu(v))$. Nous avons bien montré E_1 est équivalent à E_2^1

Nous avons bien établi la correction de la règle.
- Il reste la dernière règle. Sa correction découle immédiatement de la proposition 27 page 118.

La correction de l'algorithme est bien établie.

◇

Théorème 11 *L'algorithme de résolution de formule de similitude est correct.*

Preuve C'est une conséquence immédiate du lemme précédent. ◇

Lemme 32 *Soient deux mots paramétrés avec indices et exposants $u|\phi$ et $v|\psi$. Soit χ une forme résolue de la formule de similitude $u \equiv v|\phi \ \wedge \ \psi$. Les deux mots sont localement similaires si et seulement si*
- $\phi \ \wedge \ \psi \Rightarrow \chi$ *est une formule valide et*
- $\phi \Leftrightarrow \psi$ *est une formule valide.*

Preuve La preuve est identique à celle du lemme 22 page 80. ◇

Notons, comme dans le cas des exposants, que χ contient la contrainte $\phi \ \wedge \ \psi$. Nous aurions pu résoudre simplement la formule de similitude $u \equiv v|\top$ et énoncer le même théorème. Nous choisissons de garder $\phi \ \wedge \ \psi$ afin de simplifier la résolution de la formule de similitude. En effet cela permet de limiter le nombre de fois que *IsPrefix* sera satisfiable et donc d'empêcher un trop grand nombre d'applications de *LFact*.

6.2 Équations - Règles - Réécriture

La seule notion qui diffère dans cette section par rapport à la section 5.2 page 81 est celle de décomposition. Toutes les autres sont des généralisations immédiates des notions concernant les mots avec indices et exposants.

Décompositions d'un mot paramétré avec exposants

Dans cette section nous allons définir les préfixes, les facteurs et les suffixes d'un mot paramétré avec produits finis.

Définition 117 *Soit $u|\phi$ un mot contraint avec produit fini. On peut toujours écrire ce mot $\left(\prod_{l=e_1}^{f_1} U_1\right) \ldots \left(\prod_{l=e_k}^{f_k} U_k\right) |\psi$ où pour tout j, e_j et f_j sont des expressions arithmétiques éventuellement constantes. La* décomposition *de $u|\phi$ est l'ensemble* **fini** *$Dec(u|\phi)$ des triplets de mots contraints $(pre, sub, suf)|\psi'$ de $Dec_1 \cup Dec_2 \cup Dec_3$ définis dans les figures 6.1, 6.2 et 6.3.*

Dans cette définition on considère que U_0 et U_{k+1} désignent le mot vide ϵ et que p et p' sont des nouvelles variables. De plus nous avons omis les contraintes $p, p' \geq 0$ afin de ne pas alourdir encore la notation.

Cette notion jouit des mêmes propriétés que la décomposition des mots avec exposants.

Proposition 28 *Soit $u|\psi$ un mot contraint avec produits finis et soit D sa décomposition. Pour*
- *Toute interprétation $\nu : \mathcal{N} \to N$,*
- *Toute interprétation $\mu : Var(u|\phi) \to \mathbb{Z}$,*
- *Tous mots v_1, v_2 et v_3 tels que $v_1.v_2.v_3 \in [\![u|\psi]\!]_\nu(\mu)$,*

il existe

$$
Dec_1(u \mid \phi) = \bigcup_{\substack{0 \leq i < j \leq k+1 \\ v_1.v_2 = U_i \\ w_1.w_2 = U_j}}
\left\{
\begin{array}{l}
\left(\begin{array}{l}
\left(\prod_{l=e_1}^{f_1} U_1\right) \cdots \left(\prod_{l=e_i}^{p} U_i\right) .v_{1[l \leftarrow p+1]}, \\
v_{2[l \leftarrow p+1]}. \left(\prod_{l=p+2}^{f_i} U_i\right) \cdots \left(\prod_{l=e_j}^{p'} U_j\right) .w_{1[l \leftarrow p'+1]}, \\
w_{2[l \leftarrow p'+1]}. \left(\prod_{l=p'+2}^{f_j} U_j\right) \cdots \left(\prod_{l=e_k}^{f_k} U_k\right)
\end{array}\right) \mid \phi \ \wedge \ e_i - 1 \leq p \leq f_i - 1 \ \wedge \ e_j - 1 \leq p' \leq f_j - 1; \\
\left(\begin{array}{l}
\left(\prod_{l=e_1}^{f_1} U_1\right) \cdots \left(\prod_{l=e_i}^{p} U_i\right), \\
\left(\prod_{l=p+1}^{f_i} U_i\right) \cdots \left(\prod_{l=e_j}^{p'} U_j\right) .w_{1[l \leftarrow p'+1]}, \\
w_{2[l \leftarrow p'+1]}. \left(\prod_{l=p'+2}^{f_j} U_j\right) \cdots \left(\prod_{l=e_k}^{f_k} U_k\right)
\end{array}\right) \mid \phi \ \wedge \ e_i - 1 \leq p \leq f_i \ \wedge \ e_j - 1 \leq p' \leq f_j - 1; \\
\left(\begin{array}{l}
\left(\prod_{l=e_1}^{f_1} U_1\right) \cdots \left(\prod_{l=e_i}^{p} U_i\right), \\
\left(\prod_{l=p+1}^{f_i} U_i\right) \cdots \left(\prod_{l=e_j}^{p'} U_j\right), \\
\left(\prod_{l=p'+1}^{f_j} U_j\right) \cdots \left(\prod_{l=e_k}^{f_k} U_k\right)
\end{array}\right) \mid \phi \ \wedge \ e_i - 1 \leq p \leq f_i \ \wedge \ e_j - 1 \leq p' \leq f_j; \\
\left(\begin{array}{l}
\left(\prod_{l=e_1}^{f_1} U_1\right) \cdots \left(\prod_{l=e_i}^{p} U_i\right) .v_{1[l \leftarrow p+1]}, \\
v_{2[l \leftarrow p+1]}. \left(\prod_{l=p+2}^{f_i} U_i\right) \cdots \left(\prod_{l=e_j}^{p'} U_j\right), \\
\left(\prod_{l=p'+1}^{f_j} U_j\right) \cdots \left(\prod_{l=e_k}^{f_k} U_k\right)
\end{array}\right) \mid \phi \ \wedge \ e_i - 1 \leq p \leq f_i \ \wedge \ e_j - 1 \leq p' \leq f_j - 1;
\end{array}
\right\}
$$

FIG. 6.1 – Découpage dans deux facteurs distincts

$$
Dec_2(u \mid \phi) = \bigcup_{\substack{0 \leq i \leq k+1 \\ v_1.v_2 = U_i \\ w_1.w_2 = U_i}} \left\{ \begin{array}{l}
\left(\begin{array}{l}
\left(\prod_{l=e_1}^{f_1} U_1\right) \ldots \left(\prod_{l=e_i}^{p} U_i\right) .v_{1\,[l \leftarrow p+1]}, \\
v_{2\,[l \leftarrow p+1]} \cdot \left(\prod_{l=p+2}^{p'} U_i\right) .w_{1\,[l \leftarrow p'+1]}, \\
w_{2\,[l \leftarrow p'+1]} \cdot \left(\prod_{l=p'+2}^{f_i} U_i\right) \ldots \left(\prod_{l=e_k}^{f_k} U_k\right)
\end{array} \right) \mid \phi \ \wedge\ e_i - 1 \leq p \ \wedge\ p+1 \leq p' \ \wedge\ p'+1 \leq f_i; \\
\left(\begin{array}{l}
\left(\prod_{l=e_1}^{f_1} U_1\right) \ldots \left(\prod_{l=e_i}^{p} U_i\right), \\
\left(\prod_{l=p+1}^{p'} U_i\right) .w_{1\,[l \leftarrow p'+1]}, \\
w_{2\,[l \leftarrow p'+1]} \cdot \left(\prod_{l=p'+2}^{f_i} U_i\right) \ldots \left(\prod_{l=e_k}^{f_k} U_k\right)
\end{array} \right) \mid \phi \ \wedge\ e_i - 1 \leq p \ \wedge\ p \leq p' \ \wedge\ p'+1 \leq f_i; \\
\left(\begin{array}{l}
\left(\prod_{l=e_1}^{f_1} U_1\right) \ldots \left(\prod_{l=e_i}^{p} U_i\right), \\
\left(\prod_{l=p+1}^{p'} U_i\right), \\
\left(\prod_{l=p'+1}^{f_i} U_i\right) \ldots \left(\prod_{l=e_k}^{f_k} U_k\right)
\end{array} \right) \mid \phi \ \wedge\ e_i - 1 \leq p \ \wedge\ p \leq p' \ \wedge\ p' \leq f_i; \\
\left(\begin{array}{l}
\left(\prod_{l=e_1}^{f_1} U_1\right) \ldots \left(\prod_{l=e_i}^{p} U_i\right) .v_{1\,[l \leftarrow p+1]}, \\
v_{2\,[l \leftarrow p+1]} \cdot \left(\prod_{l=p+2}^{p'} U_i\right), \\
\left(\prod_{l=p'+1}^{f_i} U_i\right) \ldots \left(\prod_{l=e_k}^{f_k} U_k\right)
\end{array} \right) \mid \phi \ \wedge\ e_i - 1 \leq p \ \wedge\ p+1 \leq p' \ \wedge\ p' \leq f_i;
\end{array} \right\}
$$

FIG. 6.2 – Découpage dans même facteur

$$
Dec_3(u \mid \phi) = \bigcup_{\substack{0 \leq i \leq k+1 \\ v_1.v_2.v_3 = U_i}} \left\{ \left(\begin{array}{l}
\left(\prod_{l=e_1}^{f_1} U_1\right) \ldots \left(\prod_{l=e_i}^{p} U_i\right) .v_{1\,[l \leftarrow p+1]}, \\
v_{2\,[l \leftarrow p+1]}, \\
v_{3\,[l \leftarrow p+1]} \left(\prod_{l=p+2}^{f_i} U_i\right) \ldots \left(\prod_{l=e_k}^{f_k} U_k\right)
\end{array} \right) \mid \phi \ \wedge\ e_i - 1 \leq p \ \wedge\ p+1 \leq f_i; \right\}
$$

FIG. 6.3 – Découpage d'un même facteur au même indice

- *Un triplet contraint* $(pre, sub, suf)|\zeta \in D$ *et*
- *Une extension de* μ*, encore notée* $\mu : I \to \mathbb{Z}$

tels que

- $v_1 = [\![pre|\zeta]\!]_\nu(\mu)$,
- $v_2 = [\![sub|\zeta]\!]_\nu(\mu)$ *et*
- $v_3 = [\![suf|\zeta]\!]_\nu(\mu)$.

Preuve Par cas, la preuve est claire. ◇

Proposition 29 *Soit* $u|\psi$ *un mot paramétré avec indices et produits finis. Soit* $(pre, sub, suf, \zeta) \in Dec(u|\psi)$. *Pour toutes interprétations* $\nu : \mathcal{N} \to \mathbb{Z}$ *et* $\mu : I \to \mathbb{Z}$, *si*

$$\nu(\mu(\zeta)) = \top$$

alors

$$\nu(\mu(\psi)) = \top$$

et

$$\nu(\mu(pre)).\nu(\mu(sub)).\nu(\mu(suf)) = \nu(\mu(u))$$

Preuve Par cas sur la définition de *Dec*. ◇

Nous avons donc un algorithme capable de représenter de façon finie les facteurs d'un mot contraint avec produits finis. Dans la suite nous allons éliminer une partie de ces décompositions, celles qui renforcent la contrainte.

Définition 118 *Nous notons* $UDec(u|\psi)$ *l'ensemble des* décompositions uniformes *défini par*

$$UDec(u|\psi) = \{(pre, sub, suf, \zeta) \in Dec(u|\psi) \ / \ \psi \Leftrightarrow (\exists (Var(\zeta) \setminus Var(u|\psi)) \ \zeta)\}$$

Un élément de $UDec(u|\phi)$ *sera appelé* position *dans le mot* $u|\psi$.

6.3 Propriétés fondamentales

À partir des définitions de similitude locale et de décomposition uniforme, nous pouvons définir les étapes équationelles, les systèmes de réécritures, les notions de confluence et les paires critiques. Nous ne reprenons pas en détail ces définitions ainsi que leur propriétés, puisqu'elles sont exactement identiques à celles concernant les mots avec exposants. Nous pouvons donc établir les résultats fondamentaux suivant :

Théorème 12 *Soit* E *un système équationel paramétré avec indices et produits finis et* $E' = Th(E)$ *et* ν *une interprétation des paramètres. Le diagramme*

$$\begin{array}{ccc} E & \xrightarrow{Th} & E' \\ \nu \downarrow & & \downarrow \nu \\ [\![E]\!]_\nu & \xrightarrow{Th} & [\![E']\!]_\nu \end{array}$$

commute.

Théorème 13 *Soit R un système de réécriture paramétré avec indices et produits finis. Si toutes ses paires critiques sont confluentes alors toutes les ν-interprétations de R sont localement confluentes.*

Nous donnerons des exemples d'utilisation de ces théorèmes dans la partie III page 137.

6.4 Généralisations

La linéarité des contraintes est utilisée de façon essentielle dans le calcul de la formule de similitude de deux mots avec produits finis. La seule propriété que nous utilisons de la linéarité est le fait que si deux expressions linéaires coïncident en deux points distincts, elles sont égales partout. C'est un cas particulier du théorème de d'Alembert dont un corollaire est que si deux polynômes de degré n coïncident en au moins n points, ils coïncident partout. Nous pouvons donc autoriser les formules polynomiales de degré non paramétré dans les indices. Une autre généralisation possible est issue du théorème sur les multinômes de k variables et de degré 1 : ils coïncident partout s'ils coïncident en au moins k points. Nous pouvons donc autoriser les produits finis indexés par plusieurs variables, comme $\left(\prod_{0<i<j<N}^{P} a_i a_{i+j}\right)$. Ces généralisations augmentent toutefois la complexité des algorithmes. Remarquons que de toutes façons ces généralisations restent limitées par la décidabilité de la logique des contraintes.

Chapitre 7

Étude des formes normales des systèmes paramétrés

Dans ce chapitre nous allons décrire des outils permettant d'utiliser les systèmes paramétrés avec indices, exposants ou produits finis pour étudier l'ensemble des formes normales de la famille des interprétations de ces systèmes. En particulier nous allons étudier le nombre de formes normales. Nous pourrons ainsi appliquer la technique de preuve, exposée dans [11] et rappelée dans la section 2.3, pour montrer que des systèmes équationnels sont des spécification complètes de structures mathématiques dont on connaît le cardinal a priori. Si l'on ne connaît pas le cardinal de telles structures, l'étude de systèmes équationnels peut permetter de montrer qu'elles sont finies et donner dans ce cas des bornes sur leur cardinal.

7.1 Cardinal de l'ensemble des formes normales

Dans toute cette section nous fixons une signature paramétrée $\Sigma = \{a_i|\varphi\}_{\mathcal{N}}$. Pour simplifier les preuves et les notations, nous supposerons aussi que $\mathcal{N}$ est le singleton $\{N\}$. Tous nos résultats s'étendent trivialement au cas où $\mathcal{N}$ est un ensemble fini quelconque de paramètres. Nous appellerons système paramétré un système de réécriture paramétré avec indices, exposants ou produits finis.

Proposition 30 *Soit R un système paramétré. Pour tout entier $l \geq 0$, la fonction Λ_l : $\mathbb{Z} \to \overline{\mathbb{N}}$ qui à $n \in \mathbb{Z}$ associe le nombre de mots de $[\![\Sigma]\!]^*_{\nu:N\mapsto n}$ de longueur l et irréductibles par $[\![R]\!]_{\nu:N\mapsto n}$ est exprimable comme le nombre de solutions d'une contrainte de larithmétique de Presburger. Elle est donc calculable.*

Preuve Soient $n \in \mathbb{Z}$ et ν une interprétation telle que $\nu(N) = n$. Notons $\psi_l = \varphi(k_1) \wedge \cdots \wedge \varphi(k_l)$ la formule de juste contrainte de $a_{k_1} \ldots a_{k_l}$. Choisissons une famille de variables $(k_i)_{i\in\mathbb{Z}}$ dont aucune n'apparaît pas dans les règles de R. Nous allons définir récursivement la famille des formules $\Phi_l(k_1, \ldots, k_l)$ telles que le nombre de solutions distinctes de la contrainte $\Phi_l(k_1, \ldots, k_l)$ est exactement le nombre de mots irréductibles par $[\![R]\!]_\nu$ et de longueur l.

$$\Phi_0 = \bigwedge_{(g\to d|\xi)\in R} \neg(\exists Var(g \to d|\xi)\ g \equiv \epsilon|\xi)$$

$$\begin{array}{rcl}\Phi_l(k_1,\dots,k_l) & = & \Phi_{l-1}(k_1,\dots,k_{l-1}) \ \wedge\ \Phi_{l-1}(k_2,\dots,k_l) \ \wedge \\ & & \displaystyle\bigwedge_{(g\to d|\xi)\in R} \psi_l \ \wedge\ \neg\exists Var(g \to d|\xi)\ (g \equiv a_{k_1}\dots a_{k_l}|\xi)\end{array}$$

Soit $n \in \mathbb{Z}$, en posant $\nu : N \mapsto n$ nous pouvons maintenant définir

$$\Lambda_0(n) = \begin{cases} 1 \text{ si } \nu(\Phi_0) \text{ est vraie} \\ 0 \text{ sinon} \end{cases}$$

$$\Lambda_l(n) = Card\{(k_1,\dots,k_l)\ /\ \nu(\Phi_l(k_1,\dots,k_l)) \text{ vraie}\}$$

Montrons que pour tout $n \in \mathbb{Z}$, $\Lambda_l(n)$ est le nombre de mots en forme normale dans $[\![\Sigma]\!]^*_{\nu:N\mapsto n}$ pour le système $[\![R]\!]_\nu$ et de longueur l.

- Pour cela nous montrons que toute solution de $\nu(\Phi_l(k_1,\dots,k_l))$ définit une interprétation μ des variables $k_1,\dots,k_l$ telle que $[\![a_{k_1}\dots a_{k_l}|\psi_l]\!]_\nu(\mu)$ est une forme normale pour $[\![R]\!]_\nu$. Toutes les autres substitutions définissent des mots réductibles. Comme tous les mots de longueur l peuvent être vu comme étant de la forme $[\![a_{k_1}\dots a_{k_l}|\psi_l]\!]_\nu(\mu)$, nous avons bien caractérisé toutes les formes normales.
  - Soit $l = 0$. Comme $\nu(\Phi_0)$ est close, si elle est fausse alors ϵ n'est le membre gauche d'aucune règle de $[\![R]\!]_\nu$. Il est donc en forme normale et comme il est le seul mot de longueur nulle, nous avons bien un seul élément en forme normale de taille nulle.
  - Soit $l > 0$. Supposons que la propriété estvraie pour $l-1$. Soit μ telle que $\nu(\mu(\Phi_l))$ est vraie. Nous avons donc

$$\begin{array}{ll}\nu(\mu(\Phi_{l-1}(k_1,\dots,k_{l-1}))) & \text{vraie,} \\ \nu(\mu(\Phi_{l-1}(k_2,\dots,k_l))) & \text{vraie et} \\ \nu(\mu(\displaystyle\bigwedge_{(g\to d|\xi)\in R} \psi_l \ \wedge\ \neg\exists Var(l \to r|\xi)\ (g \equiv a_{k_1}\dots a_{k_l}|\xi))) & \text{vraie}\end{array}$$

    Ce qui signifie, en appliquant l'hypothèse de récurrence, que les deux facteurs de taille $l-1$ sont en forme normale. La troisième condition implique qu'aucune instance de règle ne possède le mot $[\![a_{k_1}\dots a_{k_l}|\psi_l]\!]_\nu(\mu)$ comme membre gauche. Le mot est donc en forme normale. La réciproque est claire.

  Nous pouvons conclure par récurrence.

Comme le nombre de solutions d'une formule de Presburger est calculable (même s'il est infini), nous avons donc correctement défini $\Lambda_l(n)$ pour tout $l \geq 0$ et $n \in \mathbb{Z}$. $\diamond$

Donnons quelques exemples élémentaires.

Exemple 119 Posons $\Sigma = \{a_i | 0 \leq i \leq N\}_N$.

1. Soit $R = \{a_i \to a_j | N \geq i > 0 \ \wedge \ N \geq j \geq 0\}$.

$$\begin{array}{rcl} \Phi_0 & = & \neg\exists i \exists j \ (a_i \equiv \epsilon | N \geq i > 0 \ \wedge \ N \geq j \geq 0) \\ & = & \neg\bot \\ & = & \top \end{array}$$

$$\begin{array}{rcl} \Phi_1(k_1) & = & \Phi_0 \ \wedge \ (0 \leq k_1 \leq N \ \wedge \ \neg\exists i \exists j (a_i \equiv a_{k_1} | N \geq i > 0 \ \wedge \ N \geq j \geq 0)) \\ & = & 0 \leq k_1 \leq N \ \wedge \ \neg\exists i \exists j (i = k_1 \ \wedge \ N \geq i > 0 \ \wedge \ N \geq j \geq 0) \\ \Phi_2(k_1, k_2) & = & \Phi_1(k_1) \ \wedge \ \Phi_1(k_2) \ \wedge \\ & & (0 \leq k_1, k_2 \leq N \ \wedge \ \neg\exists i \exists j (a_i \equiv a_{k_1} a_{k_2} | N \geq i > 0 \ \wedge \ N \geq j \geq 0)) \\ & = & 0 \leq k_1 \leq N \ \wedge \ \neg\exists i \exists j (i = k_1 \ \wedge \ N \geq i > 0 \ \wedge \ N \geq j \geq 0) \\ & = & \Phi_1(k_1) \ \wedge \ \Phi_1(k_2) \ \wedge \ \neg\bot \\ & = & \Phi_1(k_1) \ \wedge \ \Phi_1(k_2) \end{array}$$

Si N est positif ou nul, il n'y a que $k_1 = 0$ qui vérifie la formule $\Phi_1(k_1)$. Nous avons donc $\Lambda_1(N) = 1$ et par récurrence nous pouvons montrer que $\Lambda_l(N) = 1$ pour $N \geq 0$. Nous avons

$$\Lambda_l(N) = \begin{cases} 1 & \text{si } N \geq 0 \\ 0 & \text{si } N < 0 \end{cases}$$

Si $N < 0$ le phénomène n'est pas très intéressant puisque dans ce cas l'interprétation de la signature est vide. Nous avons décrit l'ensemble des mots qui ne contiennent que la lettre a_0.

2. Soit $R = \{a_i \to a_j | i = 0 \ \wedge \ N \geq j \geq 0$.

$$\begin{array}{rcl} \Phi_0(\epsilon) & = & \neg\exists i \exists j \ (a_i \equiv \epsilon | i = 0 \ \wedge \ N \geq j \geq 0) \\ & = & \neg\bot \\ & = & \top \end{array}$$

$$\begin{array}{rcl} \Phi_1(k_1) & = & \Phi_0(\epsilon) \ \wedge \ (0 \leq k_1 \leq N \ \wedge \ \neg\exists i \exists j (a_i \equiv a_{k_1} | i = 0 \ \wedge \ N \geq j \geq 0)) \\ & = & 0 \leq k_1 \leq N \ \wedge \ \neg\exists i \exists j (i = k_1 \ \wedge \ i = 0 \ \wedge \ N \geq j \geq 0) \\ \Phi_1(k_1, k_2) & = & \Phi_1(k_1) \ \wedge \ \Phi_1(k_2) \ \wedge \\ & & (0 \leq k_1, k_2 \leq N \ \wedge \ \neg\exists i \exists j (a_i \equiv a_{k_1} a_{k_2} | i = 0 \ \wedge \ N \geq j \geq 0)) \\ & = & 0 \leq k_1 \leq N \ \wedge \ \neg\exists i \exists j (i = k_1 \ \wedge \ i = 0 \ \wedge \ N \geq j \geq 0) \\ & = & \Phi_1(k_1) \ \wedge \ \Phi_1(k_2) \ \wedge \ \neg\bot \\ & = & \Phi_1(k_1) \ \wedge \ \Phi_1(k_2) \end{array}$$

Si N est positif ou nul, $k_1 = 1, \ldots, N$ vérifient la formule $\Phi_1(k_1)$. Nous avons donc $\Lambda_1(N) = N$ et par récurrence nous pouvons montrer que $\Lambda_l(N) = N^l$ pour $N \geq 0$. Nous avons

$$\Lambda_l(N) = \begin{cases} N^l & \text{si } N \geq 0 \\ 0 & \text{si } N < 0 \end{cases}$$

Notons que les formes normales de longueur l sont tous le mots de la forme $a_{k_1} \ldots a_{k_l}$ avec $k_j \in [1; N]$, c'est à dire les mots qui ne contiennent pas a_0 comme facteur.

Nous pouvons maintenant calculer le nombre de formes normales.

Théorème 14 *Définissons la fonction*

$$\begin{aligned} \mathcal{F} : \ & \mathbb{Z} \to \overline{\mathbb{N}} \\ & n \mapsto \mathcal{F}(n) = \sum_{l \in \mathbb{N}} \Lambda_l(n) \end{aligned}$$

Il existe un semi-algorithme tel que :

- *si $\mathcal{F}(n)$ est fini, le semi-algorithme le calcule en un temps fini et*
- *si $\mathcal{F}(n)$ est infini alors le semi-algorithme ne termine pas forcément,*

ce que nous pouvons reformuler en : "le prédicat $\mathcal{F}(n) \neq +\infty$ est semi-décidable".

Preuve S'il existe $l \geq 0$ tel que $\Lambda_l(n) = 0$ alors pour tout $l' \geq l$ nous avons $\Lambda_{l'}(n) = 0$. Nous en déduisons le semi-algorithme suivant :

Algorithme 14 (Calcul de $\mathcal{F}(n)$) **Entrée :** $n \in \mathbb{Z}$ *une valeur pour le paramètre*
Résultat : $\mathcal{F}(n) \in \overline{\mathbb{N}}$
Algorithme :

```
l <- 0;
s <- 0;
Répéter
 x <- Lambda_l(n);
 s <- s + x;
 l <- l + 1;
Tant que (x>0 et s<>infini)
Renvoyer s
```

◇

Reprenons les deux exemples précédents.

Exemple 120

1. Dans ce cas on peut vérifier que pour $n \geq 0$, $\mathcal{F}(n) = \sum_{l \in \mathbb{N}} 1 = +\infty$.
   Notons que nous sommes dans un cas où l'algorithme précédent ne termine pas puisque les termes de la série forment une suite stationnaire.
2. Ici nous avons si $n \geq 0$, $\mathcal{F}(n) = \sum_{l \in \mathbb{N}} n^l = +\infty$.

Dans ces deux cas notre semi-algorithme ne termine pas si $n \geq 0$, mais calcule bien 0 si $n < 0$.

Les système paramétrés dont le nombre de formes normales est fini pour chaque valeur du paramètre donnent des résultats bien plus intéressants , puisqu'ils peuvent permettent de définir expolicitement la fonction $\mathcal{F}$. De tels exemples seront traités dans la partie III.

7.2 Méthode d'identification de familles de structures finies

Nous proposons une méthode qui étant donnée une famille de structures algébriques dépendant de paramètres, permet de vérifier si une représentation équationnelle paramétrée (avec exposants ou produits finis) donnée l'a pour modèle. Par exemple connaissant $\mathcal{S}_N$ la famille des groupes symétriques d'ordre N, nous savons que chaque groupe a le cardinal $N!$. De plus nous savons le présenter comme une structure algébrique engendrée par une famille de générateurs : les transpositions. Ces générateurs peuvent être présentés comme les éléments d'une signature paramétrée Σ_N.

Nous pouvons donner un certain nombre de relations que les générateurs doivent vérifier et ces relation peuvent s'exprimer comme des équations paramétrées sur Σ_N. Notons E_N un ensemble de telles équations. La question que l'on se pose est celle de la complétude de cette présentation équationnelle, c'est-à-dire : y-a-t'il suffisamment de relations pour que pour tout ν, $[\![\Sigma_N]\!]^*_\nu/[\![E_N]\!]_\nu$ et $\mathcal{S}_N$ soient isomorphes ? Comme toutes les équations de E_N sont vraies dans $\mathcal{S}_N$, il est clair que S_N est un quotient de E_N. Supposons que nous connaissions un système convergent de règles paramétrées R_N tel que pour tout ν,

$$[\![\Sigma_N]\!]^*_\nu/[\![E_N]\!]_\nu = [\![\Sigma_N]\!]^*_\nu/[\![R_N]\!]_\nu$$

nous pouvons peut-être calculer le nombre de ses formes normales pour $N = n \in \mathbb{Z} : \digamma(n)$. Il ne reste plus qu'à prouver que $\digamma(n) = n!$ pour tout $n \geq 0$, et nous pourrons en déduire que $[\![\Sigma_N]\!]^*_\nu/[\![E_N]\!]_\nu$ et $\mathcal{S}_n$ sont isomorphes pour tout n.

Précisons la méthode avec le pseudo-algorithme suivant :

Algorithme 15 (Méthode d'identification) ***Entrée :***

- *A_N : une famille de structures algébriques finies,*
- *Σ_N une signature paramétrée telle que pour tout $n \in \mathbb{Z}$, $[\![\Sigma_N]\!]_{\nu:N\mapsto n}$ est une famille de générateurs de A_n,*
- *E_N un système équationnel paramétré tel que $[\![E_N]\!]_\nu$ est un ensemble d'équations vraies entre les générateurs de A_n*

  Méthode :
  - *Soit R_N un système de réécriture paramétré tel que*

  $$[\![\Sigma_N]\!]^*_\nu/[\![E_N]\!]_\nu = [\![\Sigma_N]\!]^*_\nu/[\![R_N]\!]_\nu$$

  - *Si pour tout $n \in \mathbb{Z}$, $\digamma(n) = Card(A_n)$ alors R_N est une présentation de A_N.*

Ainsi les conjectures quant aux présentations équationnelles éventuelles des structures algébrique sont ramenées à :

- la recherche d'un système de réécriture paramétré convergent R_N correpondant à E_N. Il peut être trouvé par complétion de E_N, si celle-ci réussit. Sinon, la divergence de la complétion peut être analysée pour éventuellement guider la recherche d'un système E_N plus gros.

– la recherche d'une formule arithmétique générale contenant le paramètre N qui coïncide avec $\mathcal{F}(N)$ pour toute valeur N. Cette recherche est guidée par le calcul exact pour un nombre arbitraire de valeurs du paramètre.

Pour le groupe symétrique, nous partons de

$$E_N = \begin{array}{lcl|l} a_i^2 & = & \epsilon & 1 \leq i \leq N-1 \\ a_j a_i & = & a_i a_j & 1 \leq i < j \leq N-1 \ \wedge \ j \neq i+1 \\ (a_i a_{i+1})^3 & = & \epsilon & 1 \leq i \leq N-2 \end{array}$$

En essayant de compléter E_N, nous pouvons conjecturer que le système

$$R_N = \begin{array}{lcl|l} a_i^2 & \rightarrow & \epsilon & 1 \leq i \leq n-1 \\ (a_i a_{i+1})^3 & \rightarrow & \epsilon & 1 \leq i \leq n-2 \\ a_j a_i & \rightarrow & a_i a_j & 1 \leq i < j < n \ \wedge \ j \neq i+1 \\ (\prod_{k=0}^{i-j} a_{i-k}) a_i & \rightarrow & a_{i-1}(\prod_{k=0}^{i-j} a_{i-k}) & 1 \leq j < i \leq N-1 \end{array}$$

convient. Sa confluence locale s'établit automatiquement au niveau paramétré.

Nous calculons $\mathcal{F}(n)$ pour quelques valeur de n et nous en déduisons la conjecture : $\mathcal{F}(N) = N!$. Cette conjecture peut être vérifiée “à la main”. Éventuellement un assistant de preuve comme Coq [13] peut être utilisé pour mener la preuve de façon complétement formelle puisque la conjecture peut se formaliser assez facilement.

7.3 Étude de structures plus complexes : approximations

Un certain nombre de structures mathématiques algébriques n'entrent pas dans le cadre de notre formalisme même si elles sont engendrées par des équations. Par exemple la famille des monoïdes de Burnside se définit par l'équation avec la variable x : $x^{N+P} = x^N$ c'est-à-dire

$$\mathcal{B}_{N,P} = \langle x / \{x^{N+P} = x^N\} \rangle$$

Les questions suivantes se posent :

– Par quoi est-il engendré ? Nous nous restreindrons au cas ou il est engendré par un nombre dénombrable de générateurs, sinon notre cadre d'étude ne peut s'appliquer.
– Comment exprimer dans notre formalisme l'ensemble des règles de la forme $x^{N+P} = x^N$ pour tout x ?

A priori nous pouvons essayer une approximation du monoïde en supposant qu'il est engendré dénombrablement. Nous fixons la signature paramétrée $\{a_i | i \in \mathbb{Z}\}_{\{N;P\}}$. Le système paramétré que l'on veut exprimer n'est pas fini puisque nous ne savons pas exprimer de façon finie l'ensemble de tous les mots de longueur quelconque. Par contre nous savons exprimer l'ensemble des mots d'une taille donnée. Nous pouvons donc construire une famille de système présentant des monoïdes plus “gros” que les monoïdes de Burnside, c'est-à-dire une famille de monoïdes dont un quotient est un monoïde de Burnside. Nous introduisons donc les règles

$$(a_{i_1} \dots a_{i_l})^{N+P} = (a_{i_1} \dots a_{i_l})^N$$

pour tout $l \in \mathbb{N}$ inférieur à une valeur entière fixée. En étudiant ce sytème pour différentes valeurs de l, nous avons un moyen d'étudier des approximations des monoïdes de Burnside.

Ce procédé permet d'étudier des monoïdes que l'on ne sait pas étudier directement à l'aide des techniques de réécriture.

Troisième partie

Quelques exemples non élémentaires

Dans cette partie nous allons présenter les résultats expérimentaux que nous avons obtenus. Nous allons donner quelques exemples de structures mathématiques classiques qui ont des présentations équationelles paramétrées et qui peuvent s'étudier complètement à l'aide de nos techniques. Dans quelques cas particulier nous donnerons une présentation équationelle classique, une formalisation en un système paramétré, une conjecture concernant un système convergent et enfin une étude du cardinal de l'ensemble des formes normales de ce système.

Chapitre 8

Groupes cycliques

Le groupe cyclique d'ordre N est monogène. On peut le décrire par

$$S_N = < a / \{a^N = \epsilon\} >$$

Dans CiME ce système se décrit par le fichier suivant :

```
let N = parameters "N";

let S = pword_signature N "a | {N>=1} " ;

let R = psrs S "a^{N} -> | {N>0};";
```

Nous calculons automatiquement 17 paires critiques de la forme u, v, ϕ. Elles sont toutes triviales, c'est-à-dire joignables en 0 pas de réécriture. Nous donnons ci-dessous une des paires critiques :

```
 ( a a^{q_7} , a^{p_4} ,
    { (q_7 >= 0 and
        q_7 + 2 = N and N > 0 and p_4 >= 0 and p_4 + 1 = N)
        or
       (p_4 + q_5 + 1 = N and q_7 >= 0 and p_4 >= 0 and
        not (p_6 + -1 = 0) and N > 0 and p_6 >= 0 and
        q_5 >= 0 and p_6 + -1 = q_5 and p_6 > 0 and
        p_6 + q_7 + 1 = N)
       or
       (p_6 + q_7 + 1 = N and p_4 + q_5 + 1 = N and p_6 > 0 and
        q_7 >= 0 and q_5 = 0 and p_4 >= 0 and q_5 >= 0 and
        p_6 + -1 = 0 and p_6 >= 0 and N > 0) })
```

En résolvant la première conjonction, on nous pouvons constater que cette paire décrit entre autre la paire $aa^{N-2}, a^{N-1}, N \geq 2$. Dès que la valeur de N est fixée, cette paire n'apparaît plus.

Nous avons donc prouvé la confluence locale de cette présentation de groupe cyclique d'ordre N, pour tout entier N.

Chapitre 9

Groupe symétrique S_n

On donne une description de S_n en fonction des transpositions $a_i = (i \;\; i+1)$.

La signature est $\mathcal{F} = \{a_i | 1 \leq i \leq N-1, \}_N$. Donnons une présentation équationelle classique.

$$
\begin{array}{lcll}
a_i^2 & = & \epsilon & | \;\; 1 \leq i \leq N-1 \\
a_j a_i & = & a_i a_j & | \;\; 1 \leq i < j \leq N-1 \;\wedge\; j \neq i+1 \\
(a_i a_{i+1})^3 & = & \epsilon & | \;\; 1 \leq i \leq N-2
\end{array}
$$

Nous pouvons conjecturer que le système complet suivant décrit le groupe symétrique :

$$
\begin{array}{lcll}
a_i^2 & \to & \epsilon & | \;\; 1 \leq i \leq n-1 \\
(a_i a_{i+1})^3 & \to & \epsilon & | \;\; 1 \leq i \leq n-2 \\
a_j a_i & \to & a_i a_j & | \;\; 1 \leq i < j < n \;\wedge\; j \neq i+1 \\
(\prod_{k=0}^{i-j} a_{i-k}) a_i & \to & a_{i-1}(\prod_{k=0}^{i-j} a_{i-k}) & | \;\; 1 \leq j < i \leq N-1
\end{array}
$$

La terminaison peut s'établir en utilisant la précédence $a_1 < a_2 < \ldots < a_{n-1}$ et un ordre multi-lex ou length-lex au choix.

Chapitre 10

Le groupe abélien libre à deux générateurs G

Cet exemple est adapté de [15]. Ce groupe est engendré par deux éléments x et y et leurs inverses X et Y. Les équations de ce groupes sont évidemment :

$$\begin{array}{ll} X.x & = \epsilon \\ x.X & = \epsilon \\ Y.y & = \epsilon \\ y.Y & = \epsilon \\ x.y & = y.x \end{array}$$

Ce système n'est pas convergent (une paire critique est $X.y.x, y.x$), mais la complétion de Knuth-Bendix classique termine si l'on considère l'ordre Length/Lex défini par $x < X < y < Y$. On trouve :

$$\begin{array}{ll} X.x & \rightarrow \epsilon \\ x.X & \rightarrow \epsilon \\ Y.y & \rightarrow \epsilon \\ y.Y & \rightarrow \epsilon \\ y.x & \rightarrow x.y \\ Y.x & \rightarrow x.Y \\ y.X & \rightarrow X.y \\ Y.X & \rightarrow X.Y \end{array}$$

Par contre, si l'ordre Lenght/Lex choisi est défini par $x < y < X < Y$, la complétion classique ne termine pas. Une analyse élémentaire de la divergence montre que l'algorithme n'échoue pas mais a un succès infini. Le système que l'on peut conjecturer comme le bon est le suivant :

$$
\begin{array}{ll}
X.x & \rightarrow \epsilon \\
x.X & \rightarrow \epsilon \\
Y.y & \rightarrow \epsilon \\
y.Y & \rightarrow \epsilon \\
y.x & \rightarrow x.y \\
X.y & \rightarrow y.X \\
Y.X & \rightarrow X.Y \\
Y.x & \rightarrow x.Y \\
x.y^n.X & \rightarrow y^n | n > 0 \\
y.X^n.Y & \rightarrow X^n | n > 0
\end{array}
$$

C'est un système paramétré avec exposant mais sans paramètres. CiME nous donne une paire critique non joignable : $y^n.X, X.y^n, n > 0$. Cette paire critique est en fait triviale en appliquant un argument inductif. C'est un cas où la forme normale uniforme est nécessaire pour continuer. Nous devons donc essayer une complétion. Nous pouvons proposer la généralisation suivante du système dans la syntaxe de CiME :

```
let Rcomplete = psrs S " X x -> ;
                 x X -> ;
                 Y y -> ;
                 y Y -> ;
                 y^{n} x^{p} -> x^{p} y^{n} | {n>0 /\ p>0};
                 X^{n} y^{p} -> y^{p} X^{n} | {n>0 /\ p>0};
                 Y^{n} x^{p} -> x^{p} Y^{n} | {n>0 /\ p>0};
                 y^{n} X^{p} -> X^{p} y^{n} | {n>0 /\ p>0};
                 X^{n} Y^{p} -> Y^{p} X^{n} | {n>0 /\ p>0};
                ";
```

Ce système possède 478 paires critiques paramétrées.

Chapitre 11

Groupes de Baumslag-Solitar

La famille des groupes de Baumslage-Solitar est définie par

$$G_{p,q} =< (x,y)/\{yx^p y^{-1} x^{-q}\} >$$

où p et q sont strictement positifs. Ce groupe est souvent cité comme un exemple élémentaire de groupe automatique asynchrone qui n'est pas un groupe automatique synchrone si $p \neq q$ [15].

On peut vérifier que le groupe peut être présenté par le système de règles suivant :

$$\begin{array}{lcl} X.x & \rightarrow & \epsilon \\ x.X & \rightarrow & \epsilon \\ Y.y & \rightarrow & \epsilon \\ y.Y & \rightarrow & \epsilon \\ X.y & \rightarrow & x^{q-1}.y.X^p \\ x^p.Y & \rightarrow & Y.x^q \\ X.Y & \rightarrow & x^{p-1}.Y.X^q \end{array}$$

Sa convergence peut s'étudier à l'aide de la réécriture paramétrée.

Chapitre 12

Groupes de tresses d'Artin $\mathcal{B}_n$

La famille des groupes de tresses d'Artin(voir [16] par exemple) peut se définir naturellement à l'aide d'un système équationel paramétré sur la signature $\Sigma = \{t_i, T_i | 1 \leq i \leq n-1\}_n$.

$$\begin{array}{lcll}
T_i t_i & = & \epsilon & | \ 1 \leq i \leq n-1 \\
t_i T_i & = & \epsilon & | \ 1 \leq i \leq n-1 \\
t_i t_j & = & t_j t_i & | \ |i-j| > 1 \ \wedge \ 1 \leq i, j \leq n-1 \\
t_i t_{i+1} t_i & = & t_{i+1} t_i t_{i+1} & | \ 1 \leq i \leq n-2
\end{array}$$

L'orientation de ces équations de gauche à droite est un système de réécriture qui n'est pas localement confluent. CiME calcule une paire critique non joignable :

```
t{i_31 + 1} t{i_31}
T{i} t{i_31 + 1} t{i_31} t{i_31 + 1}

 |{ (i = i_31 and 1 <= i and i_31 <= n + - 2 and
                    i <= n + - 1 and 1 <= i_31)}
```

On peut vérifier que le groupe peut être présenté par le système de règles suivant :

$$\begin{array}{lcll}
T_i t_i & \rightarrow & \epsilon & | \ 1 \leq i \leq n-1 \\
t_i T_i & \rightarrow & \epsilon & | \ 1 \leq i \leq n-1 \\
t_i t_j & \rightarrow & t_j t_i & | \ |i-j| > 1 \ \wedge \ 1 \leq i, j \leq n-1 \\
t_i t_{i+1} t_i & \rightarrow & t_{i+1} t_i t_{i+1} & | \ 1 \leq i \leq n-2 \\
T_i t_j & \rightarrow & t_j t_i T_j T_i & | \ |i-j| = 1 \ \wedge \ 1 \leq i, j \leq n-1 \\
T_i t_j & \rightarrow & t_j T_i & | \ |i-j| > 1 \ \wedge \ 1 \leq i, j \leq n-1
\end{array}$$

À cause de la troisième règle ce système ne termine pas. La vérification de la confluence de ses paires critiques ne peut donc pas utiliser la notion de forme normale. Par contre nous pouvons automatiquement calculer les 42 paires critiques et vérifier leur confluence à la main.

Un autre système de réécriture est très lié aux tresses :

$$\begin{array}{lcl|l}
T_i t_i & = & \epsilon & 1 \leq i \leq n-1 \\
T_i t_j & = & t_j T_i & |i-j| > 1 \ \wedge \ 1 \leq i,j \leq n-1 \\
T_i t_j & = & t_j t_i T_j T_i & |i-j| = 1 \ \wedge \ 1 \leq i,j \leq n-1
\end{array}$$

Ce système possède trois paires critiques dont la joignabilité est vérifiée automatiquement par CiME.

Intuitivement ce système va regrouper les t_k au début du mot en une tresse positive et les T_l à la fin en une tresse négative.

Ce système permet de décider le problème du mot pour les tresses. En effet une forme normale pour une tresse est de la forme $p.d$ où p est un mot ne comportant que des t_i et d est un mot ne contenant que des T_i. En utilisant la loi de l'inverse dans le groupe, il suffit de retourner le mot pour éliminer échanger la tresse positive et négative puis de normaliser. Par exemple, $t_j t_i T_i T_j = \epsilon$ qui est déjà normalisée, se résout en normalisant $(T_i T_j)^{-1}(t_j t_i)^{-1} = t_j t_i T_i T_j$.

Nous avons ainsi prouvé automatiquement la confluence d'un système paramétré très courant.

Chapitre 13

Groupes diédraux D_N

Le groupe de transformations des n-gones est engendré par une rotation et une symétrie. On peut montrer que

$$D_N =< (r,s)/\{s^2 = r^N = (s.r)^2 = \epsilon\} >$$

Suivant l'ordre choisi nous pouvons montrer que ce système admet un certain nombre de systèmes canoniques paramétrés.

– Avec l'ordre qui compare les longueurs d'abord puis ordonne lexicographiquement avec la précédence $r > s$ nous pouvons montrer que les présentations suivantes sont localement confluentes :

$$R_{2N} = \{ \begin{array}{ll} s^2 & \rightarrow \epsilon \\ rsr & \rightarrow s \\ r^{N+1} & \rightarrow sr^{N-1}s \\ r^N s & \rightarrow sr^N \} \end{array}$$

$$R_{2N+1} = \{ \begin{array}{ll} s^2 & \rightarrow \epsilon \\ rsr & \rightarrow s \\ r^{N+2} & \rightarrow sr^{N-1}s \\ sr^{N+1} & \rightarrow r^N s \\ r^{N+1}s & \rightarrow sr^N \\ sr^N s & \rightarrow r^{N+1} \} \end{array}$$

– Avec l'ordre multi-ensemble engendré par l'ordre lexicographique avec la précédence $s > r$:

$$R_{2n} = \{ \begin{array}{ll} s^2 & \rightarrow \epsilon \\ rsr & \rightarrow s \\ r^{n+1} & \rightarrow sr^{n-1}s \\ r^n s & \rightarrow sr^n \} \end{array}$$

$$R_{2n+1} = \{ \begin{array}{ll} s^2 & \rightarrow \epsilon \\ rsr & \rightarrow s \\ r^{n+1} & \rightarrow sr^n s \} \end{array}$$

- Avec l'ordre multi-ensemble engendré par l'ordre lexicographique avec la précédence $s < r$:

$$R_{2n} = \{ \begin{array}{ll} s^2 & \rightarrow \epsilon \\ rsr & \rightarrow s \\ r^{2n} & \rightarrow \epsilon \\ sr^is & \rightarrow r^{2n-i} | 1 \leq i \leq n-1 \\ r^{n+1}s & \rightarrow sr^{n-1} \\ sr^n & \rightarrow r^ns \} \end{array}$$

  Ce système contient $n + 4$ règles.

$$R_{2n+1} = \{ \begin{array}{ll} s^2 & \rightarrow \epsilon \\ rsr & \rightarrow s \\ r^{2n+1} & \rightarrow \epsilon \\ sr^is & \rightarrow r^{2n+1-i} | 1 \leq i \leq n \\ sr^{n+1} & \rightarrow r^ns \\ r^{n+1}s & \rightarrow sr^n \} \end{array}$$

  Ce système contient $n + 5$ règles.

Quatrième partie

Conclusion

Dans ce manuscrit, nous avons proposé un formalisme permettant d'augmenter de façon significative le champ d'application des techniques de réécriture. Dans ce formalisme nous pouvons exprimer des propriétés *uniformes* de familles de systèmes de réécritures qui dépendent de paramètres entiers. Ces propriétés se transposent dans les familles de structures mathématiques représentées par les systèmes de réécriture que nous considérons. La propriété que nous avons étudiée le plus précisément est la confluence, prérequis d'un grand nombre d'applications, en particulier de la résolution du problème du mot. Nous avons aussi étudié le problème de la terminaison de nos systèmes sans toutefois entrer dans le problème de la recherche d'ordres paramétrés stables par interprétation.

Les systèmes paramétrés sont très courants dans la littérature et de nombreuses études ad-hoc avaient été menées les concernant. Notre formalisme permet d'unifier l'étude de ces familles. Dans un premier temps nous avons vérifié que les résultats antérieurs concernant ces familles peuvent se retrouver à l'aide de nos techniques. Dans un second temps nous nous sommes attachés à démontrer que nos techniques sont utilisables pour étudier des problèmes complexes issus de la théorie des groupes et monoïdes finis. En effet notre formalisation permet de mener des expériences qui peuvent éclairer la compréhension des structures de familles de groupes en passant par l'étude de leurs classes d'équivalences, de leur noyau, de leurs constantes de structures, de leurs quotients et sous-groupes distingués. Un autre champ d'application fondamental est celui des preuves automatiques dans les spécifications algébriques, dans lesquelles des paramètres entiers apparaissent naturellement. Plus généralement encore, nos techniques permettent d'aborder le problème de l'automatisation des preuves des programmes écrits sous forme de règles de réécriture, comme on peut en trouver dans les logiciels Maude [7] et ELAN [3] par exemple. Dans ce cadre, nos techniques permettent d'étudier la confluence de programmes dépendants de paramètres qui peuvent être des arguments du programme ou de ses modules. Le problème des N reines, exemple paradigmatique dans la programmation par réécriture, pourrait être étudié uniformément par rapport à N. L'étude de cette application de nos techniques est une des plus prometteuses perspectives de notre travail. Une autre perspective se trouve dans l'étude des programmes utilisant des systèmes de types dépendants. La confluence de tels programmes est fondamentale et souvent difficile à établir. Nos techniques permettent de formaliser le problème et même d'automatiser sa résolution dans les cas favorables. Elles peuvent aussi guider la preuve de confluence, en transformant le problème en un certain nombre de problèmes "plus simples" à résoudre. Dans le cadre des assistants de preuves comme Coq, cette technique s'intègre pleinement. De nombreuses expériences sont encore à mener dans ce domaine et notre implantation développée au sein du logiciel CiME semble le parfait laboratoire pour les conduire.

Bibliographie

[1] G. Birkhoff. On the structure of abstract algebras. In *Proc. Cambridge Phil. Society, 31*, 1935.

[2] R.V. Book and F. Otto. *String Rewriting Systems.* Text and monographs in Computer Science. Springer-Verlag, Berlin, 1993.

[3] P. Borovanský, C. Kirchner, H. Kirchner, P.-E. Moreau, and M. Vittek. ELAN : A logical framework based on computational systems. In J. Meseguer, editor, *Proceedings of the 1st International Workshop on Rewriting Logic and its Applications, Electronic Notes in Theoretical Computer Science 4*, 1996.

[4] Alexandre Boudet and Hubert Comon. Diophantine equations, Presburger arithmetic and finite automata. In H. Kirchner, editor, *Proc. Coll. on Trees in Algebra and Programming (CAAP'96)*, Lecture Notes in Computer Science, pages 30–43, 1996.

[5] Bruno Buchberger. *An Algorithm for Finding a Basis for the Residue Class Ring of a Zero-Dimensional Ideal.* PhD thesis, University of Innsbruck, Austria, 1965. (in German).

[6] H. Chen, Jieh Hsiang, and H.-C. Kong. On finite representations of infinite sequences of terms. In Stéphane Kaplan and Mitsuhiro Okada, editors, *Proc. CTRS 90*, Montreal, 1990.

[7] M. Clavel, S. Eker, P. Lincoln, and J. Meseguer. Principles of maude. In J. Meseguer, editor, *Proceedings of the 1st International Workshop on Rewriting Logic and its Applications, Electronic Notes in Theoretical Computer Science 4*, 1996.

[8] H. Comon. Completion of rewrite systems with membership constraints. Part I : deduction rules. *Journal of Symb. Computation*, 25 :397–419, 1998. This is a first part of a paper whose abstract appeared in Proc. ICALP 92, Vienna. To appear in the Journal of Symbolic Computation.

[9] H. Comon. Completion of rewrite systems with membership constraints. Part II : Constraint solving. *Journal of Symb. Computation*, 25 :421–453, 1998. This is the second part of a paper whose abstract appeared in Proc. ICALP 92, Vienna.

[10] Hubert Comon. On unification of terms with integer exponents. *Math. Systems Theory*, 28 :67–88, 1995.

[11] Evelyne Contejean, Antoine Coste, and Benjamin Monate. Rewriting techniques in theoretical physics. In Leo Bachmair, editor, *11th International Conference on Rewri-*

ting Techniques and Applications, volume 1833 of *Lecture Notes in Computer Science*, pages 80–94, Norwich, UK, July 2000. Springer-Verlag.

[12] Evelyne Contejean, Claude Marché, Benjamin Monate, and Xavier Urbain. Cime version 2, 2000. Prerelease available at `http://cime.lri.fr/`.

[13] Coq. The Coq Proof Assistant, 2001. `http://coq.inria.fr/`.

[14] M. Dehn. Uber unendliche diskontinuierliche gruppen. *Mathematische Ann.*, 71 :116–144, 1911.

[15] D. Epstein, J. Cannon, D. Holt, S. Levy, M. Paterson, J. Thurst, and J. Bartlett. *Word Processing in Groups.* 1992.

[16] F. A. Garside. The braid group and other groups. *Quarterly Journal Mathematics Oxford*, 20 :235–254, 1969.

[17] Deepak Kapur and Paliath Narendran. A finite Thue system with decidable word problem and without equivalent finite canonical system. *Theoretical Computer Science*, 35 :337–344, 1985.

[18] Claude Kirchner and Hélène Kirchner. Constrained equational reasoning. In *Proceedings of the ACM-SIGSAM 1989 International Symposium on Symbolic and Algebraic Computation, Portland (Oregon)*, pages 382–389. ACM Press, 1989.

[19] Donald E. Knuth and Peter B. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, 1970.

[20] Philippe Le Chenadec. *Canonical forms in finitely presented algebras.* Pitman, London, 1986.

[21] Philippe Le Chenadec. Analysis of dehn's algorithm by critical pairs. *Theoretical Computer Science*, 51(1-2) :27–52, 1987.

[22] Claude Marché. Normalized rewriting : an alternative to rewriting modulo a set of equations. *Journal of Symbolic Computation*, 21(3) :253–288, 1996.

[23] Claude Marché. Normalized Rewriting : an unified view of Knuth-Bendix completion and Gröbner bases computation. *Progress in Computer Science and Applied Logic*, 15 :193–208, 1998.

[24] Benjamin Monate. Automates de formes normales et réductibilité inductive. Rapport de DEA, Université Paris-Sud, Orsay, France, September 1997.

[25] Benjamin Monate. Automates de formes normales et réductibilité inductive. In *Journées du pôle Contraintes et programmation logique*, pages 21–31, Rennes, nov 1997. PRC/GDR Programmation du CNRS.

[26] M. H. A. Newman. On theories with a combinatorial definition of 'equivalence'. *Ann. Math.*, 43(2) :223–243, 1942.

[27] John Pedersen. *Confluence Methods and the Word Problem in Universal Algebra.* PhD thesis, Emory University, 1984.

[28] Emil L. Post. Recursive unsolvability of a problem of Thue. *Journal of Symbolic Logic*, 13 :1–11, 1947.

[29] M. Presburger. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. In *Comptes Rendus du premier Congrès des Mathématiciens des Pays slaves*, Warszawa, 1929.

[30] C. C. Squier. Word problems and a homological finiteness condition for monoids. *Journal of Pure and Applied Algebra*, 49 :201–217, 1987.