Mémoire de stage de DEA
effectué au LSV :
Laboratoire Spécification et Vérification
Ecole Normale Supérieure de Cachan
61, av. du Pdt Wilson,
94235 CACHAN Cedex.

Vérification de Systèmes à Compteurs

Mémoire présenté par Véronique Cortier
Sous la direction de
Hubert Comon et Alain Finkel

Année 1999

Table des matières

Introduction

0.1 Contexte

De nos jours, les systèmes informatiques sont impliqués dans presque tous les objets de la vie courante et surtout, ce sont de plus en plus souvent des systèmes informatiques qui contrôlent des processus à la place de l'homme. Ainsi, comme l'explique B. Boigelot dans sa thèse ([4]), le pilotage des avions gros porteurs dépend plus des ordinateurs embarqués que du pilote à bord. Et de nombreux cas de graves perturbations, parfois fatales, proviennent de problèmes informatiques. Ceci est encore plus vrai lorsqu'il s'agit de construire du matériel destiné à des missions spatiales.

Aussi, la fiabilité des logiciels (absence de "bugs" graves) est une notion de plus en plus importante. Il s'agit donc de vérifier qu'un logiciel fait bien ce qu'on lui demande et ne va pas déclencher une situation catastrophique : c'est la vérification de logiciels.

Les méthodes de vérification automatisées (ou assistées par ordinateur) s'appuient nécessairement sur des descriptions formelles à la fois du logiciel et des propriétés qu'on en attend. On distingue habituellement deux types de méthodes automatisées de vérification.

La preuve : le logiciel est spécifié dans une logique appropriée. Il s'agit alors de prouver que la propriété attendue est une conséquence logique de la spécification. On utilise à cet effet des démonstrateurs automatiques ou assistant de preuve.

Le "model-checking" : le programme est ici formalisé comme une interprétation possible de la logique dans laquelle sont décrites les propriétés attendues. Les "model-checkers" décident alors si cette interprétation est un modèle (ou non) des propriétés attendues.

La formalisation la plus courante des programmes est celle de l'automate : le programme est modélisé par un système d'états finis et pour lequel on ne peut effectuer qu'un nombre fini d'actions. Il s'agit du modèle de calcul le plus simple étudié depuis longtemps par les informaticiens (cf [3] par exemple). D'un point de vue logique, un automate est une structure finie dans laquelle sont interprétées des relations binaires $R_{G_1}, \cdots, R_{G_n}$. La spécification des systèmes réactifs, comme l'énoncé des propriétés qu'ils doivent vérifier, s'effectue traditionnellement par le biais de logiques temporelles.

Les logiques temporelles (propositionnelles) sont des logiques modales dans lesquelles, sans parler du temps explicitement, il est possible d'exprimer qu'un énoncé est "toujours vrai" (modalité G : always), ou "sera vrai un jour" (modalité F : eventually), ou encore que ϕ est vraie jusqu'à ce que ψ le soit ($\mathcal{U}$: until). Plusieurs sémantiques sont possibles : ainsi les formules de LTL (logique temporelle linéaire) sont construites à partir d'un ensemble de variables propositionnelles et sont closes par opérations booléennes et l'application du connecteur logique temporel X ("next" qui signifie que la formule sera vraie à l'étape suivante) et $\mathcal{U}$ (until).

Une autre logique : CTL (logique temporelle d'arbres) permet un contrôle sur les chemins que l'on suit dans l'automate : elle comporte les opérateurs A ("pour tous les chemins") et E ("il existe un chemin"). Les formules de CTL sont construites à partir d'un ensemble de variables propositionnelles et sont closes par les opérations booléennes et l'application des opérateurs AX, EX, $A\mathcal{U}$, $E\mathcal{U}$ où X désigne toujours "next" et $\mathcal{U}$ "until".

Ces logiques sont décidables, et la vérification de modèles est d'une complexité relativement faible : PSPACE-complet pour LTL (voir [18]), polynômiale pour CTL (voir [17]). Ces résultats sont bien compris aujourd'hui comme le montre M. Vardi dans [19] : on peut associer, par exemple, à chaque formule de LTL, un automate de Büchi qui accepte l'ensemble des modèles de la formule.

0.2 Limite des résultats existants

Cependant, modéliser un système par un ensemble fini d'états possède des inconvénients : d'une part, même si la plupart des systèmes réellement constructibles sont finis, leur taille peut dépasser largement ce qu'il est possible de traiter dans le cadre fini, et il est alors plus réaliste de les modéliser par des systèmes infinis d'états. D'autre part, il est intéressant de raisonner indépendamment d'une borne qui limiterait à l'avance la taille des systèmes considérés.

On considère alors d'autres modèles avec un nombre d'états infini comme par exemple, les automates à compteurs : un état du système est donné par un état de contrôle (les états de contrôle sont en nombre fini) et un vecteur de nombres (entiers ou réels) représentant les différents paramètres du système à l'état considéré (vitesse, temps, température, ...).

Ce modèle d'automate à compteurs est cependant trop expressif (on peut simuler une machine de Minsky) pour que l'on puisse trouver des propriétés générales vérifiables sur n'importe quel automate à compteurs. Il y a plusieurs façons de restreindre les automates à compteurs : réseaux de Petri, automates temporisés, ... On s'est aussi aperçu que le fait de ne pas avoir plusieurs boucles imbriquées dans un automate (on dit alors que l'automate est plat) est une propriété qui élargit beaucoup les possibilités de vérification (voir [6]). Nous allons ainsi nous demander quelles propriétés temporelles sont décidables dans ce contexte, est-ce que l'on peut donner une version logique (i.e. une logique décidable qui exprime cependant des propriétés sur les compteurs) de ce résultat de ces résultats de décidabilité sur les automates plats.

0.3 Contributions

Nous nous sommes surtout intéressés à des systèmes qui possèdent justement cette notion de platitude, et ceci sous deux angles différents.

L'accessibilité est une propriété particulière des programmes ayant une grande importance pratique. Ainsi, les problèmes de sécurité s'expriment souvent comme l'inaccessibilité d'une configuration catastrophique. Elle est décidable dans les réseaux de Petri mais il est difficile d'en connaître la complexité. Aussi, on s'est intéressé à un problème d'accessibilité "restreinte" (peut-on aller de l'état U à l'état V ?) pour un système qui permet des transitions plus générales que les réseaux de Petri mais qui possède la propriété d'être plat. Les transitions autorisées sont tous les contrôles qui peuvent s'écrire comme une fonction affine. Cette famille de transitions est incomparable (ni plus petite, ni plus grande) avec celle utilisée par H.

Comon et Y. Jurski qui ont montré (dans [6]) que pour leur famille de transitions, la relation d'accessibilité est expressible dans l'arithmétique de Presburger et donc a fortiori décidable.

La première partie de ce mémoire est consacrée à l'étude de l'accessibilité pour ce modèle plat avec transitions affines. Je montre tout d'abord que, même si les transitions sont plus générales que celles des réseaux de Petri, l'accessibilité est décidable dans le cas où il n'y a qu'une ou deux fonctions affines (cf parties 1.2 et 1.3). Ces algorithmes, bien qu'effectifs ont une complexité que je n'arrive pas à déterminer, aussi je donne ensuite une autre méthode dans le cas d'une seule fonction affine qui permet de dériver une borne de complexité à l'algorithme de décision (partie 1.6). Ensuite, je montre que malgré la restriction de platitude, l'accessibilité dans le cas général est indécidable (partie 1.4). Enfin, je donne quelques exemples de restrictions sur les fonctions affines qui permettent de garantir la décidabilité de l'accessibilité même dans le cas où il y a un nombre arbitraire de fonctions affines.

Dans la deuxième partie du mémoire, je vais m'intéresser à la relation qu'il peut y avoir entre des logiques temporelles et les automates à compteurs. En effet, si le lien entre logique temporelle et automates finis est bien compris (d'où découle la décision des logiques temporelles et de la vérification de modèles), il l'est moins, voire pas du tout, dans le cas des automates à compteurs. Par ailleurs, on peut remarquer que les automates associés aux formules de logique temporelle (des automates alternants pour LTL, voir [19]) sont particuliers : ils ont une propriété qui s'apparente à la platitude. Cette ressemblance est le point de départ de la deuxième partie du mémoire.
On va, en effet, commencer par exposer une construction qui permet de passer d'un automate plat à compteurs à des formules d'une logique temporelle que l'on définira (partie 2.2.1). On exploitera ensuite les résultats de H. Comon et Y. Jurski, qui montrent que la relation d'accessibilité dans les automates plats à compteurs est expressible dans l'arithmétique de Presburger, pour montrer la décidabilité de cette logique et du model-checking (sous certaines conditions). Enfin, on étendra cette logique à une logique qui contient tout LTL (portant sur des variables propositionnelles) et qui reste décidable.

Chapitre 1

Accessibilité restreinte dans un système plat

On considère dans ce chapitre des fonctions affines de la forme $f(X) = AX + B$ où A est une matrice à coefficients entiers positifs et B un vecteur d'entiers relatifs. De plus f est une fonction de $\mathbb{N}^k$ à valeurs dans $\mathbb{N}^k$. (voir exemple (1))

On veut savoir si étant donnés $f_1, \ldots, f_p$, ($f_i(X) = A_iX + B_i$), U et V, deux vecteurs d'entiers positifs, est-ce que $V \in \{f_p^{n_p} \ldots f_1^{n_1}(U) | n_1, \ldots, n_p \in \mathbb{N}\}$ est décidable ?

Il est important de noter que l'on itère les fonctions successivement : on ne peut pas revenir en arrière, ce qui correspond à une notion d'automates plats.

Les transitions permises (du type $f(X) = AX + B$) généralisent les réseaux de Petri : si $A = Id$, on obtient un réseau de Petri; cependant, les séquences de transitions d'un réseau de Petri ne s'écrivent en général pas toutes $f_1^* \cdots f_p^*$.

Ces fonctions affines de transition sont aussi plus générales que beaucoup d'autres transitions étendues, étudiées dans [12] mais avec la restriction de toujours "avancer" dans l'automate : voir exemples d'autres fonctions de transitions sur la figure 1.1.

1.1 Préliminaires

On considère l'ordre partiel sur les vecteurs d'entiers suivant :

Définition 1 *Soient* $U, V \in \mathbb{N}^k$,

$U \leq V$ *si et seulement si* $\forall 1 \leq i \leq k, \quad U_i \leq V_i$.

On dira que $U \leq V$ si et seulement si U a toutes ses coordonnées plus petites que celles de V.

Proposition 1 $\leq$ *est un beau préordre.*

Preuve : voir [9]. ■

Définition 2 *On notera* $V \tilde{>} U$ *si l'une des coordonnées au moins de* V *est plus grande que celle de* U. *(En fait si* $U \ngeq V$.*)*

Proposition 2 *Si* $V_1 \geq V_2$ *et* $V_2 \tilde{>} U$ *alors* $V_1 \tilde{>} U$.

Exemple 1 : Fonction Multiplication

$$f(X) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} X + \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}$$

$$f\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} a-1 \\ b \\ c+b \end{pmatrix} \text{ si } a \geq 1, \quad f^n\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} a-n \\ b \\ c+nb \end{pmatrix} \text{ si } a \geq n$$

Si $a = 0$, $f\begin{pmatrix} a \\ b \\ c \end{pmatrix}$ n'est pas défini ! En effet, $f(X)$ ne peut devenir négatif.

Cette fonction f ainsi définie réalise la multiplication de a par b. En effet

$$\exists n \quad f^n\begin{pmatrix} a \\ b \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ b \\ d \end{pmatrix} \text{ si et seulement si } d = ab$$

Extensions de réseaux de Petri	Fonctions affines associées
Réseau de Petri	$A + Id$
Double réseau de Petri	$Id \leq A \leq 2Id$
Réseau de Petri avec transfert généralisé	$0 \leq A \quad \forall j A_j \neq 0$
Réseau de Petri avec reset	$0 \leq A \leq Id$

FIG. 1.1 – *Exemple de classes de transitions comprises dans $f(X) = AX + B$ étudiées dans [12]*

Preuve : V_2 a l'une de ses coordonnées plus grande que celle de U, V_1 a toutes ses coordonnées plus grandes que celles de V_2, donc a fortiori, V_1 a l'une de ses coordonnées plus grande que celle de U. ∎

Proposition 3 *Soient $f(X) = AX + B$, $A \in M_k(\mathbb{N})$, $B \in \mathbb{Z}^k$.*
Si $U_1 \leq U_2$ alors $f(U_1) \leq f(U_2)$. On dit que f est "monotone"pour l'ordre $\leq$.

Preuve : Si $U_1 \leq U_2$ alors $AU_1 \leq AU_2$ car A est une matrice à coefficients entiers positifs et donc $AU_1 + B \leq AU_2 + B$, i.e., $f(U_1) \leq f(U_2)$. ∎

Définition 3 *On appelle* taille *de V, notée $|V|$, la somme des valeurs absolues des coordonnées de V.*

Remarque : si V est un réel, $|V|$ est sa valeur absolue.

1.2 Décidabilité pour une fonction

En utilisant la proposition 1, à savoir que $\leq$ est un beau préordre sur $\mathbb{N}^k$, on peut décider la question posée lorsque $p = 1$ en procédant par extraction.

Théorème 1 *Soient $f(X) = AX + B$, $A \in M_k(\mathbb{N})$, $B \in \mathbb{Z}^k$, et $U, V \in \mathbb{N}^k$. L'accessibilité restreinte $V \in \{f^n(U) | n \in \mathbb{N}\}$ est décidable.*

Preuve : On considère la suite des $U, f(U), f^2(U), \ldots, f^n(U), \ldots$

Soit elle devient non positive à partir d'un certain rang N, on ne peut plus itérer la fonction donc $\{f^n(U)|n \in \mathbb{N}\} = \{f^n(U)|n \leq N\}$, on peut donc tester si $V \in \{f^n(U)|n \in \mathbb{N}\}$.

Soit il existe $N_1 < N_2$ tels que $f^{N_1}(U) \leq f^{N_2}(U)$ car $\leq$ est un beau préordre (proposition 1) donc il y a au moins deux éléments comparables. Soit $l = N_2 - N_1$. Pour tout $0 \leq i \leq l-1$, $f^{N_1}(U) \leq f^{N_2}(U) \Rightarrow f^i(f^{N_1}(U)) \leq f^i(f^{N_2}(U))$ par monotonie de f (proposition 3), i.e. $f^{N_1+i}(U) \leq f^{N_2+i}(U)$.

On a donc :

$$\begin{cases} f^{N_1}(U) \leq f^{N_1+l}(U) \leq f^{N_1+2l}(U) \leq \ldots \leq f^{N_1+kl}(U) \leq \ldots \\ f^{N_1+1}(U) \leq f^{N_1+1+l}(U) \leq f^{N_1+1+2l}(U) \leq \ldots \leq f^{N_1+1+kl}(U) \leq \ldots \\ \vdots \\ f^{N_1+(l-1)}(U) \leq f^{N_1+(l-1)+l}(U) \leq f^{N_1+(l-1)+2l}(U) \leq \ldots \leq f^{N_1+(l-1)+kl}(U) \leq \ldots \end{cases}$$

- soit l'une de ces l suites a deux termes consécutifs égaux, alors cette suite est stationnaire et donc toutes les suites sont stationnaires, $\{f^n(U)|n \in \mathbb{N}\}$ est alors un ensemble fini, on peut donc tester si $V \in \{f^n(U)|n \in \mathbb{N}\}$.
- Soit toutes les suites sont strictement croissantes et on calcule chacune d'entre elles jusqu'à ce qu'au moins l'une des coordonnées dépasse celle de V.

Concrètement, l'algorithme est le suivant : on calcule successivement $U, f(U), f^2(U), \ldots, f^n(U), \ldots$ (en testant à chaque fois si c'est égal à V), jusqu'à trouver deux termes comparables, à moins que la suite ne soit finie (si on sort du domaine de la fonction). Dans ce dernier cas, l'algorithme est terminé, on n'a pas trouvé V. Sinon, on calcule successivement $f^{N_1}(U) \leq f^{N_1+l}(U) \leq f^{N_1+2l}(U) \leq \ldots \leq f^{N_1+kl}(U) \leq \ldots$ jusqu'à ce qu'au moins l'une des coordonnées dépasse celle de V, à moins que la suite ne devienne stationnaire (dès que l'on trouve deux termes consécutifs égaux), dans ce cas $\{f^n(U)|n \in \mathbb{N}\}$ est alors un ensemble fini, on teste si V est dedans, sinon, on recommence avec $f^{N_1+1}(U) \leq f^{N_1+1+l}(U) \leq f^{N_1+1+2l}(U) \leq \ldots \leq f^{N_1+1+kl}(U) \leq \ldots$ et ainsi de suite pour les l suites.

Cependant, même si cet algorithme est effectif, on n'a pas obtenu de borne sur sa complexité. On présentera au paragraphe 1.6 un autre algorithme permettant de décider si $V \in \{f^n(U)|n \in \mathbb{N}\}$, plus compliqué à décrire, mais dont on peut donner une borne de la complexité, malheureusement cette borne, certes élémentaire, est très élevée (une tour d'exponentielles). ∎

1.3 Décidabilité pour deux fonctions

Le résultat reste vrai quand on itère deux fonctions successivement.

Théorème 2 *Soient $f(X) = AX+B$, $g(X) = CX+D$ avec A et C des matrices à coefficients entiers positifs, B et D des vecteurs d'entiers relatifs et U, V, deux vecteurs d'entiers positifs.*

$V \in \{g^n f^m(U)|n, m \in \mathbb{N}\}$ est décidable.

La démonstration de ce théorème est assez technique et se fait en plusieurs temps :
- On cherche à contrôler la suite $(g^n f^m(U))_n$ à m fixé.
- On cherche à contrôler la suite $(g^n f^m(U))_m$ à n fixé.
- On démontre le théorème.

1.3.1 Contrôle de la suite $(g^n f^m(U))_n$ à m fixé

On reprend de façon précise la démonstration du théorème 1 pour contrôler $(g^n f^m(U))_n$.

Lemme 1 *Soient $f(X) = AX + B$, $g(X) = CX + D$ avec A et C des matrices à coefficients entiers positifs, B et D des vecteurs d'entiers relatifs.*

Etant donnés m, U, V, deux vecteurs d'entiers positifs, on peut construire (algorithmiquement) n_m tel que :

1. *ou bien $\forall n \geq n_m \quad g^n f^m(U) \tilde{>} V$,*
2. *ou bien on peut construire (algorithmiquement) N_m, l_m tels que*
   $\forall n \geq n_m \quad \forall i < l_m \quad g^{N_m+i+nl_m} f^m(U) = g^{N_m+i+n_m l_m} f^m(U)$,
   c'est à dire que $\forall n \geq N_m + (n_m+1)l_m \quad g^n f^m(U) = g^{N'_n+i} f^m(U)$ pour un certain $i < l_m$, avec $N'_n = N_m + n_m l_m$,
3. *ou bien pour tout $n \geq n_m$, $g^n f^m(U)$ est non défini (passage par un vecteur non positif).*

Preuve : On reprend pas à pas la démonstration du théorème 1 en posant $U' = f^m(U)$.
On considère la suite des $U', g(U'), g^2(U'), \ldots, g^n(U'), \ldots$.
Soit elle devient non positive à partir d'un certain rang n_m, c'est le cas 3.
Soit il existe $N_1 < N_2$ tels que $g^{N_1}(U) \leq g^{N_2}(U)$, on considère alors les suites :

$$\begin{cases} g^{N_1}(U) \leq g^{N_1+l}(U) \leq \ldots \leq g^{N_1+kl}(U) \leq \ldots \\ g^{N_1+1}(U) \leq g^{N_1+1+l}(U) \leq \ldots \leq g^{N_1+1+kl}(U) \leq \ldots \\ \vdots \\ g^{N_1+(l-1)}(U) \leq g^{N_1+(l-1)+l}(U) \leq \ldots \leq g^{N_1+(l-1)+kl}(U) \leq \ldots \end{cases}$$

- soit l'une de ces l suites a deux termes consécutifs égaux, alors cette suite est stationnaire et donc toutes les suites sont stationnaires, c'est le cas 2.,
- Soit toutes les suites sont strictement croissantes et on calcule chacune d'entre elles jusqu'à ce qu'au moins l'une des coordonnées dépasse celle de V et on arrive au cas 1.

■

Corollaire 1 *Pour m fixé, on peut décider si $V \in \{g^n f^m(U) | n, m \in \mathbb{N}\}$.*

Remarque : ce résultat est une conséquence du théorème 1 en posant $U' = f^m(U)$.

1.3.2 Contrôle de la suite $(g^n f^m(U))_m$ à n fixé

Pour contrôler $(g^n f^m(U))_m$, on commence par établir un lemme qui nous servira fréquemment pour assurer la décidabilité de la procédure.

Lemme 2 *Soient $f(X) = AX + B$, $g(X) = CX + D$ avec A et C des matrices à coefficients entiers positifs, B et D des vecteurs d'entiers relatifs, k la dimension de l'espace. Soient*

$U \in \mathbb{N}^k$, $n \in \mathbb{N}$ et $l > 0, l \in \mathbb{N}$, on considère la suite
$g^n(U), g^n f^l(U), g^n f^{2l}(U), \dots, g^n f^{ml}(U), \dots$ (n fixé), alors :

ou bien la suite est stationnaire (à partir d'un certain rang), ou bien elle ne stationne jamais plus de $k+1$ pas.

Preuve : La démonstration de ce lemme est simple et utilise principalement des propriétés élémentaires d'algèbre.

Supposons que la suite stationne plus de $k+1$ pas : $g^n f^{m_0 l}(U) = g^n f^{(m_0+1)l}(U) = \dots = g^n f^{(m_0+k+1)l}(U)$.

Montrons que $g^n f^{(m_0+k+2)l}(U) = g^n f^{m_0 l}(U)$, ce qui prouvera par récurrence que la suite est stationnaire.

$f^{(m_0+1)l}(U) - f^{m_0 l}(U), \dots, f^{(m_0+k+1)l}(U) - f^{m_0 l}(U)$ sont $k+1$ vecteurs du $\mathbb{Q}$ espace vectoriel $\mathbb{Q}^k$, espace de dimension k, ils sont donc liés dans $\mathbb{Q}$ et donc dans $\mathbb{Z}$ (en multipliant par un entier adéquat). Donc

$$\exists q_1, \dots, q_{k+1} \in \mathbb{Z} \quad \sum_{i=1}^{k+1} q_i (f^{(m_0+i)l}(U) - f^{m_0 l}(U)) = 0.$$

Soit N le plus grand i tel que q_i est non nul ($1 \leq N \leq k+1$).

$$q_N (f^{(m_0+N)l}(U) - f^{m_0 l}(U)) + \sum_{i=1}^{N-1} q_i (f^{(m_0+i)l}(U) - f^{m_0 l}(U)) = 0.$$

On prend l'image par f^{k+2-N} :

$$q_N (f^{(m_0+k+2)l}(U) - f^{(m_0+k+2-N)l}(U))$$
$$+ \sum_{i=1}^{N-1} q_i (f^{(m_0+i+k+2-N)l}(U) - f^{(m_0+k+2-N)l}(U)) = 0,$$

et donc

$$q_N g^n (f^{(m_0+k+2)l}(U) - f^{(m_0+k+2-N)l}(U))$$
$$+ \underbrace{\sum_{i=1}^{N-1} q_i g^n (f^{(m_0+i+k+2-N)l}(U) - f^{(m_0+k+2-N)l}(U))}_{=0} = 0.$$

car

$$\forall 1 \leq i \leq N-1 \quad g^n f^{(m_0+\overbrace{i+k+2-N}^{1\leq \quad \leq k+1})l}(U) = g^n f^{(m_0+\overbrace{k+2-N}^{1\leq \quad \leq k+1})l}(U)$$

par hypothèse.

On a donc bien $g^n f^{(m_0+k+2)l}(U) = g^n f^{m_0 l}(U)$. ■

Définition 4 *On dira qu'une suite est k-*presque strictement croissante *si cette suite est croissante et si elle ne stationne jamais plus de k pas consécutifs.*

Lemme 3 *A n fixé, on peut décider si $V \in \{g^n f^m(U) | m \in \mathbb{N}\}$.*

Preuve : On considère la suite des $f^n(U)$.

Ou bien elle devient non positive à partir d'un certain rang N, on arrête et on teste si $V \in \{g^n f^m(U) | m < N\}$.

Ou bien il existe N_1, l tels que $f^{N_1}(U) \leq f^{N_1+l}(U)$, on considère alors

$$\left\{\begin{array}{l} f^{N_1}(U) \leq f^{N_1+l}(U) \leq \ldots \leq f^{N_1+kl}(U) \leq \ldots \\ f^{N_1+1}(U) \leq f^{N_1+1+l}(U) \leq \ldots \leq f^{N_1+1+kl}(U) \leq \ldots \\ \vdots \\ f^{N_1+(l-1)}(U) \leq f^{N_1+(l-1)+l}(U) \leq \ldots \leq f^{N_1+(l-1)+kl}(U) \leq \ldots \end{array}\right.$$

$$\Rightarrow \left\{\begin{array}{l} g^n f^{N_1}(U) \leq g^n f^{N_1+l}(U) \leq \ldots \leq g^n f^{N_1+kl}(U) \leq \ldots \\ g^n f^{N_1+1}(U) \leq g^n f^{N_1+1+l}(U) \leq \ldots \leq g^n f^{N_1+1+kl}(U) \leq \ldots \\ \vdots \\ g^n f^{N_1+(l-1)}(U) \leq g^n f^{N_1+(l-1)+l}(U) \leq \ldots \leq g^n f^{N_1+(l-1)+kl}(U) \leq \ldots \end{array}\right.$$

D'après le lemme 2 (en posant $U' = f^{N_1}(U)$), on sait que chaque suite est :

- ou bien $(k+1)$-presque strictement croissante,
- ou bien stationnaire.

Donc pour chacune de ces l suites, il existe m_i (constructible algorithmiquement) tel que $\forall m \geq m_i$:

- soit $g^n f^{N_1+i+ml}(U) \tilde{>} V$, tous les termes suivants de cette suite seront donc différents de V (car $\tilde{>} V$), il est inutile de les calculer,
- soit $g^n f^{N_1+i+ml}(U) = g^n f^{N_1+i+m_i l}(U)$, il est donc aussi inutile de calculer les termes suivants de la suite, puisqu'îls sont égaux à des termes précédemment calculés.

En effet, on calcule pas à pas la suite $(g^n f^{N_1+i+ml}(U))_m$:

- si on "dépasse" V (i.e., pour au moins l'une des coordonnées) et on arrête,
- si cela fait plus de $k+1$ fois que l'on trouve la même valeur, la suite est stationnaire,
- si on atteint V, on arrête,
- sinon on recommence et on atteint nécessairement l'un des trois cas précédents.

■

1.3.3 Preuve

Pour arriver au résultat annoncé, il faut contrôler $g^n f^m(U)$ quand n et m varient en même temps, pour cela on va procéder en faisant une sorte de quadrillage.

Le lemme qui suit va servir à initialiser la démonstration :

Lemme 4 *Soient $f(X) = AX + B$, $g(X) = CX + D$ avec $A, C \in \mathcal{M}_k(\mathbb{N})$, $B, D \in \mathbb{Z}^k$*

Si $f^{N_1}(U) \leq f^{N_1+l}(U) \leq f^{N_1+2l}(U) \leq \ldots \leq f^{N_1+ml}(U) \leq \ldots$

$$\text{et si} \quad \begin{array}{ll} \exists n_0 & g^{n_0} f^{N_1}(U) \not\geq 0 \\ \exists n_1 & g^{n_1} f^{N_1+l}(U) \not\geq 0 \\ \vdots & \\ \exists n_k & g^{n_0} f^{N_1+kl}(U) \not\geq 0 \end{array}$$

avec n_i le plus petit n tel que $g^n f^{N_1+il}(U) \not\geq 0$,

alors

1. $n_0 \leq n_1 \leq \cdots \leq n_k$, *2.* $\forall m \geq k \quad g^{n_k} f^{N_1+ml}(U) \not\geq 0$

Preuve : La preuve de 1 est simple : $g^{n_0-1} f^{N_1+l}(U) \geq g^{n_0-1} f^{N_1}(U) \geq 0$ donc $n_1 \geq n_0$ et ainsi de suite.

La preuve de 2 utilise une nouvelle fois des résultats élémentaires d'algèbre.

$f^{N_1}(U), \ldots, f^{N_1+kl}(U)$ sont $(k+1)$ vecteurs liés dans $\mathbb{Q}$, donc aussi dans $\mathbb{Z}$. Donc

$$\exists n_0, \ldots, n_k \in \mathbb{Z} \quad \sum_{i=0}^{k} n_i f^{N_1+il}(U) = 0.$$

Soit N le plus grand i tel que n_i est non nul, on suppose $n_N > 0$ quitte à multiplier l'équation par -1.

Soit $I = \{i | n_i > 0\}$, $J = \{j | n_j < 0\}$, pour $j \in J$ on pose $m_j = -n_j > 0$.

$$n_N f^{N_1+Nl}(U) + \sum_{i \in I} n_i f^{N_1+il}(U) = \sum_{j \in J} m_j f^{N_1+jl}(U) \tag{1.1}$$

Comme $f^{N_1}(U) \leq \ldots \leq f^{N_1+kl}(U)$, on a $g^{n_k} f^{N_1}(U) \leq \ldots \leq g^{n_k} f^{N_1+kl}(U)$. De plus, $g^{n_k} f^{N_1+kl}(U) \not\geq 0$ donc il existe une coordonnée i_0 strictement négative pour chacun des $g^{n_k} f^{N_1+il}(U) \quad 1 \leq i \leq k$.

$$(1) \Rightarrow n_N f^{N_1+Nl}(U) \leq \sum_{j \in J} n_j f^{N_1+jl}(U)$$

En appliquant $f^{(k+1-N)l}$ à cette dernière équation, on obtient :

$$n_N f^{N_1+(k+1)l}(U) \leq \sum_{j \in J} n_j f^{N_1+\overbrace{(k+1-N+j)}^{\leq k} l}(U)$$

$$\Rightarrow n_N g^{n_k} f^{N_1+(k+1)l}(U) \leq \sum_{j \in J} n_j \underbrace{g^{n_k} f^{N_1+(k+1-N+j)l}(U)}_{i_0^{\text{ième}} \text{ coordonnée } < 0}$$

donc $g^{n_k} f^{N_1+(k+1)l}(U)$ a la $i_0^{\text{ième}}$ coordonnée négative (strictement) et donc

$$g^{n_k} f^{N_1+(k+1)l}(U) \not\geq 0.$$

Et on montre $\forall m \geq k \quad g^{n_k} f^{N_1+ml} \not\geq 0$ par récurrence.

■

On peut maintenant s'attaquer à la preuve du théorème.

On commence par éliminer le cas où $g^n f^m(U)$ devient non positif pour n assez grand puis on montre que pour n et m assez grands, $g^n f^m(U)$ est stationnaire ou a "dépassé" V. Pour cela, on commence par extraire de $(f^m(U))_m$ une sous-suite croissante et on examine $(g^n(U'))_n$ sur chacun des termes de cette sous-suite, après suffisamment d'itérations (de g ou de f), les suites sont nécessairement stationnaires ou ont dépassé V.

Preuve :

On considère la suite des $f^n(U)$

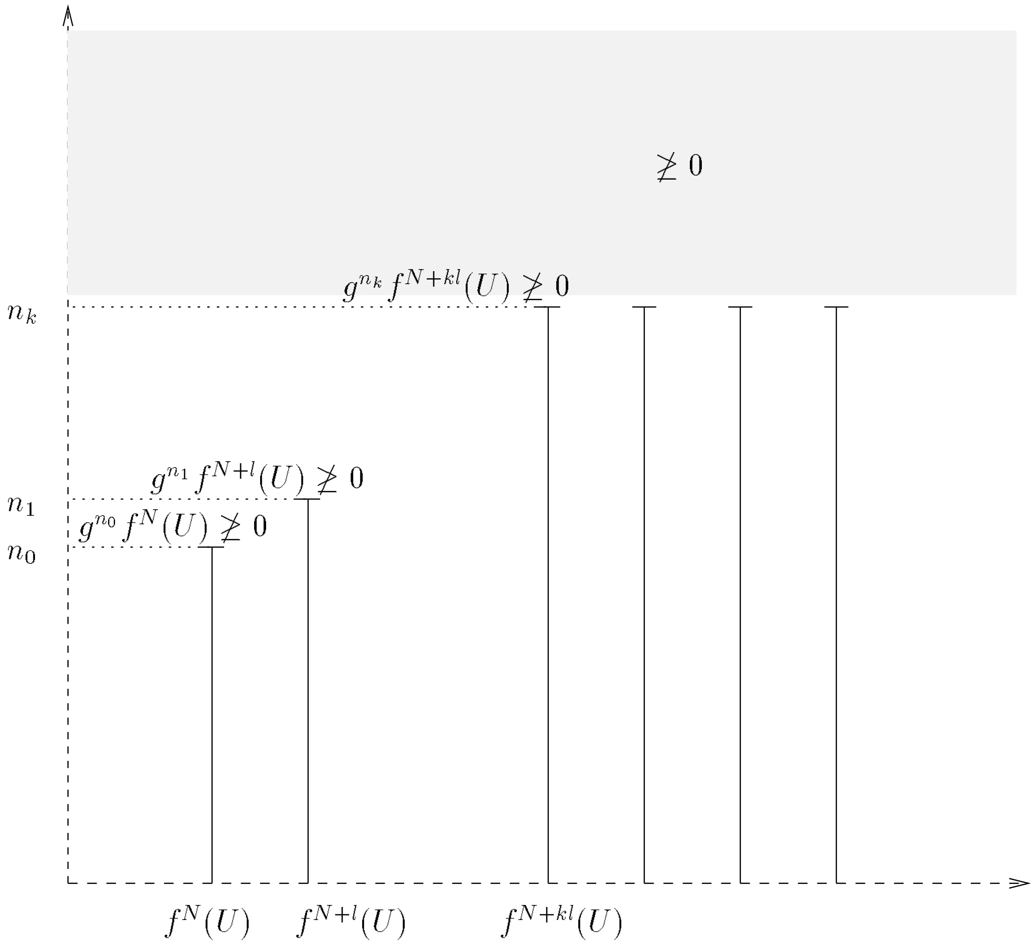


FIG. 1.2 – *Cas 1*

Ou bien elle devient non positive à partir d'un certain rang N, on arrête et pour chaque $m < N$, on teste si $V \in \{g^n f^m(U) | n \in \mathbb{N}\}$.

Ou bien il existe N, l tels que $f^N(U) \leq f^{N+l}(U) \leq \ldots \leq f^{N+kl}(U) \leq \ldots$.

On considère alors la suite $g^n f^N(U))_n$. D'après le lemme 1 :

1. Ou bien il existe n_0 tel que $g^{n_0} f^N(U) \ngeq 0$, on considère alors la suite $(g^n f^{N+l}(U))_n$.
   Si il existe n_1 tel que $g^{n_1} f^{N+l}(U) \ngeq 0$, on considère alors la suite $(g^n f^{N+2l}(U))_n$ et ainsi de suite jusque $(g^n f^{N+kl}(U))_n$.
   Si à nouveau, il existe n_k tel que $g^{n_k} f^{N+kl}(U) \ngeq 0$, alors, d'après le lemme 4, $\forall m \geq k \quad g^{n_k} f^{N+ml} \ngeq 0$. On teste alors pour chaque $n < n_k$, si $V \in \{g^n f^{N+ml}(U) | m \in \mathbb{N}\}$ et l'algorithme est terminé, c'est ce qui correspondant à la figure 1.2.
   Sinon, quitte à renommer $(N := N + il)$, on peut supposer que
   $\forall n \quad g^n f^N(U) \geq 0$ donc

$$\forall n \, \forall m \quad g^n f^{N+ml}(U) \geq g^n f^N(U) \geq 0 \tag{1.2}$$

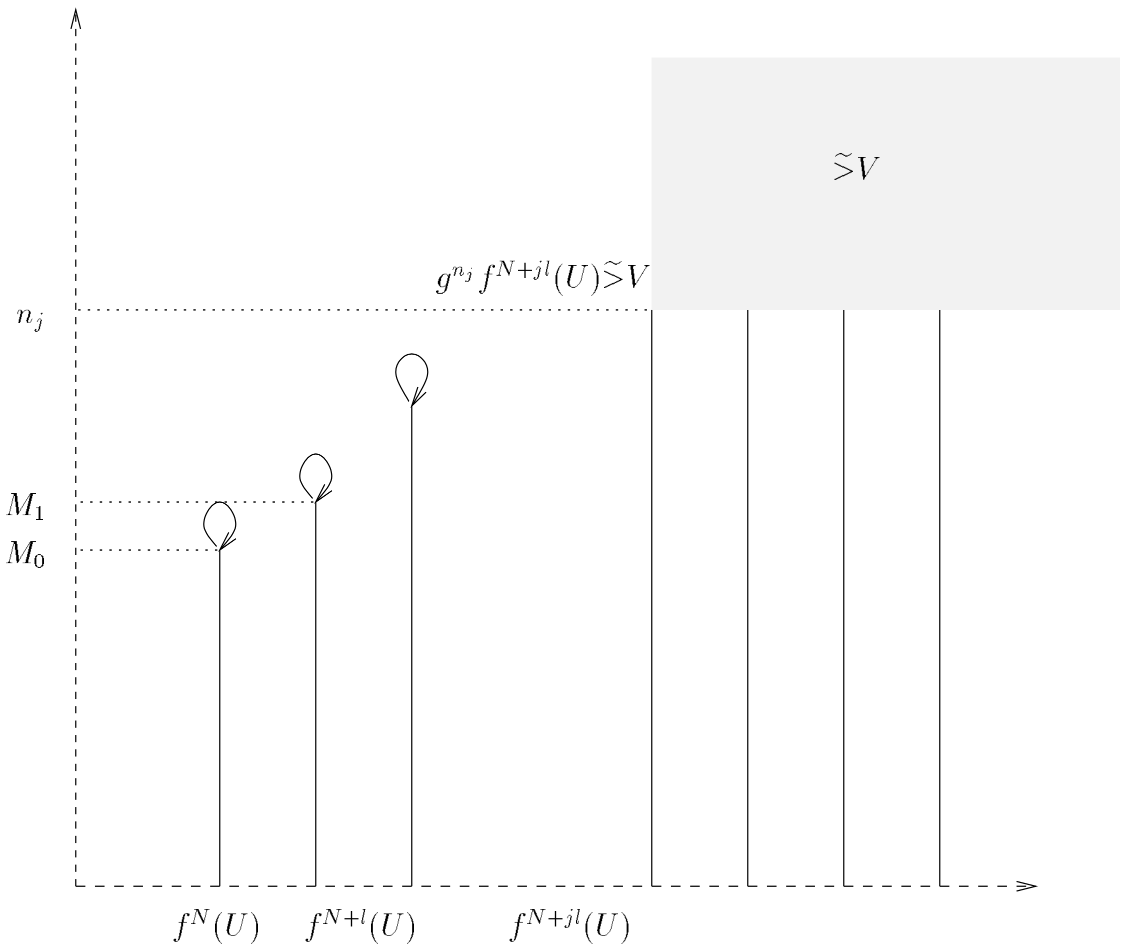


FIG. 1.3 – *Cas 2*

2. Ou bien il existe n_0 tel que $\forall n \geq n_0 \quad g^n f^N(U) \widetilde{>} V$, dans ce cas,
$$\forall n \geq n_0 \, \forall m \geq 0 \quad g^n f^{N+ml}(U) \geq g^n f^N(U) \widetilde{>} V \tag{1.3}$$
voir figure 1.3.
3. Ou bien il existe M_0, l_0 tels que
$$\forall n \geq M_0 + l_0 \quad \exists i < l_0 \quad g^n f^N(U) = g^{M_0+i} f^N(U)$$
Dans ce cas, on considère $(g^n f^{N+l}(U))_n$: toujours d'après le lemme 1,
   - ou bien il existe n_1 tel que $g^{n_1} f^N(U) \not\geq 0$, exclu d'après l'équation 1.2,
   - ou bien n_1 tel que $\forall n \geq n_1 \quad g^n f^{N+l}(U) \widetilde{>} V$, dans ce cas,
$$\forall n \geq n_1 \, \forall m \geq 1 \quad g^n f^{N+ml}(U) \geq g^n f^{N+l}(U) \widetilde{>} V \tag{1.4}$$
   - ou bien il existe M_1, l_1 tels que
$$\forall n \geq M_1 + l_1 \quad \exists i < l_1 \quad g^n f^{N+l}(U) = g^{M_1+i} f^{N+l}(U)$$
   et on considère $(g^n f^{N+2l}(U))_n$.

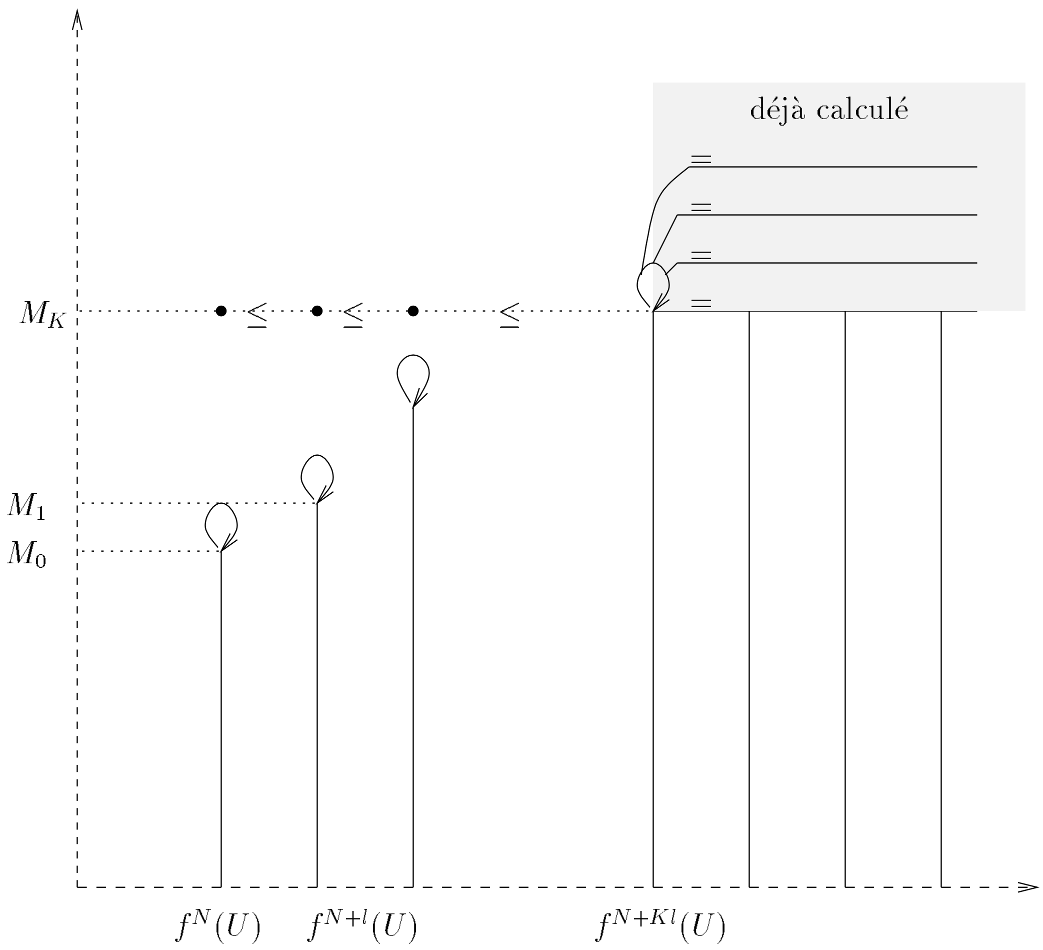


FIG. 1.4 – *Cas 3*

On recommence ainsi au plus $K = (k+1)|V|$ fois et :
ou bien on a trouvé n_j tel que

$$\forall n \geq n_j \, \forall m \geq j \quad g^n f^{N+ml}(U) \geq g^n f^{N+jl}(U) \widetilde{>} V \tag{1.5}$$

et dans ce cas, si $V \in \{g^n f^{N+ml}(U) | n, m \in \mathbb{N}\}$ ssi $V \in \{g^n f^{N+ml}(U) | n \leq n_j, m \in \mathbb{N}\} \cup \{g^n f^{N+ml}(U) | m \leq j, n \in \mathbb{N}\}$ ce qui est décidable.
ou bien on a construit

$$\begin{array}{l}
M_0, l_0 \text{ tels que } \forall n \geq M_0 + l_0 \quad \exists i < l_0 \quad g^n f^N(U) = g^{M_0+i} f^N(U) \\
M_1, l_1 \text{ tels que } \forall n \geq M_1 + l_1 \quad \exists i < l_1 \quad g^n f^{N+l}(U) = g^{M_1+i} f^{N+l}(U) \\
\qquad\qquad\qquad\qquad\qquad\vdots \\
M_K, l_K \text{ tels que } \forall n \geq M_K + l_K \quad \exists i < l_K \quad g^n f^{N+Kl}(U) = g^{M_K+i} f^{N+Kl}(U) \; (*)
\end{array}$$

(voir la figure 1.4.)
On a alors :

$$g^{M_K+i} f^{N+Kl}(U) \geq g^{M_K+i} f^{N+(K-1)l}(U) \geq \ldots \geq g^{M_K+i} f^N(U).$$

D'après le lemme 2, cette suite est :
– soit est déjà stationnaire et

$$\forall m \geq K \quad g^{M_K+i} f^{N+ml}(U) = g^{M_K+i} f^{N+Kl}(U),$$

– soit est $(k+1)$ presque strictement croissante (au moins jusqu'au $K^{\text{ième}}$ terme de la suite) et donc

$$|g^{M_K+i} f^{N+Kl}(U)| \geq \frac{K}{k+1} + |g^{M_K+i} f^N(U)| = |V| + |g^{M_K+i} f^N(U)|$$

$$\Rightarrow g^{M_K+i} f^{N+Kl}(U) \tilde{>} V$$

Dans tous les cas, il existe M, K tels que :

$$\forall n \geq M \, \forall m \geq K \quad g^n f^{N+ml}(U) \tilde{>} V \tag{1.6}$$

ou bien

$$\forall n \geq M \, \forall m \geq K \quad g^n f^{N+ml}(U) = g^n f^{N+Kl(U)} \tag{1.7}$$

En effet, on pose $M = M_K + l_K$; soient $m \geq K$, $n \geq M$, $n = M_K + i + n' l_K$, $i < l_K$ et

$$g^n f^{N+Kl}(U) = g^{M_K+i} f^{N+Kl}(U) \text{ d'après } (*)$$

– soit $\forall m \geq K \quad g^{M_K+i} f^{N+ml}(U) = g^{M_K+i} f^{N+Kl}(U)$ et alors $g^n f^{N+ml}(U) = g^n f^{N+Kl}(U)$ soit 1.7
– soit $g^{M_K+i} f^{N+Kl}(U) \tilde{>} V$ et $g^n f^{N+ml}(U) \geq g^n f^{N+Kl}(U) = g^{M_K+i} f^{N+Kl}(U) \tilde{>} V$ soit 1.6.

On recommence exactement la même chose avec les l suites :

$$f^{N+i}(U) \leq f^{N+i+L}(U) \leq \ldots f^{N+i+ml}(U) \leq \ldots$$

et on obtient le résultat suivant :

Il existe K_i, M_i constructibles algorithmiquement tels que :
- soit $\forall n \geq M_i \, \forall m \geq K_i \quad g^n f^{N+i+ml}(U) \tilde{>} V$
- soit $\forall n \geq M_i \, \forall m \geq K_i \quad g^n f^{N+i+ml}(U) = g^n f^{N+i+K_i l}(U)$

A ce stade là, on a suffisamment localisé où pouvait se trouver V :
– pour tout $n \leq \sup_{0 \leq i \leq l-1} M_i$, on teste si $V \in \{g^n f^m(U) | m \in \mathbb{N}\}$ (décidable d'après le lemme 3),
– pour tout $m \leq \sup_{0 \leq i \leq l-1} (N + i + K_i)$, on teste si $V \in \{g^n f^m(U) | n \in \mathbb{N}\}$ (décidable d'après le corollaire 1)

En effet, si l'algorithme répond oui, c'est qu'il a explicitement trouvé n, m tels que $V = g^n f^m(U)$.

Si l'algorithme répond non, supposons que $V = g^{n_0} f^{m_0}(U)$ et montrons que c'est impossible :

– si $n_0 \leq \sup M_i$, on a testé si $V \in \{g^{n_0} f^m(U) | m \in \mathbb{N}\}$, contradiction,

– si $m_0 \leq \sup(N+i+K_i)$, on a testé si $V \in \{g^n f^{m_0}(U) | n \in \mathbb{N}\}$, contradiction,

– si $n_0 \geq \sup M_i$ et $m_0 \geq \sup(N+i+K_i)$, alors $m_0 = N + i_0 + m'_0 l$ avec $m'_0 \geq N_{i_0}$. Donc, par construction,

  – soit $g^{n_0} f^{m_0}(U) = g^{n_0} f^{N+i_0+m'_0 l}(U) \widetilde{>} V$, contradiction,

  – soit $g^{n_0} f^{m_0}(U) = g^{n_0} f^{N+i_0+m'_0 l}(U) = g^{n_0} f^{N+i_0+K_{i_0} l}(U)$ et on a testé si $V \in \{g^n f^{N+i_0+K_{i_0} l}(U) | n \in \mathbb{N}\}$, contradiction.

Conclusion : l'algorithme est correct et $V \in \{g^n f^m(U) | n, m \in \mathbb{N}\}$ est décidable. ■

1.4 Indécidabilité dans le cas général

Dans le cas général, c'est-à-dire si on itère un nombre arbitraire p (mais fixé) de fonctions dans un ordre donné, savoir si $V \in f_p^* \dots f_1^*(U)$ est indécidable.

Théorème 3 *Etant donnés $f_1, \dots, f_p$, $f_i(X) = A_i X + B_i$ où $A_i \in M_k(\mathbb{N})$, $B_i \in \mathbb{Z}^k$, $U, V \in \mathbb{N}^k$, alors $V \in f_p^* \dots f_1^*(U)$ est indécidable.*

Preuve : On se ramène au dixième problème de Hilbert :

Théorème 4 *Etant donné un polynôme à 5 variables, à coefficients entiers, savoir si ce polynôme admet un zéro dans $\mathbb{N}^5$ est indécidable.*

Preuve : voir [14].

$$\text{Soit } P(x_1, \dots, x_5) = \sum_{n=0}^{N} \left(\sum_{n_1+\dots+n_5=n} a_{n_1,\dots,n_5} x_1^{n_1} \cdots x_5^{n_5} \right)$$

un polynôme de 5 variables à coefficients entiers.

On lui associe le polynôme

$$\tilde{P}(x_1, \dots, x_5) = \begin{cases} \sum_{n=1}^{N} (\sum_{n_1+\dots+n_5=n} a_{n_1,\dots,n_5} x_1^{n_1} \cdots x_5^{n_5}) \text{ si } a_{0,\dots,0} < 0 \\ -\sum_{n=1}^{N} (\sum_{n_1+\dots+n_5=n} a_{n_1,\dots,n_5} x_1^{n_1} \cdots x_5^{n_5}) \text{ si } a_{0,\dots,0} \geq 0 \end{cases}$$

$$P(x_1, \dots, x_5) = 0 \text{ ssi } \tilde{P}(x_1, \dots, x_5) = |a_{0,\dots,0}|$$

L'idée de la preuve est de faire correspondre à chaque polynôme P de cette forme, des fonctions affines $f_1, \dots, f_n$ telles que P admet un zéro entier si et seulement si

$$V_P \in \{f_p^{n_p} \dots f_1^{n_1}(0) | n_1, \dots, n_p \in \mathbb{N}\}$$

, où V_P est un vecteur construit en fonction de P. On aura alors montré le théorème 3.

On reprend la fonction multiplication de l'exemple (1) :

$$f(X) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} X + \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}$$

Lorsque l'on itère f sur $\begin{pmatrix} a \\ b \\ 0 \end{pmatrix}$, on obtient exactement ab sur la $3^{ième}$ coordonnée lorsque la première est nulle. On dit que f multiplie la $1^{ière}$ coordonnée par la $2^{ième}$ et écrit le résultat sur la $3^{ième}$.

Soient i, j, l des entiers distincts, on définit de la même façon $f_{i.j \to l}$, la fonction qui "multiplie la $i^{ière}$ coordonnée par la $j^{ième}$ et écrit le résultat sur la $l^{ième}$", au sens où :

$$\exists n \quad f_{i.j\to l}^{n} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ x_i \\ 0 \\ \vdots \\ 0 \\ x_j \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ x_j \\ 0 \\ \vdots \\ d \end{pmatrix} \text{ ssi } d = x_i x_j$$

A chaque monôme $m = a_{n_1,\ldots,a_5} x_1^{n_1} \cdots x_5^{n_5}$, on associe un bloc $\mathcal{B}_m$ de taille $N_m \times N_m$, $N_m = 2(n_1 + \cdots + n_5) - 1$. On ne considérera par la suite que des matrices par blocs et des vecteurs par blocs de la forme :

$$\left(\begin{array}{cccc|c} \mathcal{B}_{m_1} & 0 & 0 & 0 & 0 \\ 0 & \mathcal{B}_{m_2} & 0 & 0 & \vdots \\ 0 & 0 & \ddots & 0 & \vdots \\ 0 & 0 & 0 & \mathcal{B}_{m_K} & 0 \\ \hline 0 & \cdots & \cdots & 0 & 1 \end{array}\right) \quad , \quad \begin{pmatrix} \mathcal{V}_{m_1} \\ \mathcal{V}_{m_2} \\ \vdots \\ \mathcal{V}_{m_K} \\ a \end{pmatrix}$$

où K est le nombre de monômes.

En fait, on alloue de la place à chacun des monômes. On va en effet procéder de la manière suivante :

- on "crée" $x_1, \ldots, x_5$ entiers positifs en les disposant de manière adéquate pour chaque monôme,
- pour chaque monôme, on calcule $x_1^{n_1} \cdots x_5^{n_5}$ en multipliant x_1 par x_1 par x_1, etc ... n_1 fois, puis en multipliant le tout par x_2 et ainsi de suite.
- on additionne petit à petit les $a_{n_1,\ldots,n_5} x_1^{n_1} \cdots x_5^{n_5}$ en veillant à ce que la somme reste toujours positive (du moins si P admet $x_1, \ldots, x_5$ pour solution) ,
- on annule les coordonnées qui ne sont pas à tester
- on teste si $V \in \{f_p^{n_p} \ldots f_1^{n_1}(U) | n_1, \ldots, n_p \in \mathbb{N}\}$, où U est le vecteur nul et $V = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ |a_{0,\ldots,0}| \end{pmatrix}$, ce qui revient à tester si toutes les opérations ont été faites comme

on le souhaitait et si $P(x_1, \ldots, x_5) = 0$, ce dernier test étant réalisé sur la dernière coordonnée.

1.4.1 Création des x_i

On construit 5 fonctions $C_{x_i} = Id + V_{x_i}$ où V_{x_i} est un vecteur par blocs construits de la manière suivante : à chaque monôme correspond un vecteur bloc $V_m^{x_i}$ de taille N_m qui comporte des 0 sur toutes les coordonnées correspondant aux x_j, $j \neq i$ et des 1 sur toutes les coordonnées correspondant aux x_i puis des 0 sur toutes les coordonnées suivantes.

Exemple : soit $m = x_1^2 x_2 x_3 x_5^3$, $N_m = 2\,(2+1+1+0+3) - 1 = 13$

$$V_m^{x_1} = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ N \end{pmatrix}, V_m^{x_2} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ N \end{pmatrix}, V_m^{x_3} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ N \end{pmatrix}, V_m^{x_4} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ N \end{pmatrix}, V_m^{x_5} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ N \end{pmatrix}$$

où N est le vecteur nul de taille $\frac{N_m - 1}{2} = 6$

Pour calculer m, on aura plus qu'à multiplier la $1^{ière}$ coordonnée par la $2^{ième}$ en écrivant le résultat sur la $8^{ième}$, puis à remultiplier le tout par la $3^{ième}$ en écrivant sur la $9^{ième}$ et ainsi de suite.

En posant $C_{x_{im}} = Id + V_m^{x_i}$, on a que :

$$C_{x_{1m}}^5 C_{x_{2m}}^0 C_{x_{3m}}^3 C_{x_{4m}}^2 C_{x_{5m}}^6 \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ N \end{pmatrix} = \begin{pmatrix} 5 \\ 5 \\ 0 \\ 3 \\ 6 \\ 6 \\ 6 \\ N \end{pmatrix}$$

En calculant $\{C_{x_1}^{n_1} C_{x_2}^{n_2} C_{x_3}^{n_3} C_{x_4}^{n_4} C_{x_5}^{n_5} | n_1, \ldots, n_5 \in \mathbb{N}\}$, on génère ainsi tous les $n_1, \ldots, n_5$ entiers positifs sur lesquels on teste le polynôme P.

1.4.2 Fonctions de multiplication

Pour chaque monôme $m = a_{n_1,\ldots,a_5} x_1^{n_1} \cdots x_5^{n_5}$, on construit $F_{m,1}, \ldots, F_{m,N-1}$ où $N = n_1 + \cdots + n_5$, $F_{m,i}(X) = A_{m,i}X + B_{m,i}$.

$A_{m,i}$: tous les blocs sont l'identité sauf le bloc correspondant à $\mathcal{B}_m$. $B_{m,i}$: tous les vecteurs blocs sont nuls sauf le bloc correspondant à $\mathcal{V}_m$.

On reprend les fonctions multiplication $f_{i,j\to l}$ définies plus haut, $f_{i,j\to l}(X) = A_{i,j\to l}X + B_{i,j\to l}$.

Le bloc de $A_{m,1}$ correspondant à $\mathcal{B}_m$ est $A_{1,2\to N+1}$, le bloc de $B_{m,1}$ correspondant à $\mathcal{V}_m$ est $B_{1,2\to N+1}$, Le bloc de $A_{m,2}$ correspondant à $\mathcal{B}_m$ est $A_{3,N+1\to N+2}$, le bloc de $B_{m,2}$ correspondant à $\mathcal{V}_m$ est $B_{3,N+1\to N+2}$.

$\vdots$

Le bloc de $A_{m,N-1}$ correspondant à $\mathcal{B}_m$ est $A_{N,2N-2\to 2N-1}$, le bloc de $B_{m,N-1}$ correspondant à $\mathcal{V}_m$ est $B_{N,2N-2\to 2N-1}$.

En fait, $F_{m,1}$ fait le travail de $f_{1,2\to N+1}$ sur le bloc de coordonnées correspondant à $\mathcal{B}_m$ et en laissant intact toutes les autres coordonnées, $F_{m,2}$ fait le travail de $f_{3,N\to N+2}$ sur le bloc de coordonnées correspondant à $\mathcal{B}_m$ et en laissant intact toutes les autres coordonnées, et ainsi de suite.

De plus, on construit $I_m = \{i_m, i_m+3, i_m+4, \dots, i_m+N-2, i_m+N-1, i_m+2N-1\}$, où i_m est la coordonnée de la première ligne du bloc $\mathcal{B}_m$. Cet ensemble correspond aux coordonnées qu'il faudra tester : après avoir itérer $F_{m,1}$ puis $F_{m,2}$, puis, ..., puis $F_{m,N-1}$, on obtient le produit des N premières coordonnées écrit sur la dernière coordonnée si et seulement si les coordonnées dont le numéro est dans $I_m - \{i_m + 2N - 1\}$ sont nulles.

1.4.3 Calcul du polynôme

On répartit les monômes du polynôme $\tilde{P}$ en 2 catégories :

$$I = \{m | a_{n_1,\dots,n_5} \geq 0\}, \quad J = \{m | a_{n_1,\dots,n_5} < 0\}$$

Pour tout m dans I, on définit

$$A_m^+ = Id + \begin{pmatrix} 0 \\ \vdots \\ 0 \\ -1 \\ 0 \\ \vdots \\ 0 \\ a_{n_1,\dots,n_5} \end{pmatrix},$$

(-1) étant sur la coordonnée $i_m + 2N - 1$. Pour tout m dans J, on définit

$$A_m^- = Id + \begin{pmatrix} 0 \\ \vdots \\ 0 \\ -1 \\ 0 \\ \vdots \\ 0 \\ a_{n_1,\dots,n_5} \end{pmatrix},$$

(-1) étant sur la coordonnée $i_m + 2N - 1$, c'est-à-dire la coordonnée où est écrit le résultat de la multiplication de $x_1^{n_1} \cdots x_5^{n_5}$.

On commencera par itérer tous les A_m^+ (ce qui revient à additionner les monômes positifs), puis on itèrera tous les A_m^-. Ainsi, on s'assure que si le polynôme P admet un 0 en la valeur testée, i.e, si $\tilde{P}$ prend la valeur $|a_{n_1,\dots,n_5}|$, on reste positif tout le long du chemin.

1.4.4 Tests

On construit une matrice diagonale D, telle que $D_{i,i} = 1$ si $i \in \bigcup_m I_m$, $D_{i,i} = 0$ sinon. En fait, D annule toutes les coordonnées qui ne sont pas à tester.

On a alors que

$$\exists x_1 \dots \exists x_5 \quad P(x_1, \dots, x_5) = 0$$

ssi

$$\begin{gathered}
\exists n_D \exists n_{A^-_{m_1}} \dots \exists n_{A^-_{m_M}} \exists n_{A^+_{m_1}} \dots \exists n_{A^+_{m_{M'}}} \\
\exists n_{m_1,1} \dots \exists n_{m_1,N_{m_1}} \dots \exists n_{m_K,1} \dots \exists n_{m_K,N_{m_K}} \exists n_1 \dots \exists n_5 \\
D^{n_D} {A^-_{m_1}}^{n_{A^-_{m_1}}} \dots {A^-_{m_M}}^{n_{A^-_{m_M}}} {A^+_{m_1}}^{n_{A^+_{m_1}}} \dots {A^+_{m_{M'}}}^{n_{A^+_{m_{M'}}}} \\
{F_{m_1,1}}^{n_{m_1,1}} \dots {F_{m_1,N_{m_1}}}^{n_{m_1,N_{m_1}}} \dots {F_{m_K,1}}^{n_{m_K,1}} \dots {F_{m_K,N_{m_K}}}^{n_{m_K,N_{m_K}}} \\
C^{n_1}_{x_1} \dots C^{n_5}_{x_5} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ |a_{n_1,\dots,n_5}| \end{pmatrix}
\end{gathered}$$

où K est le nombre total de monômes, M le nombre de monômes positifs, M' le nombre de monômes négatifs.

ssi

$$\begin{pmatrix} 0 \\ \vdots \\ 0 \\ |a_{n_1,\dots,n_5}| \end{pmatrix} \in \left\{ D^{n_D} \cdots C^{n_5}_{x_5} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix} \middle| n_D, \dots, n_5 \in \mathbb{N} \right\}$$

Si $V \in \{f_p^{n_p} \dots f_1^{n_1}(U) | n_1, \dots, n_p \in \mathbb{N}\}$ était décidable, alors le dixième problème de Hilbert le serait aussi en décidant ce qui précède.

Conclusion : $V \in \{f_p^{n_p} \dots f_1^{n_1}(U) | n_1, \dots, n_p \in \mathbb{N}\}$ est indécidable.

∎

1.4.5 Existence d'une borne

On vient de montrer que, étant donnés p, $f_1, \dots, f_p$, $f_i(X) = A_i X + B_i$ où $A_i \in M_k(\mathbb{N})$, $B_i \in \mathbb{Z}^k$, $U, V \in \mathbb{N}^k$, alors $V \in \{f_p^{n_p} \dots f_1^{n_1}(U) | n_1, \dots, n_p \in \mathbb{N}\}$ est indécidable. En fait, on peut affiner le théorème précédent : p n'est pas un paramètre du problème, on peut fixer à l'avance le nombre de fonctions f_i.

Théorème 5 *Il existe $f_1, \dots, f_p$, tels que étant donné $U, V \in \mathbb{N}^k$,
$V \in \{f_p^{n_p} \dots f_1^{n_1}(U) | n_1, \dots, n_p \in \mathbb{N}\}$ est indécidable. Ce résultat a pour conséquence que :*

*il existe une borne $K \in \mathbb{N}$, telle que pour $p \geq K$ fixé ("à l'avance"), étant donnés $f_1, \dots, f_p$, $f_i(X) = A_i X + B_i$ où $A_i \in M_k(\mathbb{N})$, $B_i \in \mathbb{Z}^k$, $U, V \in \mathbb{N}^k$, alors
$V \in \{f_p^{n_p} \dots f_1^{n_1}(U) | n_1, \dots, n_p \in \mathbb{N}\}$ est indécidable.*

Pour démontrer ce dernier résultat, on s'appuie sur le théorème 3.10 de [14] :

Théorème 6 *Il existe un polynôme $P(x, y_1, \dots, y_m)$ à coefficients entiers tel qu'il n'existe aucun algorithme qui décide si étant donné x_0 entier positif,
$P(x_0, y_1, \dots, y_m) = 0$ admet une solution entière positive en $y_1, \dots, y_m$.*

Preuve : La preuve de ce théorème est faite dans [14] en se ramenant au problème de l'arrêt d'une machine de Turing qui simule toutes les autres.

■

On associe à ce polynôme P, comme dans la démonstration précédente, des fonctions C_x, C_{y_i}, $F_{m_i,j}$, $A^+_{m_i}$, $A^-_{m_i}$, D et on a :

Etant donné x_0, $P(x_0, y_1, \ldots, y_m) = 0$ admet une solution entière positive en $y_1, \ldots, y_m$ si et seulement si

étant donné $U = C_x^{x_0} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}$, $V = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ |a_{n_1,\ldots,n_5}| \end{pmatrix}$

$$V \in \left\{ D^{n_D} \cdots C_{y_1}^{n_1} \cdots C_{y_m}^{n_m} U \middle| \, n_D, \ldots, n_1, \ldots, n_m \in \mathbb{N} \right\}$$

ce qui est donc indécidable, d'où le résultat.

1.5 Quelques cas de décidabilité

1.5.1 Fonctions "croissantes pour la taille"

Théorème 7 *Soient p fonctions affines f_i, "croissantes pour la taille" :*

$$\forall X \quad |f_i(X)| \geq X$$

Alors le problème $V \in f_p^* \ldots f_1^*(U)$ est décidable.

Remarque : cette hypothèse est vérifiée si (par exemple) la somme des coordonnées de chacun des B_i est positive et si aucun des A_i n'a de colonne nulle.

Preuve : Etant donné un vecteur U de $\mathbb{N}^k$, une fonction f qui vérifie $\forall X \quad |f(X)| \geq X$, on considère la suite $(|f^n(U)|)_n$.

Lemme 5 *Pour toute fonction affine $f : \mathbb{N}^k \to \mathbb{N}^k$, si la suite $(|f^n(U)|)_n$ stationne plus de k pas, elle est stationnaire.*

Preuve : On procède exactement de la même manière que pour la démonstration du lemme 2. ■

La suite des $(|f^n(U)|)_n$ est une suite croissante qui est :

- soit k-presque strictement croissante et donc il existe N tel que $\forall n \geq N \quad |f^n(U)| \geq |V|$,
- soit stationnaire et alors il existe N tel que $\forall n \geq N \quad \exists i \quad f^n(U) = f^{N-i}(U)$. En effet, si $(|f^n(U)|)_n$ est stationnaire alors $(f^n(U))_n$ ne peut prendre qu'un nombre fini de valeurs

On pose $C_f(U) = \{f^n(U) | n \leq N\} = \{f^n(U) | \, |f^n(U)| \leq |V|\}$. Et pour tout ensemble fini $X = \{X_1, \ldots, X_n\}$, on pose

$$C_f(X) = \bigcup_{i=1}^{n} C_f(X_i).$$

$C_f(X)$ est fini.

On construit alors $C_{f_p}(C_{f_{p-1}}(\cdots(C_{f_1}(U))\cdots)$, ensemble fini.

$$C_{f_p}(C_{f_{p-1}}(\cdots(C_{f_1}(U))\cdots) = \{f_p^{n_p}\dots f_1^{n_1}(U)|\,|f_p^{n_p}\dots f_1^{n_1}(U)| \leq |V|\}.$$

Donc $V \in \{f_p^{n_p}\dots f_1^{n_1}(U)|n_1,\dots,n_p \in \mathbb{N}\}$ ssi $V \in C_{f_p}(C_{f_{p-1}}(\cdots(C_{f_1}(U))\cdots)$ ce qui est décidable. ∎

1.5.2 Coefficients positifs

Théorème 8 *Soient $f_1 = A_1X + B_1, \dots, f_p = A_pX + B_p$ telles que chacun des B_i est un vecteur d'entiers positifs alors étant donnés $U, V \in \mathbb{N}^k$,*
$V \in \{f_p^{n_p}\dots f_1^{n_1}(U)|n_1,\dots,n_p \in \mathbb{N}\}$ est décidable.

Preuve : Soit $v = \max_{1\leq i\leq k} v_i$ où les v_i sont les coordonnées du vecteur V.

l'idée de la démonstration est que, dès que l'une des coordonnées dépasse v, il est inutile de la connaître exactement, savoir qu'elle est plus grande que v suffit : soit elle est multipliée par un coefficient nul et il était inutile de connaître sa valeur exacte, soit elle sera mutipliée par un nombre non nul et le résultat restera plus grand que v car les tous les B_i sont positifs.

On ajoute un nouveau symbole à $\mathbb{N}$: $\oplus \notin \mathbb{N}$ qui “symbolisera” le fait d'être plus grand que v.

Pour $X \in (\mathbb{N}\cup\{\oplus\})^k$, on définit $\overline{X} = \begin{pmatrix} \overline{X_1} \\ \vdots \\ \overline{X_k} \end{pmatrix}$ par : $\begin{array}{ll} \overline{X_i} = X_i & \text{si } X_i \leq v \\ \overline{X_i} = \oplus & \text{sinon .} \end{array}$

On définit l'opération $\overline{+}$ sur $\mathbb{N}\cup\{\oplus\}$: $\begin{array}{ll} \oplus\overline{+}n = n\overline{+}\oplus = \oplus & \\ n\overline{+}m = \overline{n+m} & \text{avec } n, m \in \mathbb{N}. \\ \oplus\overline{+}\oplus = \oplus & \end{array}$

On définit l'opération $\overline{*}$ sur $\mathbb{N}\cup\{\oplus\}$: $\begin{array}{l} \oplus\overline{*}n = n\overline{*}\oplus = \begin{array}{ll} \oplus & \text{si } n \neq 0 \\ 0 & \text{si } n = 0 \end{array} \\ n\overline{*}m = \overline{n*m} \end{array}$

On définit $\overline{f}$ sur $(\mathbb{N}\cup\{\oplus\})^k$:

$$\overline{f}(X)_i = a_{i,1}\overline{*}x_1\overline{+}\cdots\overline{+}a_{i,k}\overline{*}x_k\overline{+}b_i$$

Proposition 4 *Soit f une fonction affine, $\overline{f}(\overline{X}) = \overline{f(X)}$.*

Preuve : $f(X)_i = \sum_{j=1}^k a_{i,j}x_j + b_i = \sum_{j\in J} a_{i,j}x_j + b_i$ où $J = \{j|a_{i,j} \neq 0\}$, donc $\overline{f(X)_i} = \oplus$ ssi $\sum_{j\in J} a_{i,j}x_j + b_i > v$.

D'autre part, $\overline{f}(\overline{X})_i = a_{i,1}\overline{*}\overline{x_1}\overline{+}\cdots\overline{+}a_{i,k}\overline{*}\overline{x_k}\overline{+}b_i$,

ou bien il existe $j \in J$ tel que $\overline{x_i} = \oplus$ alors $\overline{f}(\overline{X})_i = \oplus$ et $x_j > v$ donc $f(X)_i > v \Rightarrow \overline{f(X)_i} = \oplus$

ou bien pour tout $j \in J \quad x_j \leq v$ donc

$$\overline{f}(\overline{X})_i = \overline{\sum_{j=1}^k a_{i,j}x_j + b_i} = \oplus \text{ ssi } \overline{f(X)_i} = \oplus$$

∎

Pour tout vecteur X, l'ensemble $\{\overline{f^n(X)}|n \in \mathbb{N}\} \subseteq (\{0, \dots, v\} \cup \{\oplus\})^k$ est un ensemble fini, calculable algorithmiquement : on s'arrête dès que l'on a calculé un $\overline{f^n(X)}$ qui est déjà dans l'ensemble car si $\overline{f^{n_1}(X)} = \overline{f^{n_0}(X)}$ alors $f(\overline{f^{n_1}(X)}) = f(\overline{f^{n_0}(X)})$, i.e., $\overline{f^{n_1+1}(X)} = \overline{f^{n_0+1}(X)}$ d'après la propriété qui précède et donc par récurrence,

$$\forall p \quad \overline{f^{n_1+p}(X)} = \overline{f^{n_0+p}(X)}.$$

On définit $\tilde{f}(S)$ où S est un ensemble fini de vecteurs par

$$\tilde{f}(S) = \bigcup_{X \in S} \{\overline{f^n(X)}|n \in \mathbb{N}\}.$$

On construit récursivement $\tilde{f}_p(\tilde{f}_{p-1}(\cdots(\tilde{f}_1(U))\cdots)$ et l'on a

$$\boxed{V \in \{f_p^{n_p} \dots f_1^{n_1}(U)|n_1, \dots, n_p \in \mathbb{N}\} \text{ ssi } V \in \tilde{f}_p(\tilde{f}_{p-1}(\cdots(\tilde{f}_1(U))\cdots).}$$

En effet, si $V \in \{f_p^{n_p} \dots f_1^{n_1}(U)|n_1, \dots, n_p \in \mathbb{N}\}$ alors $V = f_p^{n_p} \dots f_1^{n_1}(U)$ donc $V = \overline{V} = \overline{f_p^{n_p} \dots f_1^{n_1}(U)} = \overline{f_p^{n_p}} \dots \overline{f_1^{n_1}}(\overline{U})$, d'où $V \in \tilde{f}_p(\tilde{f}_{p-1}(\cdots(\tilde{f}_1(U))\cdots)$.

Réciproquement, si $V \in \tilde{f}_p(\tilde{f}_{p-1}(\cdots(\tilde{f}_1(U))\cdots)$, alors

$$\exists n_p \, \exists U_{p-1} \in \tilde{f}_{p-1}(\cdots(\tilde{f}_1(U))\cdots) \quad V = \overline{f_p^{n_p}}(U_{p-1})$$

et, par récurrence, $\exists n_p, \dots, n_1 \quad V = \overline{f_p^{n_p}} \dots \overline{f_1^{n_1}}(\overline{U})$ donc

$$V = \underbrace{\overline{f_p^{n_p} \cdots f_1^{n_1}(U)}}_{\text{ne contient pas de } \oplus}$$

et donc $V = f_p^{n_p} \cdots f_1^{n_1}(U)$.

Conclusion : $V \in \{f_p^{n_p} \dots f_1^{n_1}(U)|n_1, \dots, n_p \in \mathbb{N}\}$ est décidable.

■

1.6 Une borne de la complexité de la décidabilité du théorème 1

Théorème 9 *Soient $f(X) = AX + B$, $A \in M_k(\mathbb{N})$, $B \in \mathbb{Z}^k$, $U, V \in \mathbb{N}^k$. Alors $V \in \{f^n(U)|n \in \mathbb{N}\}$ est décidable en une tour d'exponentielles de $|U| + |V| + |B| + |A|$, dont la hauteur est une fonction linéaire de k.*

La preuve de ce théorème se fait en plusieurs étapes.

Algorithme

Cet algorithme n'est en fait pas le même que celui présenté dans la démonstration du théorème 1. L'intérêt de ce premier algorithme était d'être très simple à présenter mais nous n'en avons pas trouvé la complexité.

Le deuxième algorithme étudie plus finement le comportement de la fonction, malheureusement la borne de complexité de cet algorithme est "super-exponentielle".

On associe à la matrice $A \in M_k(\mathbb{N})$, le graphe (orienté) des dépendances de la manière suivante :

- à chaque coordonnée correspond un noeud du graphe étiqueté par le numéro de cette coordonnée,
- si $a_{i,j} \neq 0$, alors le graphe comporte un arc allant de j vers i étiqueté par $a_{i,j}$.

Le nombre de "jetons" qu'il y a dans chaque noeud représente la valeur de la coordonnée correspondant au noeud.

Chaque "coup" correspond à une itération de la fonction et consiste à déplacer tous les jetons suivants les flèches, en les multipliant par le coefficient étiquetant l'arc.

Par exemple, si

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix},$$

son graphe des dépendances est ci-dessus.

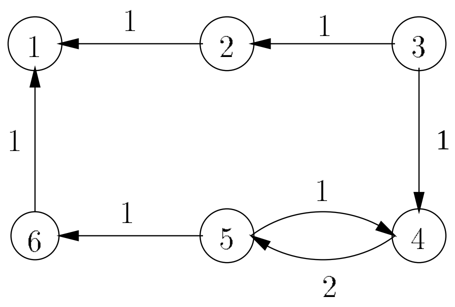


FIG. 1.5 – *Graphe des dépendances de A*

$$\text{et le vecteur } \begin{array}{c} 0 \\ 1 \\ 0 \\ 5 \\ 2 \\ 4 \end{array} \text{ est représenté par } \begin{array}{clc} 0 & \text{jeton dans le noeud numéro} & 1 \\ 1 & \text{“} & 2 \\ 0 & \text{“} & 3 \\ 5 & \text{jetons dans le noeud numéro} & 4 \\ 2 & \text{“} & 5 \\ 4 & \text{“} & 6 \end{array}.$$

On décompose ce graphe en composantes fortement connexes, pour l'exemple de la matrice A, il y a une unique composante fortement connexe, voir figure 1.6.

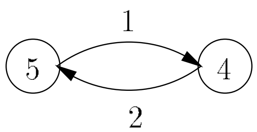


FIG. 1.6 – *Composante(s) fortement connexe(s) du graphe des dépendances de A*

Définition 5 *On dira qu'il y a* création de jetons *dans une composante fortement connexe si il existe un état de cette composante fortement connexe duquel partent au moins deux flèches* *étiquetées par $n \geq 1$ ou au moins une flèche étiquetée par ≥ 2 qui vont dans des états de la composante fortement connexe. (voir figure 1.7)*

FIG. 1.7 – *Cas où il y a création de jetons*

Lemme 6 *Soit G un sous-graphe fortement connexe avec création de jetons.*

Soit

$$b^+ = \sum_{b_i > 0} b_i, \quad b^- = \sum_{b_i < 0} -b_i, \quad b = b^- + |V|.$$

Soient $q_1, \ldots, q_k$ les états du sous-graphe, où q_1 est un état d'où part une création de jetons et tel que q_2 joint q_1 en au plus 1 coup, q_3 joint q_1 en au plus 2 coups, ... On note Σ_n la somme des jetons dans le sous-graphe au bout de n coups.

Si $\Sigma_n > (k+1)b$ alors pour tout $m \geq n$, $\Sigma_m > b$

Preuve : On note $\sharp q_i$ le nombre de jetons dans q_i à l'étape n.

$$\begin{array}{l}
\Sigma_n > (k+1)b \\
\Sigma_{n+1} \geq \Sigma_n + \sharp q_1 - b \quad (\geq kb) \\
\Sigma_{n+2} \geq \Sigma_n + \sharp q_1 + \sharp q_2 - 2b \quad (\geq (k-1)b) \\
\vdots \\
\Sigma_{n+k} \geq \Sigma_n + \sharp q_1 + \sharp q_2 + \cdots + \sharp q_n - kb \geq 2\Sigma_n - kb \quad (> (k+1)b)
\end{array}$$

et on recommence, d'où le résultat. Cela signifie que dès que $\Sigma_n > (k+1)b$, il est impossible d'atteindre V par la suite. ∎

On ordonne les composantes fortement connexes (même celles éventuellement réduites à un point) par l'ordre induit par le graphe : si G_1, G_2 sont des composantes fortement connexes et si $\exists q_1 \in G_1 \, \exists q_2 \in G_2 \quad q_1 \longrightarrow q_2$ alors $G_1 > G_2$.

Remarque : Une composante fortement connexe, non réduite à un point, qui n'a pas de création de jetons est en réalité un cycle dont toutes les arêtes sont étiquetées par des 1. (voir figure 1.8)

L'algorithme pour décider étant donné A, B, U, V, k (où k est la dimension de l'espace) si $V \in \{f^n(U) | n \in \mathbb{N}\}$ est le suivant :

On associe à A son graphe des dépendances G_A puis on décompose ce graphe en composantes fortement connexes. On choisit une composante fortement connexe maximale pour l'ordre induit (cela signifie qu'aucune autre composante ne l'alimente). Trois cas se présentent alors.

<u>Cas 1</u> : Cette composante fortement connexe est réduite à un point qui correspond à la coordonnée i.

- Ou bien $U = V$, l'algorithme est terminé. Sinon, $\forall n \geq 1 \quad f^n(U)_i = b_i$ et :
- Ou bien $U \neq V$ et $v_i \neq b_i$ alors $\forall n \quad f^n(U) \neq V$, l'algorithme est terminé.

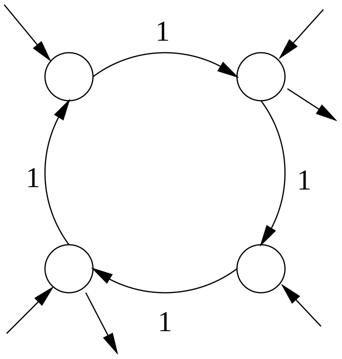


FIG. 1.8 – *Exemple de cycle*

- Ou bien $U \neq V$ et $v_i = b_i$, cette composante correspond alors à celle de V et alimente donc les autres composantes de manière constante, on peut donc recommencer l'étude en supprimant cette composante et en tenant seulement compte des constantes qu'elle apporte à chaque itération. On recommence donc l'algorithme avec A', la matrice extraite de A en enlevant la $i^{ème}$ ligne et la $i^{ème}$ colonne, B' est le vecteur extrait de B en enlevant la $i^{ème}$ coordonnée et en ajoutant le vecteur B'' où $B''_j = a_{j,i}b_i$ qui correspond à l'apport constant de la $i^{ème}$ composante, U' le vecteur extrait de $f(U)$ en enlevant la $i^{ème}$ coordonnée et $V' = V$, $k' = k - 1$.

Cas 2 : Cette composante fortement connexe n'a pas de création de jetons : c'est un cycle, de longueur k_1, soit J l'ensemble des coordonnées correspondant au cycle. On considère une itération : tous les jetons se sont déplacés d'un cran dans le cycle et ont perdu (ou gagné) b_i jetons. La somme des jetons dans le cycle a gagné (ou perdu) $\sum_{i \in J} b_i$ jetons :

- Ou bien $\sum_{i \in J} b_i > 0$, alors la somme des jetons croît strictement toutes les itérations, au bout de au plus $|V| + 1$ itérations, la somme des jetons dans le cyle est strictement supérieure à $|V|$ donc soit on a atteint V avant, soit on ne l'atteindra jamais. L'algorithme s'arrête en au plus $|V| + 1$ itérations.
- Ou bien $\sum_{i \in J} b_i < 0$, alors la somme des jetons décroît strictement toutes les itérations, au bout de au plus $|U|+1$ itérations, $f^n(U)$ est devenue non positive : $\{f^n(U)|n \in \mathbb{N}\} = \{f^n(U)|n \leq |U| + 1\}$ est fini, l'algorithme s'arrête en au plus $|U| + 1$ itérations.
- $\sum_{i \in J} b_i = 0$, le nombre de jetons dans chaque composante du cycle est alors périodique, de période n divisant k_1, la somme des jetons dans le cycle est constante égale à Σ.

  Ou bien $\forall n \leq k_1 \, \exists i \in J \quad f^n(U)_i \neq v_i$ (aucune des configuration possibles du cycle ne correspond aux coordonnées de V) et alors $\forall n \quad f^n(U) \neq V$, l'algorithme est terminé.

  Ou bien $\exists k_0 \leq k_1 \, \forall i \in J \quad f^n(U)_i = v_i$, on s'intéresse alors uniquement au comportement du graphe toutes les k_1 itérations à partir de k_0 puisque ce sont les seules susceptibles d'atteindre V.

  On recommence donc l'algorithme avec $A' = A''^{k_1}$ où A'' est la matrice extraite de A en enlevant les colonnes et les lignes correspondant aux indices de J, B' le vecteur extrait de $A^{k_1-1}B + \cdots + AB + B$ en enlevant les coordonnées de J auquel on ajoute un vecteur correspondant à l'apport induit par les jetons du cycle pendant toute la durée d'un cycle (cet apport est le même toutes les fois que l'on effectue un cycle), U' le vecteur extrait de $f^{k_0}(U)$ en enlevant les coordonnées de J, V' le vecteur extrait de V en enlevant les coordonnées de J, $k' = k - k_1$.

<u>Cas 3</u> : Cette composante fortement connexe est créatrice de jetons. Soit k_1 le nombre d'états de cette composante et J l'ensemble des coordonnées correspondant aux états de cette composante, d'après le lemme 6 si la somme des jetons de ce sous-graphe dépasse $(k+1)b$ où $b = |B| + |V|$ alors la somme des jetons dans le sous graphe restera toujours plus grande que $|V|$, on n'atteindra jamais V. On considère alors $U, f(U), \ldots, f^{((k+1)b)^{k_1}}$.

- Ou bien $f^n(U)$ est devenue non positive et l'algorithme est terminé.
- Ou bien il existe n_0, $i \in J$ tels que $f^{n_0}(U)$ dépasse $(k+1)b$ alors la somme des jetons dans le sous graphe créateur de jetons dépasse $(k+1)b$, on ne peut plus atteindre V, l'algorithme est terminé.
- Ou bien il existe n_1, n_2 tels que $f^{n_1}(U)$ et $f^{n_2}(U)$ soient égaux sur les coordonnées de J, alors le comportement du sous graphe créateur de jetons est périodique, de période $T \leq ((k+1)b)^{k_1}$. Ou bien aucune des configuration du sous-graphe ne correspond aux coordonnées de V, dans ce cas, on n'atteindra jamais V, l'algorithme est terminé. Ou bien, l'une des configurations $f^{n_0}(U)$ correspond aux coordonnées de V, on recommence l'algorithme avec $A' = A''^T$ où A'' est la matrice extraite de A en enlevant les colonnes et les lignes correspondant aux indices de J, B' le vecteur extrait de $A^{T-1}B + \cdots + AB + B$ en enlevant les coordonnées de J auquel on ajoute un vecteur correspondant à l'apport induit par les jetons du sous-graphe pendant toute la durée d'une période, U' le vecteur extrait de $f^{n_0}(U)$ en enlevant les coordonnées de J, V' le vecteur extrait de V en enlevant les coordonnées de J , $k' = k - k_1$.

L'algorithme termine toujours puisque la taille du graphe (i.e. k) diminue strictement à chaque appel de l'algorithme.

Analyse de la complexité de l'algorithme

Si l'algorithme s'attaque à un noeud et ne termine pas à cette étape là, il rappelle l'algorithme avec

$$|A'| \leq |A|,$$

$$|B'| \leq |B| + |A||B|,$$

$$|U'| \leq |f(U)| \leq |A||U| + |B|.$$

Si l'algorithme s'attaque à un cycle de longueur k_1 et ne termine pas à cette étape là, il rappelle l'algorithme avec

$$|A'| \leq |A|^{k_1},$$

$$|B'| \leq |B| + |A||B| + \cdots |A|^{k_1-1}|B| + |A||U| + \cdots + |A|^{k_1}|U| \leq |A|^{k_1}|B| + |A|^{k_1+1}|U|,$$

$$|U'| \leq |A|^{k_0}|U| + |A|^{k_0-1}|B| + \cdots + |B|$$

avec $k_0 \leq k_1$ donc

$$|U'| \leq |A|^{k_1}|U| + |A|^{k_1-1}|B| + \cdots + |B| \leq |A|^{k_1+1}|U|$$

et $|V'| = |V|$.

Si l'algorithme s'attaque à un sous-graphe créateur de jetons de cardinal k_1 et ne termine pas à cette étape là, il rappelle l'algorithme avec

$$|A'| \leq |A|^T$$

où $T \leq ((k+1)b)^{k_1}$ et $b = |B| + |V|$,

$$|B'| \leq |B| + |A||B| + \cdots |A|^{T-1}|B| + |A||U| + \cdots + |A|^T|U| \leq |A|^T|B| + |A|^{T+1}|U|,$$

$$|U'| \leq |A|^{n_0}|U| + |A|^{n_0-1}|B| + \cdots + |B|$$

avec $n_0 \leq T$ donc

$$|U'| \leq |A|^T|U| + |A|^{T-1}|B| + \cdots + |B| \leq |A|^{T+1}|U|$$

et $|V'| = |V|$.

Le cas pire pour la complexité est donc lorsqu'il n'a que des sous graphes créateurs de jetons, une borne de la complexité est alors une tour d'exponentielles de hauteur le nombre de sous graphes créateurs de jetons. La borne de complexité ainsi obtenue est tellement mauvaise que je ne l'ai pas calculée de manière plus précise. Par contre, dans le cas où le nombre de places dans le graphe est 2, 3 ou 4 (i.e. si $k = 2$, 3 ou 4), alors la complexité reste polynômiale en la taille de A, B, U, V.

Cas où $k = 2$, 3 ou 4

A chaque fois, nous allons calculer $N(|A|, |B|, |U|, |V|, k)$ une borne du nombre maximal d'itérations nécessaires pour décider si $V \in \{f^n(U) | n \in \mathbb{N}\}$.

Lemme 7 *Si G est un sous-graphe créateur de jetons à 2 places, minimal pour l'ordre induit et J l'ensemble des coordonnées correspondant aux 2 places de G, alors on peut décider en au plus $2(|V| + |U| + 1)$ coups si $V \in \{f^n(U) | n \in \mathbb{N}\}$ ou en appelant l'algorithme précédement exposé avec $A' = A''^2$ où A'' est la matrice extraite de A en enlevant les colonnes et les lignes correspondant aux indices de J, B' le vecteur extrait de $AB + B$ en enlevant les coordonnées de J auquel on ajoute un vecteur correspondant à l'apport induit par les jetons de G pendant deux itérations, U' le vecteur extrait de $f(U)$ ou U en enlevant les coordonnées de J, $V' = V$, $k' = k - 2$.*

Preuve : G est un sous-graphe créateur de jetons à 2 places, minimal pour l'ordre induit donc G est de la forme : voir figure 1.9 avec $a \geq 1$, $b \geq 1$, $aa \geq 2$.

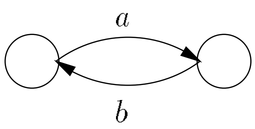


FIG. 1.9 – *Sous-graphe créateur de jetons à 2 places*

On note u_n et v_n le nombre de jetons dans chacune des deux places à la $n^{ième}$ itération de f, $\Sigma_n = u_n + v_n$.

$$\begin{array}{ll} u_{n+1} = av_n + b_1 & u_{n+2} = abu_n + ab_2 + b_1 \\ v_{n+1} = bu_n + b_2 & v_{n+2} = abv_n + bb_1 + b_2 \end{array}$$

donc $\Sigma_{n+2} = ab\Sigma_n + ab_2 + b_1 + bb_1 + b_2$, avec $ab > 0$.

– Ou bien $(\Sigma_{2n})_n$ est constante, dans ce cas u_{2n} et v_{2n} ne peuvent prendre qu'un nombre fini de valeurs. Or si $u_0 < u_2$ ou si $u_0 > u_2$, alors $(u_{2n})_n$ est strictement croissante ou strictement décroissante donc u_{2n} prend un nombre infini de valeurs. Donc $u_0 = u_2$ et $(u_{2n})_n$ est constante, de même pour $(v_{2n})_n$.

Si $(u_{2n})_n$ et $(v_{2n})_n$ sont constantes, alors, il en est de même pour $(u_{2n+1})_n$ et $(v_{2n+1})_n$ donc : ou bien aucune des deux configurations u_0, v_0 et u_1, v_1 ne correspondent à V, alors $V \notin \{f^n(U) | n \in \mathbb{N}\}$, $N = 2$, ou bien l'une des deux configurations correspond à V et on appelle l'algorithme avec $A' = A''^2$ où A'' est la matrice extraite de A en enlevant les colonnes et les lignes correspondant aux indices de J, B' le vecteur extrait de $AB+B$ en enlevant les coordonnées de J auquel on ajoute un vecteur correspondant à l'apport induit par les jetons de G pendant deux itérations, U' le vecteur extrait de $f(U)$ (si u_1, v_1 correspond à V) ou U (si u_0, v_0 correspond à V) en enlevant les coordonnées de J, $V' = V$, $k' = k - 2$.

- Ou bien $(\Sigma_{2n})_n$ est strictement monotone (si $\Sigma_0 < \Sigma_2$ ou si $\Sigma_0 > \Sigma_2$) alors $(\Sigma_{2n+1})_n$ est elle aussi strictement monotone (d'après ce qui précède, si l'une des deux suites est constantes alors l'autre l'est aussi) donc en au plus $2(|U|+1)$, $f^n(U)$ est négative ou en au plus $2(|V|+1)$, $f^n(U)$ "dépasse" V.

$\blacksquare$

Remarque : La complexité engendrée par un sous-graphe créateur de jetons à deux places est la même que celle engendrée par un cycle à une multiplication par 2 près.

<u>Si $k = 2$</u>

- Si le graphe a deux composantes fortement connexes (réduites chacune à un point) ce que l'on note $k = 1 + 1$, alors $\forall n \geq 2 \quad f^n(U) = f^2(U)$, $N = 2$.
- Si le graphe a une composante fortement connexe (un cycle ou un créateur de jetons), $N = 2(|V| + |U| + 1)$.

$$N(|A|, |B|, |U|, |V|, 2) \leq 2(|V| + |U| + 1)$$

<u>Si $k = 3$</u>

- Si $k = 1 + 1 + 1$, $N_{1,1,1}(|A|, |B|, |U|, |V|, 3) = 3$.
- Si $k = 1 + 2$ (i.e. si le graphe se décompose en un noeud - minimal pour l'ordre induit- et une composante fortement connexe à deux places) alors

$$N_{1,2}(|A|, |B|, |U|, |V|, 3) =$$

$$1 + N(|A'|, |B'|, |U'|, |V'|, 2) \leq 1 + N(|A|, |A|(|B| + 1), |A||U| + |B|, |V|, 2)$$
$$\leq 1 + 2(|V| + |A||U| + |B| + 1)$$

- Si $k = 2 + 1$,

$$N_{2,1}(|A|, |B|, |U|, |V|, 3) =$$

$$N(|A|, |B|, |U|, |V|, 2) + N(|A'|, |B'|, |U'|, |V'|, 1) \leq 2(|V| + |U| + 1) + 1$$

- Si $k = 3$, i.e. si le graphe se décompose en une unique composante fortement connexe (qui est donc soit un cycle, soit un sous-graphe créateur de jetons) à trois places, alors (voir l'analyse de la complexité plus haut)

$$N_3(|A|, |B|, |U|, |V|, 3) = 4(|B| + |V|)^3 + |U| + 1$$

Dans tous les cas, $N(|A|,|B|,|U|,|V|,3)$ est polynômial.

<u>Si $k = 4$</u>

- Si $k = 1+1+1+1$, $N_{1,1,1,1}(|A|,|B|,|U|,|V|,4) = 4$.
- Si $k = 3+1$,

$$\begin{aligned} N_{3,1}(|A|,|B|,|U|,|V|,4) &= N_3(|A|,|B|,|U|,|V|,3) + N(|A|,|B|,|U|,|V|,1) \\ &= 4(|B|+|V|)^3 + |U| + 1 + 1 \end{aligned}$$

- Si $k = 1+3$,

$$\begin{aligned} N_{1,3}(|A|,|B|,|U|,|V|,4) &= N(|A|,|B|,|U|,|V|,1) + N_3(|A'|,|B'|,|U'|,|V'|,3) \\ &\leq 1 + N_3(|A|,|A|(|B|+1),|A||U|+|B|,|V|,3) \\ &= 4(|A|(|B|+1)+|V|)^3 + |A||U| + |B| + 1 \end{aligned}$$

- Si $k = 2+2$,

$$\begin{aligned} N_{2,2}(|A|,|B|,|U|,|V|,4) &= N(|A|,|B|,|U|,|V|,2) + N(|A'|,|B'|,|U'|,|V'|,2) \\ &\leq 2(|V|+|U|+1) + N(|A|^2,|A|^2|B|+|A|^3|U|,|A|^3|U|,|V|,2) \\ &= 2(|V|+|U|+1) + 2(|V|+|A|^3|U|+1) \end{aligned}$$

- Si $k = 4$,

$$N_4(|A|,|B|,|U|,|V|,4) = 5(|B|+|V|)^4 + |U| + 1$$

Dans tous les cas, $N(|A|,|B|,|U|,|V|,4)$ est polynômial.

Nous allons résumer tous ces résultats dans la proposition suivante :

Proposition 5 *Soit $f(X) = AX+B$, $f : \mathbb{N}^k \to \mathbb{N}^k$, et soient $U, V \in \mathbb{N}^k$. Une borne N du nombre maximal d'itérations de f nécessaires pour décider si $V \in \{f^n(U) | n \in \mathbb{N}\}$ vaut :*

- *si $k = 2$, $N = \mathcal{O}(|U|+|V|)$*
- *si $k = 3$, $N = \mathcal{O}((|B|+|V|)^3 + |A||U|)$*
- *si $k = 4$, $N = \mathcal{O}(|A|(|B|+|V|)^3 + |A|^3|U|)$*

Chapitre 2

Logique et automates plats

La "platitude" d'un automate est une propriété de non imbrication de boucles. Une telle propriété du contrôle permet des résultats de décidabilité de systèmes à nombre infini d'états. Or il est intéressant de noter que cette notion de platitude revient souvent en informatique.

- Ainsi, pour étudier l'ensemble des configurations accessibles dans un automate à file, A. Finkel et O. Marcé utilisent dans [11] des expressions régulières de la forme $x_0 y_0^* \cdots x_k y_k^* x_{k+1}$ (LRE) qui se trouvent justement être des expressions plates. Elles permettent d'exprimer de manière exacte l'ensemble des états accessibles après l'itération de cycles.
- De même, A. Bouajjani et P. Habermehl utilisent des CQDD : expressions régulières de la même forme, avec une contrainte exprimée sous la forme d'une formule de Presburger, pour décrire de manière exacte l'ensemble des configurations accessibles d'un automate à plusieurs files après l'itération de cycles (voir [5]).
- P. Abdulla, A. Bouajjani et B. Jonsson utilisent dans [1] des SRE qui sont des automates presque plats (les seules boucles sont des boucles d'un état sur lui-même mais plusieurs lettres de l'alphabet peuvent alors être utilisées alternativement) pour représenter l'ensemble des mots accessibles dans un automate à file avec perte.
- H. Comon et Y. Jurski ont exhibé une classe d'automates à compteurs telle que l'accessibilité soit expressible dans l'arithmétique de Presburger. Cette classe d'automate possède la propriété particulière d'être plat : il n'y a pas de boucles imbriquées. Si on enlève cette condition, on obtient alors une classe d'automates pour laquelle on peut simuler une machine de Türing.

De plus, dans la caractérisation par automates des modèles d'une formule de logique temporelle, on trouve une notion semblable pour les automates alternants, cf [19].

Ceci amène à se poser plusieurs types de questions :

- Quelles sont les propriétés de ces automates plats? (par exemple, propriété de clôture?)
- A-t-on d'autres caractérisations?
- Le fait que les automates associés (qui ont les mêmes modèles) aux formules de logique temporelle soient alternant plats, est-il une coïncidence?

Ces questions conduisent à s'intéresser en premier lieu à une caractérisation en logique temporelle des automates plats (à compteurs) répondant (au moins partiellement) aux trois questions précédentes.

Nous allons donner une telle caractérisation logique des automates plats (partie 2.2.1). Ceci permettra entre autre de dériver un algorithme de vérification de modèles pour une logique $\mathcal{L}_p^+$ avec compteurs, étendant strictement les résultats sur la vérification de modèles avec LTL.

2.1 Préliminaires

2.1.1 Automates plats à compteurs

Définition 6 *On appelle* automate à k compteurs *un tuple* $(Q, q_i, F, G : Q \times Q \to \mathcal{P}(\mathbb{N}^k \times \mathbb{N}^k) \times \mathcal{P}(\mathcal{V}))$ *où* Q *un ensemble fini d'états de contrôle,* q_i *l'état initial de l'automate,* $F \subseteq Q$ *les états finaux de l'automate,* $\mathcal{V}$ *un ensemble de variables propositionnelles. Un état de l'automate est donné par un état de contrôle de* Q *et un vecteur d'entiers de* $\mathbb{N}^k$ *et un ensemble de variables de* $\mathcal{V}$*,*

On passe de l'état (V, A, q) *à* (V', A', q') *si* (V, V', A) *vérifie la* garde $G_{q,q'}$*.*

Un mot $w = a_1 \cdots a_n$ *(fini ou infini) est* accepté *par l'automate si chaque lettre du mot est un vecteur de* $\mathbb{N}^k$ *et un sous ensemble de* $\mathcal{V}$ *:* $a_i = (V_i, A_i)$ *et si il existe un chemin* $q_1 \cdots q_n$ *dans l'automate tel que* $q_1 = q_i$*,* $q_n \in F$ *et* $\forall i \quad G_{q_i,q_{i+1}}(V_i, V'_{i+1}, A_i)$*.*

On confondra souvent la garde $G_{q,q'}$ avec une formule qui admet $G_{q,q'}$ comme ensemble de solutions.

De plus, par convention, lorsque l'on décrira une garde $G_{q,q'}$, les variables primées désigneront les coordonnées du vecteur après la transition, les variables non primées désigneront les coordonnées du vecteur avant la transition.

Remarque : si on impose que toutes les gardes soient des conjonctions d'expressions de la forme $x' = x + c$ où c est une constante de $\mathbb{Z}$ et que l'on considère un automate à compteurs sans variables propositionnelles, alors on retrouve exactement les réseaux de Pétri.

Exemple : L'automate de la figure 2.1 est un automate à deux compteurs qui simule (de manière simplifiée) le comportement d'un téléphone à pièces :

- le compteur x compte le nombre de pièces encaissées, tandis que le compteur y compte le nombre de crédits temps écoulé.
- il y a 7 variables propositionnelles : lift!, quarter!, dial!, connected ?, signal ?, busy ?, hang!. lift! est vraie si l'utilisateur a décroché, quarter! est vraie si l'utilisateur ajoute une pièce, et ainsi de suite.

Pour alléger les notations, il est sous entendu que lorsqu'un changement de l'un des compteurs n'est pas spécifié, c'est que le compteur reste constant : on a omis $x' = x$, $y' = y$ partout où l'on indiquait pas de changement sur x ou y.

Exemple de mot accepté par l'automate :

$$\begin{array}{ccccccc} \begin{pmatrix}0\\0\end{pmatrix} & \begin{pmatrix}0\\0\end{pmatrix} & \begin{pmatrix}1\\0\end{pmatrix} & \begin{pmatrix}1\\0\end{pmatrix} & \begin{pmatrix}2\\0\end{pmatrix} & \begin{pmatrix}3\\0\end{pmatrix} & \begin{pmatrix}4\\0\end{pmatrix} \\ \text{lift!} & \text{quarter!} & \text{dial!} & \text{quarter!} & \text{quarter!} & \text{quarter!} & \text{connected?} \end{array}$$

$$\begin{array}{ccccccc} \begin{pmatrix}4\\0\end{pmatrix} & \begin{pmatrix}4\\1\end{pmatrix} & \begin{pmatrix}4\\2\end{pmatrix} & \begin{pmatrix}4\\2\end{pmatrix} & \begin{pmatrix}2\\0\end{pmatrix} & \begin{pmatrix}1\\0\end{pmatrix} & \begin{pmatrix}0\\0\end{pmatrix} \\ \text{signal?} & \text{signal?} & & \text{hang!} & \text{quarter?} & \text{quarter?} & \end{array}$$

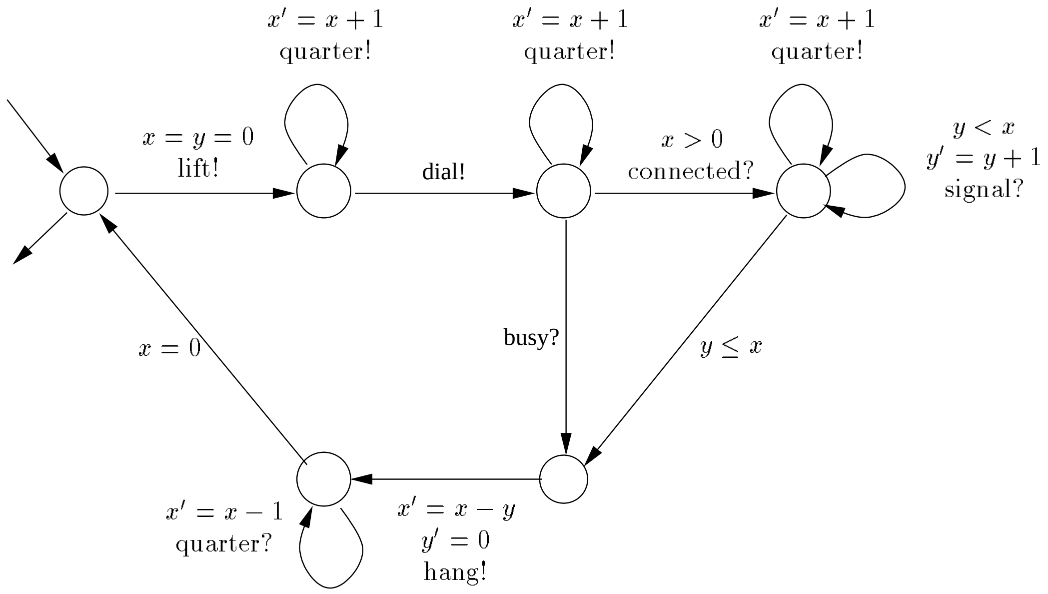


FIG. 2.1 – *Exemple d'automate à compteurs : téléphone à pièces*

L'utilisateur décroche, met une pièce, compose le numéro, ajoute trois pièces, utilise deux crédits temps, il raccroche et l'appareil lui rend deux pièces.

Définition 7 *On dira qu'un automate à compteurs $\mathcal{A}$ est* plat *si :*

- *les seules boucles dans l'automate sont des boucles d'un état sur lui-même,*
- *Il y a au plus une seule boucle sur chaque état.*

Plus formellement $\mathcal{A}$ est plat si il existe un ordre strict $>$ tel que si $G_{q,q'} \neq \emptyset$, ce que l'on notera $q \longrightarrow q'$, et si $q \neq q'$ alors $q > q'$ et s'il a au plus une seule boucle sur chaque état.

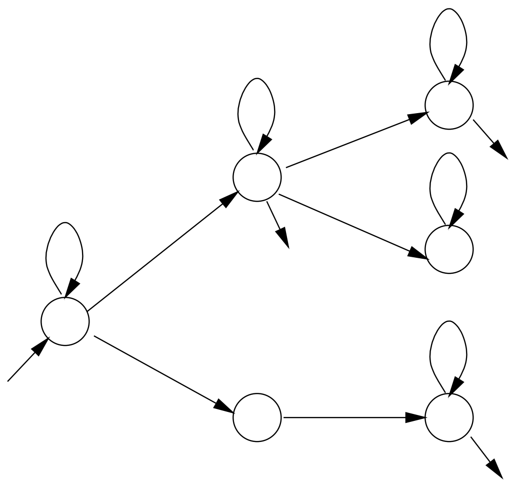


FIG. 2.2 – *Exemple d'automate plat à compteurs*

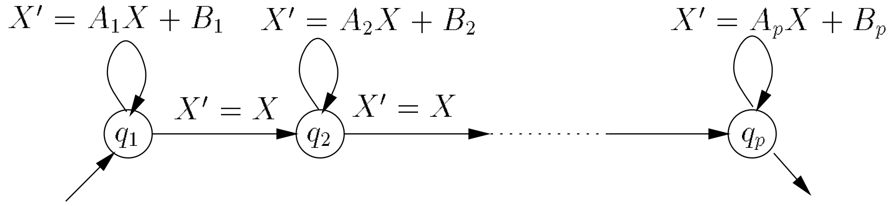


FIG. 2.3 – $V \in \{f_p^{n_p} \cdots f_1^{n_1} | n_1, \ldots, n_p \in \mathbb{N}\}$?

2.1.2 Résultats

On s'intéresse à des automates à compteurs avec des gardes qui sont (ou plutôt qui sont représentables par) des conjonctions de formules atomiques de la forme $x\sharp y' + c$ ou $x\sharp y + c$ ou $x'\sharp y' + c$ ou $x\sharp c$ ou $x'\sharp c$ où $c \in \mathbb{Z}$ et $\sharp \in \{\leq, \geq, =, >, <\}$, sans variables propositionnelles. Il s'agit d'une classe restreinte de l'arithmétique de Presburger que l'on notera $\mathcal{P}_r$.

H. Comon et Y. Jurski ont démontré le résultat suivant dans [6] :

Théorème 10 *Soit $\mathcal{A}$ un automate plat à compteurs dont les gardes sont dans $\mathcal{P}_r$, alors la relation d'accessibilité $\mathcal{R}$ dans l'automate $\mathcal{A}$ s'exprime dans l'arithmétique de Presburger. $\mathcal{R}(V, V')$ signifie qu'il existe un mot accepté dans l'automate commençant par V et finissant par V'.*

Un raffinement est même possible : soit la relation $\mathcal{R}'(V, V, n')$ ssi on peut passer de (V, q_0) à (V', q_f) en empruntant un chemin de longueur n dans l'automate où q_0 est un état initial et q_f un état final, alors $\mathcal{R}'$ s'exprime elle aussi dans l'arithmétique de Presburger.

Un autre résultat concerne les chemins infinis, résultat de Y. Jurski, article non publié :

Théorème 11 *Soit G une garde de $\mathcal{P}_r$, le prédicat $\phi(V)$: "G est franchissable infiniment de fois à partir du vecteur V" s'exprime dans l'arithmétique de Presburger.*

Remarque : On considère les automates à compteurs avec des gardes dans $\mathcal{P}_r$, si on n'impose plus que les automates soient plats, ces automates sont alors plus généraux que les machines de Minski : les seules transitions possibles dans une machine de Minski sont l'incrémentation, la décrémentation et le test à zéro de l'un des compteurs. Or, étant donné n, l'arrêt d'une machine de Minsky à deux compteurs sur $\begin{pmatrix} n \\ 0 \end{pmatrix}$ est indécidable (voir [15]), la relation d'accessibilité dans les automates à compteurs non plats est donc, a fortiori, indécidable. (voir 2.3.4)

Inversement, si on considère des automates plats à compteurs mais avec des gardes un peu différentes : de la forme $X' = AX + B$, alors la relation d'accessibilité n'est plus décidable non plus.

En effet, le problème $V \in \{f_p^{n_p} \cdots f_1^{n_1}(U) | n_1, \ldots, n_p \in \mathbb{N}\}$ revient à poser la question : peut-on passer de la configuration (U, q_1) à la configuration (V, q_p) dans l'automate plat à compteurs suivant : (voir figure 2.3), question qui est donc indécidable.

On voit donc ici que la restriction sur les gardes ET la notion de platitude sont nécessaires pour obtenir une classe décidable.

2.1.3 Logique à compteurs (temps linéaire)

Syntaxe

Les définitions qui suivent sont les définitions classiques de X et $\mathcal{U}$ (voir [?],[19]).

Définition 8 *On notera X, "next", un connecteur logique temporel unaire qui sera interprété de la manière suivante : on dira que le mot infini w vérifie $X\phi$ si et seulement si le sous mot w' obtenu à partir de w en enlevant la première lettre, vérifie ϕ. Intuitivement, $X\phi$ singnifie que ϕ est vraie à la prochaine étape.*

On notera $\mathcal{U}$, "until", un connecteur logique temporel binaire qui sera interprété de la manière suivante : on dira que le mot infini w vérifie $\psi\mathcal{U}\phi$ si et seulement s'il existe un suffixe de w qui vérifie ϕ et tel que tous les suffixes intermédiaires de w vérifie ψ. Intuitivement, $\psi\mathcal{U}\phi$ signifie que ψ est vraie jusqu'à ce que ϕ soit vraie.

Définition 9 *Soit $\mathcal{V}$ un ensemble de variables propositionnelles. On définit la logique $LTL+\mathcal{P}res$ la plus petite logique contenant $\mathcal{V}$, toutes les formules de Presburger, close par l'application de $\neg, \vee, \wedge, X, \mathcal{U}$:*

- *$\phi \in \mathcal{P}res$, $p \in \mathcal{V}$ sont des formules de$LTL+\mathcal{P}res$ $LTL+\mathcal{P}res$,*
- *Si ϕ_1 et ϕ_2 sont des formules de $LTL+\mathcal{P}res$, alors $\neg\phi_1$, $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$, $X\phi_1$, $\phi_1\mathcal{U}\phi_2$ sont des formules de $LTL+\mathcal{P}res$.*

Cette logique $LTL+\mathcal{P}res$ constitue le "contexte" dans lequel on se place : on va chercher une sous-classe décidable de cette logique en utilisant les automates plats à compteurs.

Sémantique

si $w \in (2^{\mathcal{V}} \times \mathbb{N}^k)^\omega$ on associe $\widetilde{w} \in (2^{\mathcal{V}} \times \mathbb{N}^k \times \mathbb{N}^k)^\omega$:

$w(i) = (a, v)$ et $w(i+1) = (b, w)$ entraîne $\widetilde{w}(i) = (a, v, w)$

$w \models \phi$ ssi (par définition) $\widetilde{w}, 0 \models \phi$.

- $\widetilde{w}, i \models$ **true** et $\widetilde{w}, i \not\models$ **false**
- $\widetilde{w}, i \models P$ ssi $\widetilde{w}(i) = (a, v, w)$ et $P \in a$
- $\widetilde{w}, i \models \phi$ (avec $\phi \in \Phi$) ssi $\widetilde{w} = (a, v, w)$ et $v, w \models \phi$
- $\widetilde{w}, i \models \phi_1 \wedge \phi_2$ (resp. $\phi_1 \vee \phi_2$, resp. $\neg\phi_1$) ssi $\widetilde{w}, i \models \phi_1$ et $\widetilde{w}, i \models \phi_2$ (resp. $\models \phi_1$ ou $\models \phi_2$, resp. $\not\models \phi_1$)
- $\widetilde{w}, i \models \mathrm{X}\phi$ ssi $\widetilde{w}, i+1 \models \phi$
- $\widetilde{w}, i \models \phi\,\mathcal{U}\,\psi$ ssi $\exists j \geq i, \forall k \in]i..j], \widetilde{w}, j \models \psi$ et $\widetilde{w}, k \models \phi$.

Remarque: ici et dans la suite on pourrait coder les variables propositionnelles par des entiers, mais les conditions de décidabilité vont être différentes pour les variables propositionnelles et les entiers.

Exemples

Cette logique permet d'exprimer de nombreuses propriétés que l'on voudrait vérifier sur des systèmes.

Ainsi, on peut exprimer le fait qu'une variable du système (x) est toujours plus petite qu'une valeur fixée (100 par exemple), c'est ce qu'exprime la formule suivante :

$$\neg(true\,\mathcal{U}\neg(x \leq 100)).$$

En effet, $true\,\mathcal{U}\neg(x \leq 100)$ est vraie dès que x dépasse 100 donc $\neg(true\,\mathcal{U}\neg(x \leq 100))$ est vraie si x ne dépasse jamais 100.

On peut aussi vérifier que si un paramètre dépasse une certaine valeur, alors on arrive dans un état d'alarme, ce qui est exprimé par la formule :

$$[\neg(true\,\mathcal{U}\neg(x \leq 100))] \vee [(x \leq 100)\,\mathcal{U} q_{alarme}].$$

Modèles

Définition 10 *On dira qu'un automate à compteurs représente une formule de $LTL + \mathcal{P}res$ s'il admet le même langage de mots infinis, i.e., si les mots acceptés par l'automate sont exactement les modèles de la formule.*

Les automates à compteurs utilisés pour modéliser les formules de $LTL + \mathcal{P}res$ sont des automates à compteurs "étendus" par rapport à ceux définis en 6 : les mots considérés sont cette fois des mots de $(2^{\mathcal{V}} \times \mathbb{N}^k)^\omega$, où $\mathcal{V}$ est un ensemble de variables propositionnelles et les gardes sont des conjonctions de formules de $\mathcal{P}_r$ et de variables propositionnelles (ou de leur négation)

Pour établir la correspondance entre automates à compteurs et formules de $LTL + \mathcal{P}res$, on s'est fortement inspiré de l'article de M.Vardi ([19]).

2.2 Logique Plate

2.2.1 Caractérisation des automates plats

Les résultats de H. Comon et Y. Jurski sur les automates plats à compteurs laissent à penser qu'il est possible d'exhiber une logique temporelle avec des formules de Presburger telle que toute formule ait pour modèle un automate plat à compteurs, en utilisant le lien entre logique et automates décrit par M. Vardi dans [19]. On pourrait alors appliquer les résultats précédemment énoncés sur cette logique, en particulier, une telle logique serait décidable.

Définition 11 *Soit $\mathcal{V}$ un ensemble de variables propositionnelles. On définit la logique $\mathcal{L}_p$ (ensemble des* formules plates*) suivante :*

- *Briques :*
  - *$\phi \in \mathcal{P}_r$, $p, \neg p$ où $p \in \mathcal{V}$, sont des briques,*
  - *si b_1 et b_2 sont des briques alors $b_1 \wedge b_2$ est une brique.*
- *Formules élémentaires :*
  - *Si b est une brique, b est une formule élémentaire,*

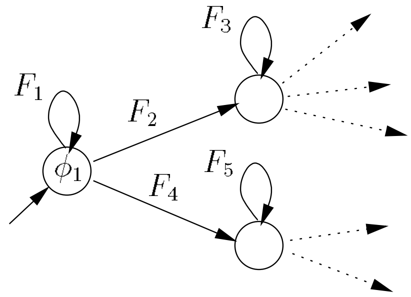


FIG. 2.4 – *Automate de* ϕ_1

  - *Si* ϕ_1 *et* ϕ_2 *sont des formules élémentaires alors* $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$, $\neg\phi_1$ *sont des formules élémentaires.*
- *Formules plates : Toutes les formules élémentaires sont plates et si* ϕ_1 *et* ϕ_2 *sont plates alors :*
  - $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$, $\neg\phi_1$ *sont des formules plates,*
  - $X\phi_1$ *est une formule plate,*
  - $b\mathcal{U}\phi_1$ *est plate où* b *est une brique,*
  - $\neg(true\mathcal{U}\neg b)$ *est plate, où* b *est une brique.*

Attention : Seules les formules élémentaires sont closes par négation, la négation d'une formule plate n'est, elle, pas forcément plate.

De plus, cette logique construite avec les mêmes opérateurs que LTL ne contient pas tout LTL.

La dernière condition : "$\neg(true\mathcal{U}b)$ est plate" peut paraître surprenante, elle correspond au codage des états finaux de l'automate plat à compteurs.

Nous allons montrer que cette logique caractérise exactement les automates plats à compteurs, c'est-à-dire que toute formule de cette logique peut être représentée par un automate plat à compteurs (i.e., qu'il existe un automate plat à compteurs qui reconnaisse exactement les mêmes modèles que ceux acceptés par la formule) et que, réciproquement, tout automate plat à compteur est le représentant d'une formule plate.

Définition 12 *On dit qu'une formule de* $LTL{+}\mathcal{P}res$ *est* modèle-plate *s'il existe un automate plat à compteurs qui reconnait les mêmes modèles.*

Notre but est donc de montrer que les formules de $\mathcal{L}_p$ sont exactement les formules modèle-plates.

Lemme 8 *Si* ϕ_1 *et* ϕ_2 *sont des formules modèle-plates alors* $\phi_1 \vee \phi_2$ *est modèle-plate.*

Preuve :

ϕ_1 a pour représentant un automate plat à compteurs $\mathcal{A}_1$ (voir figure 2.4), i.e., $\mathcal{A}_1$ admet les mêmes modèles que ϕ_1,

ϕ_2 a pour représentant un automate plat à compteurs $\mathcal{A}_2$ (voir figure 2.5),

on construit $\mathcal{A}$ comme on construit l'union de deux automates de Büchi : on ajoute un état $q_{\phi_1 \vee \phi_2}$, seul état initial de l'automate et pour toute garde $G_{q_i,q'}$ de l'automate $\mathcal{A}_1$ où q_i

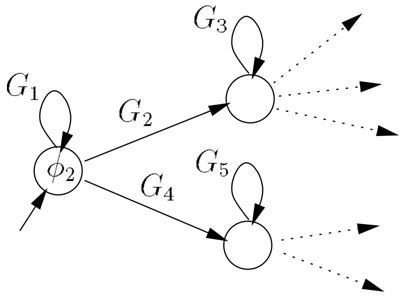


FIG. 2.5 – *Automate de ϕ_2*

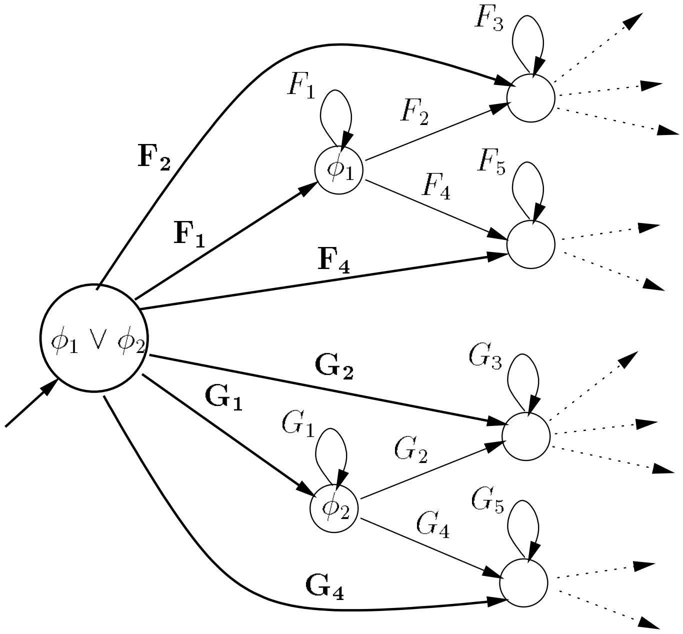


FIG. 2.6 – *Automate de $\phi_1 \vee \phi_2$*

est l'état initial de $\mathcal{A}_1$, on ajoute la garde $G_{q_{\phi_1 \vee \phi_2},q'}$, pour toute garde $G_{q_i,q'}$ de l'automate $\mathcal{A}_2$ où q_i est l'état initial de $\mathcal{A}_2$, on ajoute la garde $G_{q_{\phi_1 \vee \phi_2},q'}$.

Les états finaux restent les mêmes.

$\mathcal{A}$ admet les mêmes modèles que $\phi_1 \vee \phi_2$, c'est un automate plat à compteurs donc $\phi_1 \vee \phi_2$ est modèle-plate (voir figure 2.6). ■

Lemme 9 *Si ϕ est une formule modèle-plate, alors $X\phi$ est une formule modèle-plate.*

Preuve : ϕ a pour représentant un automate plat à compteurs $\mathcal{A}$, on construit $\mathcal{A}'$ de la manière suivante : on ajoute un état $q_{X\phi}$, seul état initial du nouvel automate et on ajoute la garde $G_{q_{X\phi},q} = true$ pour tout q état initial de $\mathcal{A}$. Les états finaux restent les mêmes. (voir figure 2.7)

$\mathcal{A}'$ admet les mêmes modèles que $X\phi$, c'est un automate plat à compteurs donc $X\phi$ est modèle-plate. ■

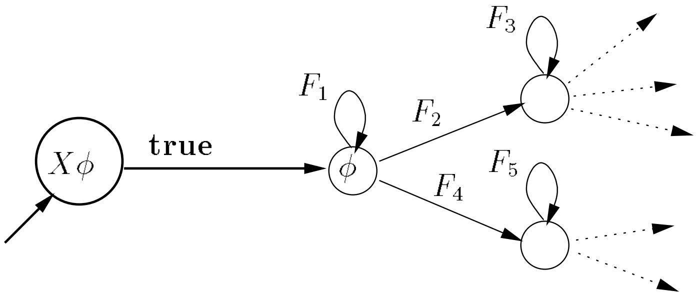


FIG. 2.7 – *Automate de $X\phi$*

Lemme 10 *Si ϕ_1 et ϕ_2 sont des formules modèle-plates alors $\phi_1 \wedge \phi_2$ est modèle-plate.*

Preuve :

ϕ_1 a pour représentant un automate plat à compteurs $\mathcal{A}_1$, ϕ_2 a pour représentant un automate plat à compteurs $\mathcal{A}_2$.

On construit l'automate intersection $\mathcal{A}$ comme on construit l'intersection de deux automates : $Q = Q_1 \times Q_2$, $q_i = q_i^1 \times q_i^2$, si $G_{q_1,q_1'}$ est une garde de $\mathcal{A}_1$, $G_{q_2,q_2'}$ une garde de $\mathcal{A}_2$ alors $G_{(q_1,q_2),(q_1',q_2')} = G_{q_1,q_1'} \wedge G_{q_2,q_2'}$ est une garde de $\mathcal{A}$, $F = F_1 \times F_2$.

Il est inutile ici de construire en quelque sorte une copie de l'automate intersection comme on le fait dans le cas de deux automates de Büchi quelconques. En effet, Comme $\mathcal{A}_1$ et $\mathcal{A}_2$ sont des automates plats, un chemin infini réussi dans l'automate est un chemin qui, à partir d'un certain rang reste tout le temps dans un même état final. Donc si un mot admet à la fois un chemin réussi dans $\mathcal{A}_1$ (qui reste dans l'état final q_{f_1}) et un chemin réussi dans $\mathcal{A}_2$ (qui reste dans l'état final q_{f_2}), il admet un chemin dans $\mathcal{A}$ qui à partir d'un certain rang reste toujours dans l'état final (q_{f_1}, q_{f_2}), c'est donc un chemin réussi.

$\mathcal{A}$ ainsi construit admet donc bien les mêmes modèles que $\phi_1 \wedge \phi_2$. Il reste à montrer que c'est bien un automate plat à compteurs.

Soit $>_1$ l'ordre strict associé à $\mathcal{A}_1$ (automate plat), $>_2$ l'ordre strict associé à $\mathcal{A}_2$, on définit $>$ sur $Q_1 \times Q_2$ comme l'ordre lexicographique associé : $(q_1, q_2) < (q_1', q_2')$ ssi $q_1 <_1 q_1'$ ou si $q_1 = q_1'$ et $q_2 <_1 q_2'$. Si il existe $G_{(q_1,q_2),(q_1',q_2')}$, avec $(q_1, q_2) \neq (q_1', q_2')$ alors $G_{(q_1,q_2),(q_1',q_2')} = G_{q_1,q_1'} \wedge G_{q_2,q_2'}$ donc $q_1 \leq q_1'$ et $q_2 \leq q_2'$
donc $(q_1, q_2) \leq (q_1', q_2')$
et donc $(q_1, q_2) < (q_1', q_2')$.

De plus si $G_{(q_1,q_2),(q_1,q_2)}$ et $G'_{(q_1,q_2),(q_1,q_2)}$ sont deux boucles sur un même état alors comme $G_{(q_1,q_2),(q_1,q_2)} = G_{q_1,q_1} \wedge G_{q_2,q_2}$ et $G'_{(q_1,q_2),(q_1,q_2)} = G'_{q_1,q_1} \wedge G'_{q_2,q_2}$, $G_{q_1,q_1} = G'_{q_1,q_1}$ et $G_{q_2,q_2} = G'_{q_2,q_2}$ donc $G_{(q_1,q_2),(q_1,q_2)} = G'_{(q_1,q_2),(q_1,q_2)}$.
$\mathcal{A}$ est bien un automate plat. ■

Lemme 11 *Si ϕ est une formule modèle-plate, ψ une brique alors $\psi \mathcal{U} \phi$ est modèle-plate.*

Preuve : ϕ est une formule modèle-plate, soit $\mathcal{A}$ un automate plat représentant ϕ.

$w \models \psi \mathcal{U} \phi$ ssi (par définition) $\widetilde{w}, 0 \models \phi$ ssi $\begin{array}{l} \widetilde{w}, 0 \models \phi \quad \text{ou} \\ \widetilde{w}, 0 \models \psi \text{ et } \widetilde{w}, 1 \models \psi \mathcal{U} \phi \end{array}$

Or $\widetilde{w}, 0 \models \psi$ avec ψ une brique ssi $\widetilde{w}(0) = (a, v, w)$ et $\psi(a, v, w)$ où $w(0)$ est la première lettre de $\widetilde{w}$.

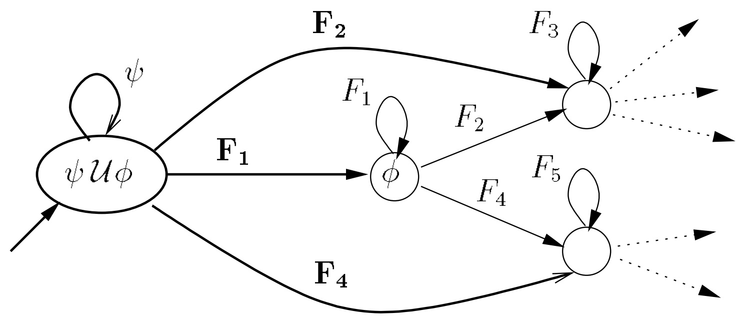


FIG. 2.8 – *Automate de $\psi\mathcal{U}\phi$*

On construit donc $\mathcal{A}'$ l'automate suivant : (Voir figure 2.8)

- On ajoute un état $q_{\psi\,\mathcal{U}\phi}$ à l'automate $\mathcal{A}$ qui devient le nouvel état initial.
- Si $G_{q_0,q}$ est une garde de $\mathcal{A}$ où q_0 est l'état initial, alors on ajoute la garde $G_{q_{\psi\,\mathcal{U}\phi},q} = G_{q_0,q}$ à l'automate.
- On pose $G_{q_{\psi\,\mathcal{U}\phi},q_{\psi\,\mathcal{U}\phi}} = \psi$.
- Les états finaux restent les mêmes.

On obtient bien un automate plat à compteurs dont toutes les gardes sont dans $\mathcal{P}_r$.

De plus, $\mathcal{A}'$ a bien les mêmes modèles que $\psi\mathcal{U}\phi$. En effet, $w \models \psi\mathcal{U}\phi$ ssi

- Ou bien $w \models \phi$ ce qui se vérifie en partant de $q_{\psi\,\mathcal{U}\phi}$ comme si c'était l'état initial de $\mathcal{A}$, automate modèle de ϕ,
- Ou bien $w \models \psi$ ce qui ce vérifie sur la première lettre : il faut que le premier vecteur du mot et le deuxième vérifient la garde ψ et $\widetilde{w}, 1 \models \psi\mathcal{U}\phi$ ce qui se vérifie en retournant à l'état initial après avoir lu la première lettre. (Car $\widetilde{w}, 1 = \widetilde{w'}, 0$ où w' est le sous mot obtenu à partir de w en enlevant la première lettre.)

Conclusion : $\psi\mathcal{U}\phi$ est modèle-plate. ■

Lemme 12 *Soit ψ une brique, alors ψ est modèle-plate.*

Preuve : ψ est une brique donc ψ est une garde possible d'un automate à compteurs. ψ est de la forme $p_1 \wedge \cdots p_n \wedge \neg p'_1 \wedge \cdots \neg p'_m \wedge \phi$ où ϕ est dans $\mathcal{P}_r$. Donc $w, 0 \models p$ ssi $w(0) = (a, u, w)$ et $\forall i \, \forall j \quad p_i \in a,\, p_j \notin a,\, \phi(u, v)$.

L'automate qui représente ψ est l'automate de la figure 2.9.

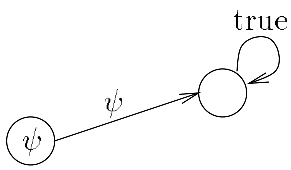


FIG. 2.9 – *Automate modèle de ψ*

Conclusion : ψ est modèle-plate. ■

Lemme 13 *Si ψ est une brique, $\neg(true\mathcal{U}\neg\psi)$ est modèle-plate.*

Preuve : $\widetilde{w}, 0 \models \neg(true\,\mathcal{U}\neg\psi)$ si et seulement si $\forall i \quad \widetilde{w}, i \models \psi$ et ψ est une brique donc l'automate $\mathcal{A}$ de la figure 2.10 a les mêmes modèles que $\neg(true\,\mathcal{U}\neg\psi)$. $\mathcal{A}$ est un automate plat à compteurs donc $\neg(true\,\mathcal{U}\neg\psi)$ est modèle-plate.

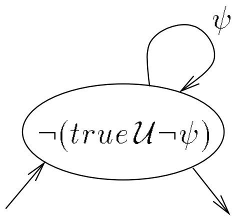


FIG. 2.10 – *Automate modèle de* $\neg(true\,\mathcal{U}\neg\psi)$

∎

A l'aide de tous ces lemmes, on peut maintenant démontrer facilement la première partie du résultat annoncé à savoir que toute formule de $\mathcal{L}_p$ admet un automate plat à compteurs pour représentant (i.e. tel qu'il reconnaisse exactement les mêmes modèles que la formule).

Théorème 12 *Soit ϕ une formule de $\mathcal{L}_p$, alors ϕ est modèle-plate.*

Preuve : Nous allons montrer ce théorème par induction sur la formule :

- Si ϕ est une brique, alors ϕ est modèle plate, cf lemme 12.
- Si ψ est une brique, ϕ_1 et ϕ_2 sont modèle-plates, alors :
  - $\phi_1 \vee \phi_2$, $\phi_1 \wedge \phi_2$ sont modèles plates, cf lemmes 8 et 10,
  - $X\phi_1$ est modèle-plate, cf lemme 9,
  - $\psi\mathcal{U}\phi_1$ est modèle-plate, cf lemme 11.
- Si ψ est une brique, $\neg(true\,\mathcal{U}\neg\psi)$ est modèle-plate, cf lemme 13.

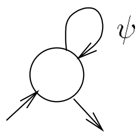
Conclusion : si ϕ est une formule de $\mathcal{L}_p$, alors ϕ est modèle-plate. ∎

Il reste à montrer la réciproque, à savoir que tout automate plat à compteurs estle représentant d'une formule de $\mathcal{L}_p$.

Théorème 13 *Si $\mathcal{A}$ est un plat à compteurs, alors il existe une formule de $\mathcal{L}_p$ qui admette exactement les mêmes modèles.*

Preuve : La démonstration se fait par récurrence sur la taille de l'automate $\mathcal{A}$. On peut supposer sans perte de généralité que tous les états finaux de l'automate n'ont pas de successeur.

- Si $\mathcal{A}$ a un unique état, il est de la forme :



  donc $\mathcal{A}$ a les mêmes modèles que $\neg(true\,\mathcal{U}\neg\psi)$.
- Si $\mathcal{A}$ est de la forme : voir figure 2.11 (pas de boucle sur l'état initial),
  L'automate $\mathcal{A}_1$ a les mêmes modèles que ϕ_1 (par hypothèse de récurrence), ..., l'automate $\mathcal{A}_n$ a les mêmes modèles que ϕ_n.
  Donc $\mathcal{A}$ a les mêmes modèles que $(F_1 \wedge X\phi_1) \vee \cdots \vee (F_n \wedge X\phi_n)$.

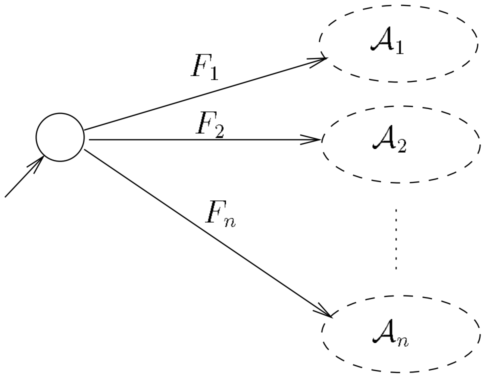


FIG. 2.11 – *Sans boucle*

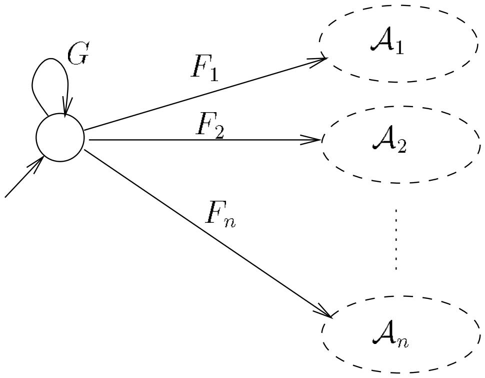


FIG. 2.12 – *Avec boucle*

– Si $\mathcal{A}$ est de la forme : voir figure 2.12 (boucle G sur l'état initial)
  L'automate $\mathcal{A}_1$ a les mêmes modèles que ϕ_1 (par hypothèse de récurrence), ... , L'automate $\mathcal{A}_n$ a les mêmes modèles que ϕ_n.
  Donc $\mathcal{A}$ a les mêmes modèles que $G\mathcal{U}[(F_1 \wedge X\phi_1) \vee \cdots \vee (F_n \wedge X\phi_n)]$.

Conclusion : Tout automate plat à compteurs a les mêmes modèles qu'une des formules de $\mathcal{L}_p$. ■

On vient donc de montrer que la logique $\mathcal{L}_p$ caractérisait les automates plats à compteurs.

2.3 Discussion

L'objet de ce paragraphe est de discuter l'intérêt et le choix de cette logique. En effet, nous avons trouvé une logique qui caractérisait les automates plats à compteurs mais rien ne dit qu'il n'existe pas d'autres logiques caractérisant ces mêmes automates plats à compteurs.

2.3.1 Pas de négation

La logique $\mathcal{L}_p$ possède l'inconvénient de ne pas être close par négation. Cela signifie que l'on ne pourra faire du model-checking qu'avec des négations de formules de $\mathcal{L}_p$ comme on le verra au paragraphe 2.4.

2.3.2 Construction du Until à droite

La construction de $\mathcal{L}_p$ impose que les Until s'imbriquent toujours à droite. Cependant, il est intéressant de remarquer que cette condition se retrouve fréquemment dans des contextes differents :

- Paul Gastin étudie ainsi dans [13] une logique dont la complexité est non élémentaire en fonction du nombre de Until à gauche.
- D. Dams introduit dans [8] une sous classe de LTL : "Flat Linear-time Temporal Logic" qui est une logique construite comme LTL, sans le "next" et avec la restriction que les formules à gauche d'un Until soit une combinaison booléennes de variables propositionnelles : les Until s'imbriquent toujours à droite.
- Il se trouve que la plupart des formules que l'on vérifie effectivement en spécification sont des formules qui n'utilisent pas (ou peu) l'imbrication des Until à gauche.

2.3.3 $\mathcal{L}_p$ ne contient pas tout LTL

Comme nous allons le voir avec le théorème 14, si une logique est telle que chacune de ses formules admet un automate plat à compteurs, alors elle ne peut contenir LTL, en fait elle ne peut même pas contenir la formule : $\neg(true\,\mathcal{U}(\neg(true\,\mathcal{U}p)))$ ("p est vraie infiniment souvent"). Il est donc, a fortiori, impossible qu'une logique caractérisant les automates plats à compteurs puisse contenir tout LTL.

Par contre, on verra dans la partie 2.5 qu'il est tout de même possible d'étendre $\mathcal{L}_p$ de manière à obtenir une logique qui contienne tout LTL et qui soit décidable.

Théorème 14 *Soit $\mathcal{L}$ une logique telle que toute formule de cette logique ait pour modèle un automate plat à compteurs. Alors cette logique ne peut contenir la formule $\neg(true\,\mathcal{U}(\neg(true\,\mathcal{U}p)))$, elle ne peut donc, a fortiori, contenir tout LTL.*

Preuve : Supposons que $\neg(true\,\mathcal{U}(\neg(true\,\mathcal{U}p)))$ ait pour modèle un automate plat à compteurs $\mathcal{A}$.

On considère les mots acceptés par $\mathcal{A}$, en oubliant les vecteurs d'entiers. Le mot $w = (p)(\neg p)(p)(\neg p)\cdots(p)(\neg p)\cdots$ est un mot accepté par $\mathcal{A}$. Comme $\mathcal{A}$ est plat, ce mot stationne à partir d'un certain rang n dans un état final q_f qui a pour garde F. F est une conjonction de formule de $\mathcal{P}_r$ et de variables propositionnelles (ou de leur négation). F ne contient ni p ni $\neg p$ sinon w ne serait pas accepté. Donc $w' = w(0)\cdots w(n)(\neg p)(\neg p)\cdots(\neg p)\cdots$ est accepté par l'automate (on garde les mêmes vecteurs d'entiers, on ne modifie que les variables), contradiction.

Conclusion : aucune logique ayant des automates plats pour représentants ne peut contenir $\neg(true\,\mathcal{U}(\neg(true\,\mathcal{U}p)))$ et ne peut donc pas contenir tout LTL. ■

Corollaire 2 *$\mathcal{L}_p$ est moins expressive que $LTL + \mathcal{P}res$.*

De plus, même en étendant la définition d'automate plat à "il existe au plus une boucle sur chaque état" (mais qui peut éventuellement passer par d'autres états intermédiaires), le théorème 14 reste vrai.

En effet, on suppose de même que $\neg(true\,\mathcal{U}(\neg(true\,\mathcal{U}p)))$ a pour modèle un automate plat "étendu" à compteurs $\mathcal{A}$ et on considère toutes les boucles qui contiennent un état final, elles sont en nombre fini car l'automate est "plat étendu", il n'y donc pas de boucles imbriquées. Soit P un nombre premier avec chacune des longueurs de ces boucles. Le mot

$$w = \underbrace{(p)(p)\cdots(p)(\neg p)}_{P \text{ termes}}\cdots\underbrace{(p)(p)\cdots(p)(\neg p)}_{P \text{ termes}}\cdots$$

est accepté par $\mathcal{A}$ donc chacune des gardes de chacunes des boucles considérées ne contient ni p ni $\neg p$ et donc il existe un mot se terminant par une infinité de $\neg p$ qui est accepté, contradiction.

2.3.4 $LTL + \mathcal{P}res$ n'est pas une logique décidable

On vient de montrer que $\mathcal{L}_p$ est strictement moins expressive que $LTL + \mathcal{P}res$ et qu'aucune logique caractérisant les automates plats à compteurs ne peut contenir tout LTL. Il est intéressant de noter que, même si on cherche uniquement une logique qui soit décidable, il est impératif de restreindre $LTL + \mathcal{P}res$, car celle-ci est indécidable d'après le théorème qui suit.

Théorème 15 *$LTL + \mathcal{P}res$ est indécidable, i.e., la satisfaisabilité d'une formule de $LTL + \mathcal{P}res$ est indécidable.*

Preuve : Pour démontrer ce résultat, nous allons montrer que l'arrêt d'une machine de Minski se réduit au problème de la satisfaisabilité d'une formule de $LTL + \mathcal{P}res$.

Théorème 16 *[15] Etant donné n, l'arrêt d'une machine de Minski à deux compteurs sur $\begin{pmatrix} n \\ 0 \end{pmatrix}$ est indécidable.*

Nous allons tout d'abord montrer le lemme suivant:

Lemme 14 *Soit $\mathcal{A}$ un automate plat à k compteurs. Il existe une formule ϕ de $LTL + \mathcal{P}res$ à $k+1$ variables telle que*

$$w \models \mathcal{A} \quad ssi \quad \exists w' \;\; \pi(w') = w \wedge w' \models \phi$$

où w est mot fini sur l'alphabet $\mathbb{N}^k$: $w \in (\mathbb{N}^k)^*$, w' un mot fini sur l'alphabet N^{k+1} : $w' \in (\mathbb{N}^{k+1})^*$, π la projection sur les k premières composantes.

Preuve : $\mathcal{A} = (Q, q_0, F, G)$, chaque $G_{q,q'}$ est représentable par une formule de $\mathcal{P}_r$, on confond donc l'ensemble $G_{q,q'}$ et cette formule.

Quitte à faire une copie des états finaux, on peut supposer qu'ils n'ont aucun successeur : si q_f est un état final, on crèe l'état final q'_f qui n'a aucune transition sortante et qui a les mêmes transitions entrantes que q_f. q'_f est alors un nouvel état final et q_f n'en est plus un.

La formule ϕ va coder la valeur des k compteurs sur ses k premières variables et l'état de contrôle sur sa dernière variable, notée c (on supposer que les états de l'automate sont codés par des entiers). Elle va coder que :

- le premier état de contrôle est bien l'état initial de l'automate,
- on peut passer de la configuration V à V' en passant de l'état de contrôle q à q' si et seulement si $G_{q,q'}(V.V')$,
- on s'arrête dans un état final.

Soit

$$\phi := c = q_0 \wedge \left(\left[\bigwedge_i c = q_i \Rightarrow \bigvee_{q_i \xrightarrow{G} q_j} \left[G_{q_i,q_j}(\vec{x}, \vec{x'}) \vee c' = q_j\right]\right] \mathcal{U} \left[\bigvee_{q_f \in F} c = q_f\right]\right)$$

Si $w \models \mathcal{A}$ alors il existe $q_1, \ldots, q_n$, un chemin dans l'automate tel que

$$q_1 = q_0,\ q_n \in F, \quad \forall i \quad G_{q_i,q_{i+1}}(w_i, w_{i+1}).$$

On choisit w' tel le mot formé par la lecture de ses k premières composante soit w et que le mot formé par la lecture de sa dernière composante soit $q_1 \cdots q_n$.

$\pi(w') = w$ et $w' \models \phi$.

Réciproquement, si $w' \models \phi$, $|w'| = n$, on a que $c_1 = q_0$, $c_n \in F$ car comme $w' \models \phi$, il existe i tel que $c_i \in F$ or aucun état final n'a de successeur donc nécessairement, $i = n$. Donc $\pi(w')$ est accepté dans $\mathcal{A}$ par le chemin $c_1 \cdots c_n$: $\pi(w') \models \mathcal{A}$.

■

On a la propriété suivante :

$$\exists w \quad w \models \mathcal{A} \text{ ssi } \exists w' \quad w' \models \phi \tag{2.1}$$

On considère donc maintenant une machine de Minski à deux compteurs, elle peut être considérée comme un automate $\mathcal{A}$ à compteurs car l'incrémentation, la décrémentation et le test à zéro des variables sont des formules de $\mathcal{P}_r$.

On complète cette machine de Minski en l'automate $\mathcal{A}'$ de la manière suivante :

- on ajoute un état "poubelle" q_p
- soit q est un état de $\mathcal{A}$ et $\phi_1, \ldots, \phi_n$ les transitions sortantes de q, chaque ϕ_i induit des conditions sur les variables d'entrée :
  - $x' = x + 1$ n'induit aucune condition,
  - $x' = x - 1$ induit $x > 0$,
  - $x = 0$ induit $x = 0$

  la transition ϕ_i peut être franchie si et seulement si on a la condition ψ_i qui s'exprime comme une conjonction de formules de la forme $x > 0$ et $x = 0$.

  On ajoute alors la transition $\neg\psi_1 \wedge \cdots \wedge \neg\psi_n$ de q vers q_b, cette transition ne peut être franchie que s'il est impossible de franchir aucune des autres.
- le seul état final de l'automate $\mathcal{A}'$ est q_b.

Soit ϕ la formule associée à $\mathcal{A}'$ par le lemme 14.

On a alors que $\mathcal{A}$ s'arrête sur $\begin{pmatrix} n \\ 0 \end{pmatrix}$

ssi q_b est accessible dans $\mathcal{A}'$ en partant de $\begin{matrix} n \\ 0 \end{matrix}$, i.e.,

ssi il existe w tel que $w = \begin{pmatrix} n \\ 0 \end{pmatrix} w'$ et $w \models \mathcal{A}'$,

ssi il existe $w' = \begin{pmatrix} n \\ 0 \\ q_0 \end{pmatrix} w''$ tel que $w' \models \phi$

ssi $\phi' = \phi \wedge (x_1 = n) \wedge (x_2 = 0)$ est satisfaisable.

ϕ' est une formule de $LTL + \mathcal{P}res$, donc on vient de montrer que la satisfaisabilité d'une formule de $LTL + \mathcal{P}res$ est indécidable. ■

2.3.5 Remarque

Les briques de $\mathcal{L}_p$ sont construites à partir de variables propositionnelles et de formules de $\mathcal{P}_r$. En fait, on aurait pu prendre n'importe quelle famille $\mathcal{F}$ de formules à la place de $\mathcal{P}_r$, on aurait alors obtenu la caractérisation des automates plats à compteurs dont les gardes sont des conjonctions de variables propositionnelles (ou de leur négation) et de formules de $\mathcal{F}$. Mais une telle transformation ne préservera pas nécessairement la décidabilité de la logique.

En effet, pour montrer que $\mathcal{L}_p$ est décidable, on va fortement utiliser les résultats de H. Comon et Y. Jurski sur les automates plats à compteurs dont les gardes sont dans $\mathcal{P}_r$.

2.4 Model-checking avec $\mathcal{L}_p$

Nous allons tout d'abord montrer que la satisfaisabilité d'une formule de $\mathcal{L}_p$ est décidable en utilisant les résultats de H. Comon et Y. Jurski sur les automates plats à compteurs. Il découlera de ce premier résultat que le model-checking sur les automates plats à compteurs avec des formules qui s'écrivent comme des négations de formules de $\mathcal{L}_p$ est décidable.

Théorème 17 *Soit ϕ une formule de $\mathcal{L}_p$, on peut décider si ϕ est satisfaisable.*

Preuve : ϕ est une formule de $\mathcal{L}_p$ donc, d'après le théorème 12, ϕ admet un automate plat à compteur $\mathcal{A}$ pour modèle. ϕ est satisfaisable si et seulement si $\mathcal{A}$ admet au moins un chemin infini.

Nous allons montrer que, étant donné un automate plat à compteur $\mathcal{A}$, "il existe un chemin infini accepté par $\mathcal{A}$" est expressible dans Presburger.

En effet, un chemin infini accepté par $\mathcal{A}$ est un chemin qui à partir d'un certain moment, atteint un état final de $\mathcal{A}$ et y reste indéfiniment. Donc il existe un chemin infini accepté par $\mathcal{A}$ si et seulement il existe un état final de $\mathcal{A}$, accessible, qui accepte un mot infini à partir du sous mot considéré.

Soient $q_1^f, \ldots, q_n^f$ les états finaux de $\mathcal{A}$ et $\phi_{q_i^f}(V)$ la formule : la boucle sur q_i^f est franchissable infiniment de fois à partir du vecteur V. D'après le théorème 11, $\phi_{q_i^f}(V)$ est expressible dans Presburger.

Soit $\phi_i(x, x')$, la relation d'accessibilité entre q_o (état initial de l'automate') et q_i^f, c'est-à-dire que $\phi_i(x, x')$ est satisfaite ssi il existe un chemin dans $\mathcal{A}$ qui commence par le vecteur x dans l'état q_0 et qui finit par le vecteur x' dans l'état q_i^f. D'après le théorème 10, $\phi_i(x, x')$ est expressible dans Presburger.

Il existe un chemin infini accepté par $\mathcal{A}$ s'exprime donc par la formule suivante :

$$\exists x\, \exists x' \quad \bigvee_{i=1}^{n} \left[\phi_i(x, x') \wedge \phi_{q_i^f}(x')\right]$$

qui dit exactement qu'il existe un état final accessible depuis q_0 tel que cet état final accepte un mot infini à partir du sous mot considéré.

On a donc démontré que, étant donné une formule $\phi \in \mathcal{L}_p$, "ϕ est satisfaisable" s'exprime dans l'arithmétique de Presburger et est donc, a fortiori, décidable. ■

On va maintenant pouvoir montrer facilement que l'on peut faire du model-checking sur les automates plats à compteurs avec des négations de formules de $\mathcal{L}_p$.

Théorème 18 *Soit ϕ une formule de $\mathcal{L}_p$, $\mathcal{A}$ un automate plat à compteurs, alors $\mathcal{A} \models \neg\phi$ est décidable.*

Preuve :

$$\mathcal{A} \models \neg\phi \text{ ssi } \mathcal{L}_{\mathcal{A}} \cap \mathcal{L}_{\phi} = \emptyset$$

où $\mathcal{L}_{\mathcal{A}}$ et $\mathcal{L}_{\phi}$ sont respectivement les langages de mots acceptés par $\mathcal{A}$ et ϕ.

On construit donc $\mathcal{A}_{\phi}$ l'automate plat à compteurs modèle de ϕ et $\mathcal{A}'$ l'automate intersection de $\mathcal{A}$ et $\mathcal{A}_{\phi}$.

$$\mathcal{L}_{\mathcal{A}} \cap \mathcal{L}_{\phi} = \mathcal{L}_{\mathcal{A}'}$$

et $\mathcal{L}_{\mathcal{A}'} = \emptyset$ si et seulement si $\mathcal{A}'$ n'accepte aucun chemin infini ce qui est décidable d'après le théorème précédent (17).

Conclusion : $\mathcal{A} \models \neg\phi$ est décidable. ∎

2.5 Une extension de $\mathcal{L}_p$

On vient de voir que $\mathcal{L}_p$ est une logique décidable, qui caractérise exactement les automates plats. Cependant, on cherchait au départ une extension de LTL encore décidable. Pour cela, on va définir $\mathcal{L}_p^+$, une logique qui étend à la fois LTL et $\mathcal{L}_p$ et qui reste décidable. Bien entendu, une telle logique ne caractérise plus les automates plats à compteurs.

2.5.1 La logique $\mathcal{L}_p^+$ et ses modèles

Définition 13 *Soit $\mathcal{V}$ un ensemble de variables propositionnelles. On définit la logique $\mathcal{L}_p^+$ (ensemble des formules* plates étendues *de la manière suivante :*

- *Briques :*
  - *$\phi \in \mathcal{P}_r$, $p, \neg p$ où $p \in \mathcal{V}$, sont des briques,*
  - *si b_1 et b_2 sont des briques alors $b_1 \wedge b_2$ est une brique.*
- *Formules élémentaires :*
  - *Si b est une brique, b est une formule élémentaire,*
  - *Si ϕ est une formule de LTL, alors, ϕ est une formule élémentaire,*
  - *Si ϕ_1 et ϕ_2 sont des formules élémentaires alors $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$, $\neg\phi_1$ sont des formules élémentaires.*
- *Formules plates étendues : Toutes les formules élémentaires sont plates et si ϕ_1 et ϕ_2 sont plates alors :*
  - *$\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$, $\neg\phi_1$ sont des formules plates,*
  - *$X\phi_1$ est une formule plate,*
  - *$b\mathcal{U}\phi_1$ est plate où b est une brique,*
  - *$\neg(true\mathcal{U}\neg b)$ est plate, où b est une brique.*

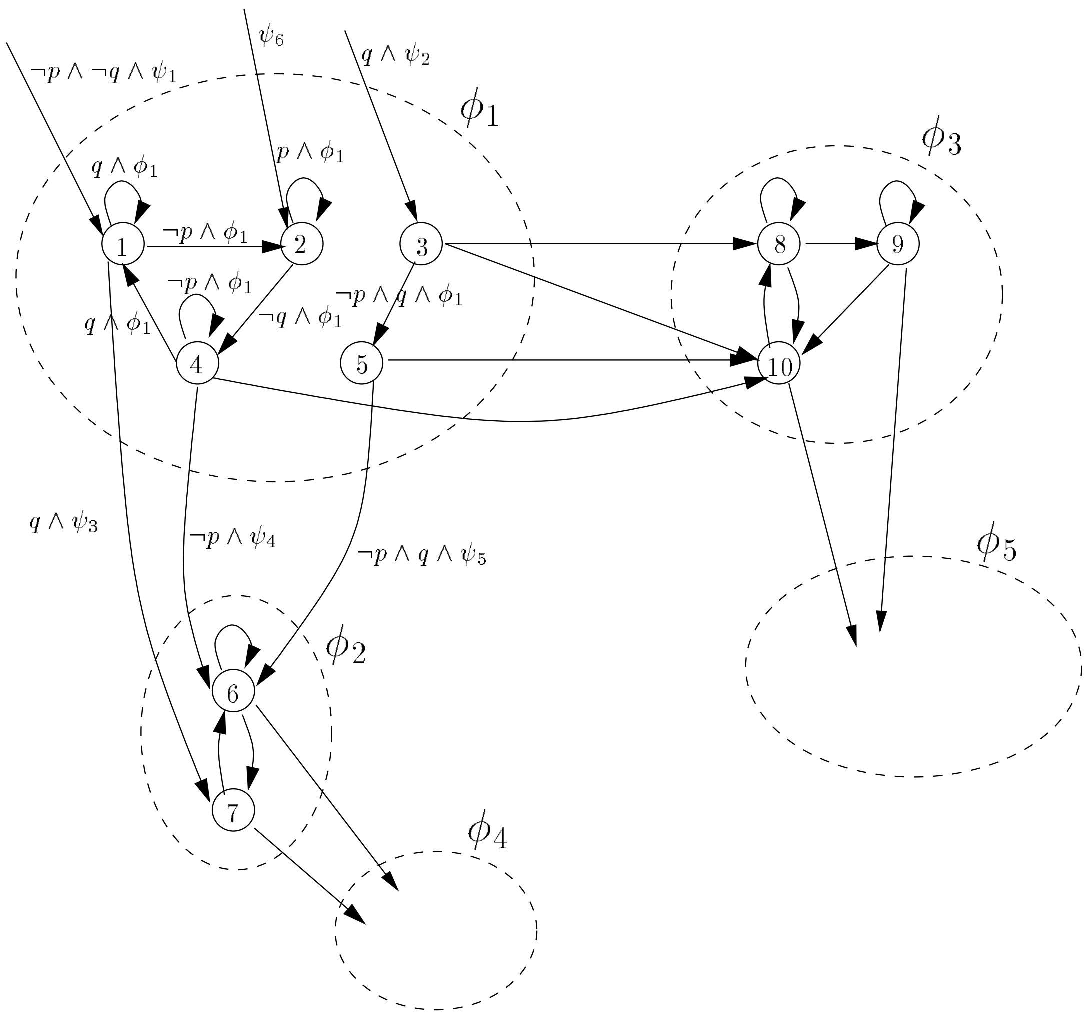


FIG. 2.13 – *Automate à compteurs plat par morceaux*

Il reste que cette logique n'est pas close par négation (par contre, les formules élémentaires le sont), mais elle contient bien tout LTL.

Pour montrer qu'une telle logique est décidable, on va commencer par lui associer une forme d'automates.

Définition 14 *On dira que $\mathcal{A}$ est* plat par morceaux *si $\mathcal{A}$ est un automate à compteurs (Q, I, F, T) et si Q se décompose en une union disjointe de Q_i telle que :*

- *A chaque ensemble d'états $Q_i$est associée une formule ϕ_i de $\mathcal{P}_r$.*
- *Si $\exists q \in Q_i$, $\exists q' \in Q_j$, $i \neq j$, tels que $q \xrightarrow{G} q'$, alors $i < j$.*
- *Si $q \xrightarrow{G} q'$ avec $q, q' \in Q_i$, alors $G = \phi_i \wedge \psi$ où ψ est une conjonction de variables propositionnelles ou de leur négation.*

(voir exemple sur la figure 2.13).

Remarque : Les automates plats à compteurs sont des automates à compteurs plats par morceaux (dans lesquels chaque Q_i est un simgleton).

On va montrer que toute formule de $\mathcal{L}_p^+$ admet un automate à compteurs plat par morceaux pour représentant (i.e., il existe un automate à compteurs plat par morceaux qui reconnaisse exactement les mêmes modèles) . Pour cela, on va commencer par démontrer quelques lemmes.

Lemme 15 *Si ϕ_1 et ϕ_2 ont les mêmes modèles qu'un automate à compteurs plat par morceaux alors $\phi_1 \vee \phi_2$ a les mêmes modèles qu'un automate à compteurs plat par morceaux.*

Preuve : ϕ_1 a les mêmes modèles qu'un automate à compteurs plat par morceaux $\mathcal{A}_1 = (Q^1, I^1, F^1, T^1)$, ϕ_2 a les mêmes modèles qu'un automate à compteurs plat par morceaux $\mathcal{A}_2 = (Q^2, I^2, F^2, T^2)$.

On procède exactement de la même manière que pour l'union de deux automates plats à compteurs (voir démonstration du lemme 8), on obtient un automate à compteurs $\mathcal{A} = (Q, I, F, T)$.

On a alors

$$Q = Q^1 \cup Q^2 \cup \{q_{\phi_1 \vee \phi_2}\} = Q_1^1 \cup \cdots \cup Q_m^1 \cup Q_1^2 \cup \cdots \cup Q_n^2 \cup \{q_{\phi_1 \vee \phi_2}\}.$$

On renumérote les Q_i de la manière suivante : $\{q_{\phi_1 \vee \phi_2}\} = Q_1$, $Q_j^1 = Q_{j+1}$, $Q_j^2 = Q_{j+m+1}$. Avec ce découpage, $\mathcal{A}$ est un automate à compteurs plat par morceaux.

Conclusion : Il existe bien un automate à compteurs plat par morceaux qui admette les mêmes modèles que $\phi_1 \vee \phi_2$.

∎

Lemme 16 *Si ϕ a pour représentant un automate à compteurs plat par morceaux, alors $X\phi$ a pour représentant un automate à compteurs plat par morceaux.*

Preuve : ϕ a pour représentant un automate à compteurs plat par morceaux $\mathcal{A}$, on construit un automate à compteurs $\mathcal{A}' = (Q', I', F', T')$ qui a les mêmes modèles que $X\phi$ comme dans la démonstration du lemme 9.

$\{q_{X\phi}\} \cup Q_1 \cup \cdots \cup Q_n$ où les Q_i forment une partition des états de Q, est une partition des états de Q' avec $\{q_{X\phi}\}$ le premier ensemble de la partition, les autres ensembles étant ensuite laissés dans le même ordre. On obtient bien une partition compatible avec la définition d'un automate à compteurs plat par morceaux, donc $X\phi$ a pour représentant un automate à compteurs plat par morceaux.

∎

Lemme 17 *Si ϕ a pour représentant un automate à compteurs plat par morceaux et si b est une brique, alors $b\mathcal{U}\phi$ a pour représentant un automate à compteurs plat par morceaux.*

Preuve : On reprend encore une fois la même construction qu'au lemme 11 en numérotant $\{q_{b\mathcal{U}\phi}\}$ comme premier ensemble de la nouvelle partition et on obtient bien de nouveau un automate à compteurs plat par morceaux.

∎

Lemme 18 *Si ϕ_1 et ϕ_2 ont pour représentant un automate à compteurs plat par morceaux alors $\phi_1 \wedge \phi_2$ a pour représentant un automate à compteurs plat par morceaux.*

Preuve : ϕ_1 a pour représentant un automate à compteurs plat par morceaux $\mathcal{A}_1 = (Q^1, \{s_1\}, F^1, T^1)$, avec $Q_1^1, \ldots, Q_m^1$ une partition de Q^1; $\phi_1^1, \cdots, \phi_m^1$ les formules associées, ϕ_1 a pour représentant un automate à compteurs plat par morceaux $\mathcal{A}_2 = (Q^2, \{s_2\}, F^2, T^2)$.

On procède comme pour l'intersection de deux automates de Büchi (mots infinis). En effet, les automates n'étant plus plats, on ne peut pas se contenter d'une intersection "simple".

On construit donc

$$\mathcal{A} = (Q^1 \times Q^2 \times \{1,2\}, \{s_1\} \times \{s_2\} \times \{1\}, F_1 \times Q_2 \times \{1\}, T)$$

avec

$$(s', t', j) \in T((s,t,i), \phi_1 \wedge \phi_2) \text{ si } \begin{array}{l} s' \in T_1(s, \phi_1) \\ t' \in T_2(t, \phi_2) \\ i = j \end{array}$$

sauf si $i = 1$ et $s \in F_1$, dans ce cas, $j = 2$,
oui si $i = 2$ et $t \in F_2$, dans ce cas, $j = 1$.

$$\bigcup_{i,j} Q_i^1 \times Q_j^2 \times \{1,2\}$$

est une partition de $Q^1 \times Q^2 \times \{1,2\}$.

A chaque $Q_i^1 \times Q_j^2 \times \{1,2\} = Q_i^1 \times Q_j^2 \times \{1\} \cup Q_i^1 \times Q_j^2 \times \{2\}$, on associe la formule $\phi_i^1 \wedge \phi_j^2$ et on ordonne les $Q_i^1 \times Q_j^2 \times \{k\}$ par ordre lexicographique sur (i,j).

En effet, supposons $(s', t', k) \in T((s,t,l), \alpha_1 \wedge \alpha_2)$ avec $s' \in Q_{i'}, t' \in Q_{j'}, s \in Q_i, t \in Q_j$:

- Si $(i', j') \neq (i, j)$ alors :
  - Si $i \neq i'$, comme $s' \in T_1(s, \alpha_1)$, on a $i < i'$ donc $(i,j) < (i', j')$,
  - Si $i = i'$, $j \neq j'$, comme $t' \in T_2(t, \alpha_2)$, on a $j < j'$ et donc $(i,j) < (i', j')$.

  On a donc bien que si $\exists q \in Q_i$, $\exists q' \in Q_j$ tels que $q \to q'$, alors $i < j$.
- Si $(i', j') = (i, j)$, alors comme $s' \in T_1(s, \alpha_1)$ avec $s, s' \in Q_i^1$, on a $\alpha_1 = \phi_i^1 \wedge \psi_1$ où ψ_1 est une conjonction de variables propositionnelles ou de leur négation.

  De même, comme $t' \in T_2(t, \alpha_2)$, avec $t, t' \in Q_i^2$, on a $\alpha_2 = \phi_i^2 \wedge \psi_2$ où ψ_2 est une conjonction de variables propositionnelles ou de leur négation, donc $\alpha_1 \wedge \alpha_2 = (\phi_i^1 \wedge \phi_i^2) \wedge \psi_1 \wedge \psi_2$.

  On a donc bien que si $q \xrightarrow{G} q'$ avec $q, q' \in Q_i$, alors $G = \phi_i \wedge \psi$ où ψ est une conjonction de variables propositionnelles ou de leur négation.

Conclusion : $\phi_1 \wedge \phi_2$ a pour représentant un automate à compteurs plat par morceaux.

■

Théorème 19 *Toute formule de $\mathcal{L}_p^+$ a pour représentant un automate plat par morceaux.*

Preuve : On procède par induction sur la construction de la formule :

- Si b est une brique, b a pour représentant un automate plat à compteurs d'après le théorème 12.
- Si ϕ est une formule élémentaire alors ϕ a pour représentant un automate à compteurs plat par morceaux. En effet :

  Si ϕ est une formule de LTL, alors d'après les résultats de M. Vardi dans [19], on peut lui associer un automate de Büchi qui reconnaisse exactement le même ensemble de mots infinis.

Or un automate de Büchi est un automate à compteurs plat par morceaux (avec aucun compteur), $Q = Q_1$ et $\phi_1 = true$. Donc ϕ a bien pour représentant un automate à compteurs plat par morceaux.

Si ϕ_1 et ϕ_2 ont pour représentant un automate à compteurs plat par morceaux, alors $\phi_1 \vee \phi_2$ et $\phi_1 \wedge \phi_2$ ont pour représentant un automate à compteurs plat par morceaux d'après les lemmes 15 et 18.

Si ϕ est une formule élementaire et a pour représentant un automate à compteurs plat par morceaux, alors $\neg\phi$ a pour représentant un automate à compteurs plat par morceaux. En effet, la négation d'une formule élémentaire est une conjonction de disjonctions de briques et de formules de LTL, donc elle a bien pour représentant un automate à compteurs plat par morceaux d'après ce qui précède.

– Si ϕ est une formule élémentaire, ϕ a pour représentant un automate à compteurs plat par morceaux.

  Si ϕ_1 et ϕ_2 ont pour représentant un automate à compteurs plat par morceaux, alors $\phi_1 \vee \phi_2$, et $\phi_1 \wedge \phi_2$, ont pour représentant un automate à compteurs plat par morceaux d'après les lemmes 15 et 18.

  Si ϕ a pour représentant un automate à compteurs plat par morceaux et que b est une brique alors $X\phi$ et $b\mathcal{U}\phi$ ont pour représentant un automate à compteurs plat par morceaux d'après les lemmes 16 et 17.

Conclusion : Toute formule de $\mathcal{L}_p^+$ a pour représentant un automate plat par morceaux.

■

2.5.2 model-checking avec $\mathcal{L}_p^+$

Pour montrer que $\mathcal{L}_p^+$ est une classe de formules décidable ou plus présicément que la satisfaisabilité est décidable, on va procéder comme pour $\mathcal{L}_p$: on montre que le vide est décidable pour l'automate qui accepte les modèles d'une formule.

Théorème 20 *Soit $\mathcal{A}$ un automate à compteurs plat par morceaux, alors "il existe un chemin infini dans $\mathcal{A}$" est expressible dans Presburger.*

Preuve : On va se ramener à l'existence d'un chemin infini dans un automate plat à compteurs. En effet, pour ce qui est de la partie propositionnelle des gardes (une conjonction de variables propositionnelles et de leur négation), soit elle contient $p \wedge \neg p$ et on ne peut jamais franchir cette garde, on la supprime, soit cette partie de la garde est toujours satisfaisable quel que soit le chemin emprunté avant.

Ceci n'est pas vrai pour les conditions sur les compteurs : ainsi si $\mathcal{A}$ est l'automate de la figure 2.14, la garde G_3 n'est franchissable que si on est parti de $x = 0$ ou $x = 1$. Par contre, si on ne regarde pas les compteurs, les gardes G_1, G_2 et G_3 sont toujours franchissables.

On ne considère donc plus que les conditions sur les compteurs. L'automate obtenu (voir figure 2.15) n'est alors toujours pas plat.

Cependant, les boucles imbriquées ne concernent qu'une seule garde à la fois, il est donc possible de "d'aplatir" l'automate : voir figure 2.16.

On appelle *noeud* de l'automate à compteurs plat par morceaux $\mathcal{A}$, chaque ensemble Q_i. Il ne peut y avoir de boucles imbriquées qu'à l'intérieur d'un noeud. "En dehors des noeuds", l'automate est plat.

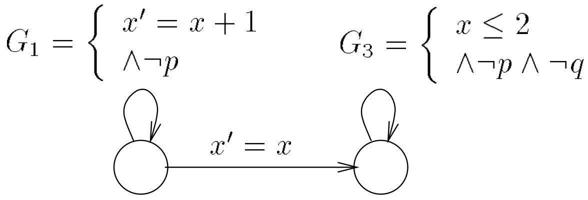


FIG. 2.14 – *Exemple*

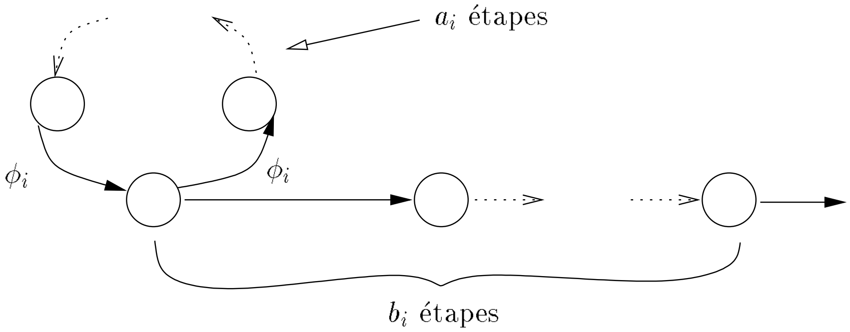
Un état de Q_i est dit *sortant* s'il existe une transition de cet état vers un autre état de Q_j, $i \neq j$, un état q de Q_i est dit *entrant* s'il existe une transition d'un autre état de Q_j, $i \neq j$, vers q.

On transforme alors $\mathcal{A}$ en automate plat à compteurs $\mathcal{A}'$ de la manière suivante : à chaque de chaque noeud Q_i est associé la formule ϕ_i

- On fait autant de copies $Q_i^{q_s}$ de Q_i qu'il y a d'états q_s "sortants" dans Q_i.
- Pour chaque copie $Q_i^{q_s}$, et pour chaque état "entrant" de $Q_i^{q_s}$, on considère l'ensemble des longueurs S des chemins qui vont de q à q_s : c'est un reconnaissable de $\mathbb{N}$, donc une union finie de progression arithmétique.

$$S = \bigcup_{i=1}^{m} \{a_i n + b_i | n \in \mathbb{N}\}$$

Pour chaque $i \in [1 \cdots m]$, on construit :



Cet automate $\mathcal{A}'$ est bien un automate plat à compteurs et il reconnait exactement les mêmes mots que $\mathcal{A}$. Donc $\mathcal{A}$ accepte un mot infini si et seulement si $\mathcal{A}'$ accepte un mot infini ce qui s'exprime dans l'arithmétique de Presburger d'après le théorème 17.

■

De ce résultat de décidabilité sur les automates à compteurs plats par morceaux découle la décidabilité de $\mathcal{L}_p^+$ et du model-checking sur les automates à compteurs plats par morceaux.

Corollaire 3 *La satisfaisabilité d'une formule de $\mathcal{L}_p^+$ est décidable.*

Preuve : Soit ϕ une formule de $\mathcal{L}_p^+$, elle a pour modèle un automate à compteurs plat par morceaux $\mathcal{A}$ d'après le théorème 19. ϕ est satisfaisable si et seulement si $\mathcal{A}$ admet un chemin infini ce qui est décidable d'après le théorème précédent (20).

■

Théorème 21 *Soit ϕ une formule de $\mathcal{L}_p^+$, $\mathcal{A}$ un automate à compteurs plat par morceaux, alors $\mathcal{A} \models \neg\phi$ est décidable.*

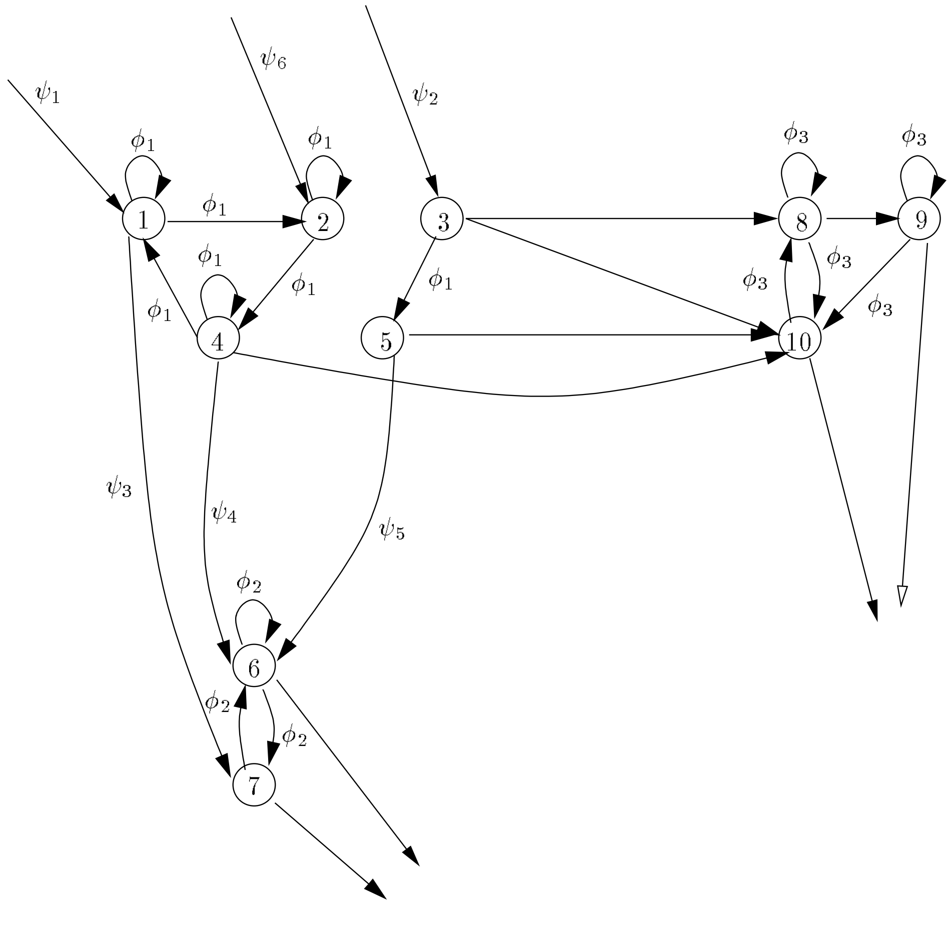


FIG. 2.15 – *Automate obtenu à partir de l'automate 2.13 en oubliant les conditions sur les variables propositionnelles*

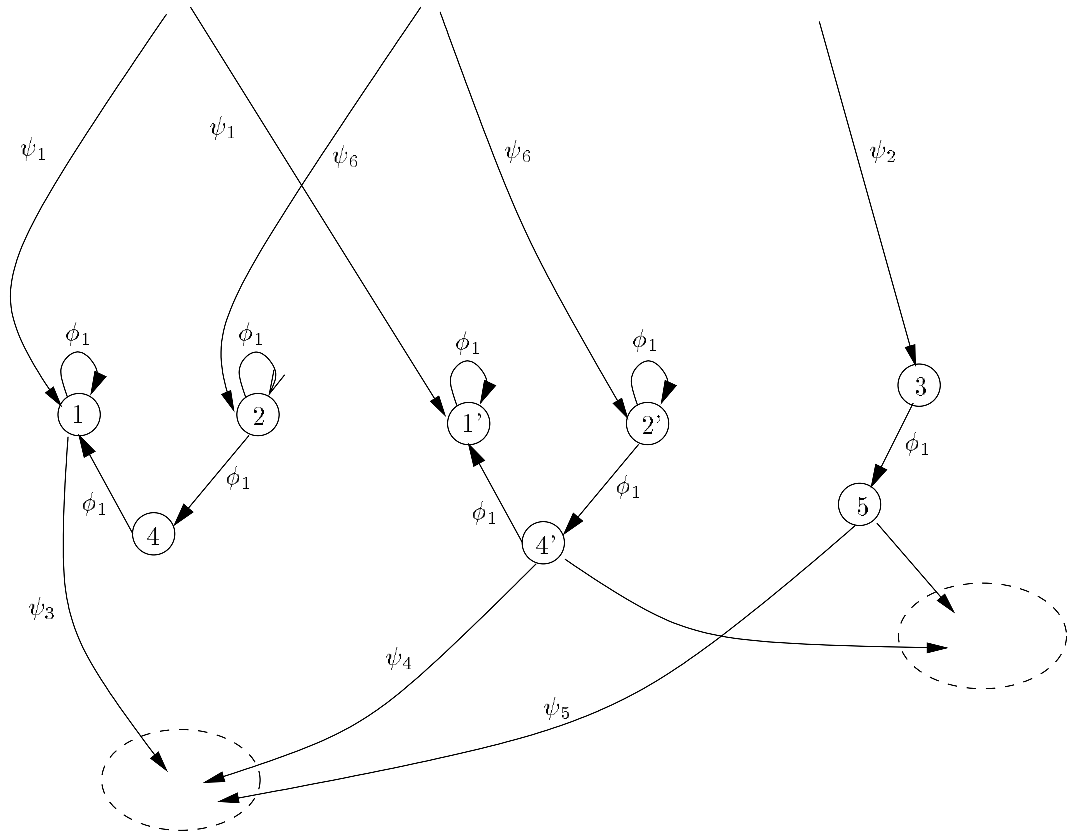


FIG. 2.16 – *Automate obtenu en "aplatissant" l'automate de la figure 2.15 (uniquement en ce qui concerne la partie ϕ_1)*

Preuve : Soit $\mathcal{A}_\phi$ un automate à compteurs plat par morceaux modèle de ϕ,

$$\mathcal{A} \models \neg\phi \text{ ssi } \mathcal{L}_\mathcal{A} \cap \mathcal{L}_{\mathcal{A}_\phi} = \mathcal{L}_{\mathcal{A}\cap\mathcal{A}_\phi} = \emptyset.$$

Or $\mathcal{A} \cap \mathcal{A}_\phi$ est un automate à compteurs plat par morceaux d'après le théorème 18, donc $\mathcal{L}_{\mathcal{A}\cap\mathcal{A}_\phi} = \emptyset$ est décidable d'après le théorème 20. Conclusion : $\mathcal{A} \models \neg\phi$ est décidable.

■

2.6 Une nouvelle caractérisation de l'arithmétique de Presburger

Les résultats de H. Comon et Y. Jurski nous permettent de donner une nouvelle caractérisation de l'arithmétique de Presburger. On remarque tout d'abord que l'ensemble des vecteurs accessibles dans un automate plat à compteurs : $\{V | \exists V_0 \quad V_0 \xrightarrow{*} V\}$ est expressible dans l'arithmétique de Presburger. Et on peut établir une sorte de réciproque à ce résultat : à toute formule de Presburger, on peut associer un automate plat à compteurs tel que l'ensemble des vecteurs satisfaisant la formule soit une projection de l'ensemble des vecteurs accessibles dans l'automate.

Théorème 22 *Soit $\mathcal{A}$ un automate plat à compteurs, alors l'ensemble des vecteurs accessibles dans $\mathcal{A}$ est expressible dans l'arithmétique de Presburger.*

Preuve : D'après le théorème 10 de H. Comon et Y. Jurski, la relation d'accessibilité ϕ dans un automate plat à compteurs s'exprime dans Presburger donc

$$\{V | \exists V_0 \quad V_0 \xrightarrow{*} V\} = \{V | \exists V_0 \quad \phi(V_0, V)\}$$

est expressible dans l'arithmétique de Presburger car $\psi(V) = \exists V_0 \quad \phi(V_0, V)$ est une formule de Presburger.

■

On va maintenant établir la réciproque.

Théorème 23 *Soit $\phi(x_1, \ldots, x_k)$ une formule de Presburger, alors il existe $l \geq k$, il existe un automate plat $\mathcal{A}$ sur l'alphabet $\mathbb{N}^l$ tel que*

$$V \models \phi \textit{ ssi } \exists U \quad U \models \mathcal{A} \wedge V = \pi(U)$$

où π est la projection suivant les k premières variables. On dira qu'une telle formule possède la propriété (*).

Preuve : Toute formule de Presburger se ramène après élimination des quantificateurs à une disjonction de conjonctions de formules de la forme :

- $s = t\,[k]$
- $s < t$

où s et t sont des termes, c'est à dire de la forme $a_1x_1 + \cdots + a_nx_n + b$ avec $a_i, b \in \mathbb{N}$, nx désigne en fait $\underbrace{x + \cdots + x}_{n \text{ fois}}$.

La preuve de ce résultat est présentée (par exemple) dans [20].

Lemme 19 *Si ϕ_1 et ϕ_2 ont la propriété (*), alors $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$ possèdent la propriété (*).*

Preuve : Soit $\mathcal{A}_1$ l'automate plat à compteurs tel que

$$V \models \phi_1 \text{ ssi } \exists U \quad U \models \mathcal{A}_1 \wedge V = \pi(U).$$

Soit $\mathcal{A}_2$ l'automate plat à compteurs tel que

$$V \models \phi_2 \text{ ssi } \exists U \quad U \models \mathcal{A}_2 \wedge V = \pi(U).$$

On renomme les variables de manière à ce que $x_1, \cdots, x_k$ soient les variables partagées par ϕ_1 et ϕ_2, $y_1, \cdots, y_n$ les variables propres à ϕ_1, $y'_1, \cdots, y'_m$ les variables propres à ϕ_2, $z_1, \cdots, z_i$ les variables de $\mathcal{A}_1$ qui ne sont pas celles de ϕ_1, $z'_1, \cdots, z'_j$ les variables de $\mathcal{A}_2$ qui ne sont pas celles de ϕ_2.

On construit $\mathcal{A}'_1$ à partir de $\mathcal{A}_1$ en ajoutant les variables $y'_1, \cdots, y'_m$ et $z'_1, \cdots, z'_j$ sans aucune garde supplémentaire. De même, on construit $\mathcal{A}'_2$ à partir de $\mathcal{A}_2$ en ajoutant les variables $y_1, \cdots, y_n$ et $z_1, \cdots, z_i$.

On construit alors l'intersection et l'union de ces deux automates et leur projection selon les $k + n + m$ variables donne respectivement le langage accepté par $\phi_1 \wedge \phi_2$ et $\phi_1 \vee \phi_2$.

■

Il suffit donc de montrer que toute formule de la forme $s = t[k]$ ou $s < t$ possède la propriété (*), c'est-à-dire qu'il existe un automate plat à compteurs qui reconnaisse le même langage à une projection près.

Nous allons donc construire un automate plat à compteurs qui reconnaît les mêmes mots que

$$a_1x_1 + \cdots a_nx_n + a < b_1y_1 + \cdots + b_my_m + b \tag{2.2}$$

sur la projection de ses $n + m$ premières variables. De même pour

$$a_1x_1 + \cdots a_nx_n + a < b_1y_1 + \cdots + b_my_m + b \quad [k]. \tag{2.3}$$

L'idée de la construction de l'automate reconnaissant les mêmes mots que la formule 2.2 est la suivante : on construit au fur et à mesure l'expression $b_1y_1 + \cdots + b_my_m + b$ sur un autre compteur c puis on lui enlève ensuite petit à petit l'expression $a_1x_1 + \cdots a_nx_n + a$, voir figure 2.17.

Pour l'automate reconnaissant les mêmes mots que la formule 2.3, on construit en fait la disjonction de deux automates : pour le premier, on construit au fur et à mesure l'expression $b_1y_1 + \cdots + b_my_m + b$ sur un autre compteur c puis on lui enlève ensuite petit à petit l'expression $a_1x_1 + \cdots a_nx_n + a$, on enlève ensuite un certain nombre de fois k jusqu'à trouver 0, sur l'autre automate, on fait la même chose mais en enlevant $b_1y_1 + \cdots + b_my_m + b$ à $a_1x_1 + \cdots a_nx_n + a$, voir figure 2.18.

■

On obtient donc bien une nouvelle caractérisation de l'arithmétique de Presburger : tout ensemble de Presburger est la projection d'un ensemble de vecteurs accessibles dans un automate et réciproquement.

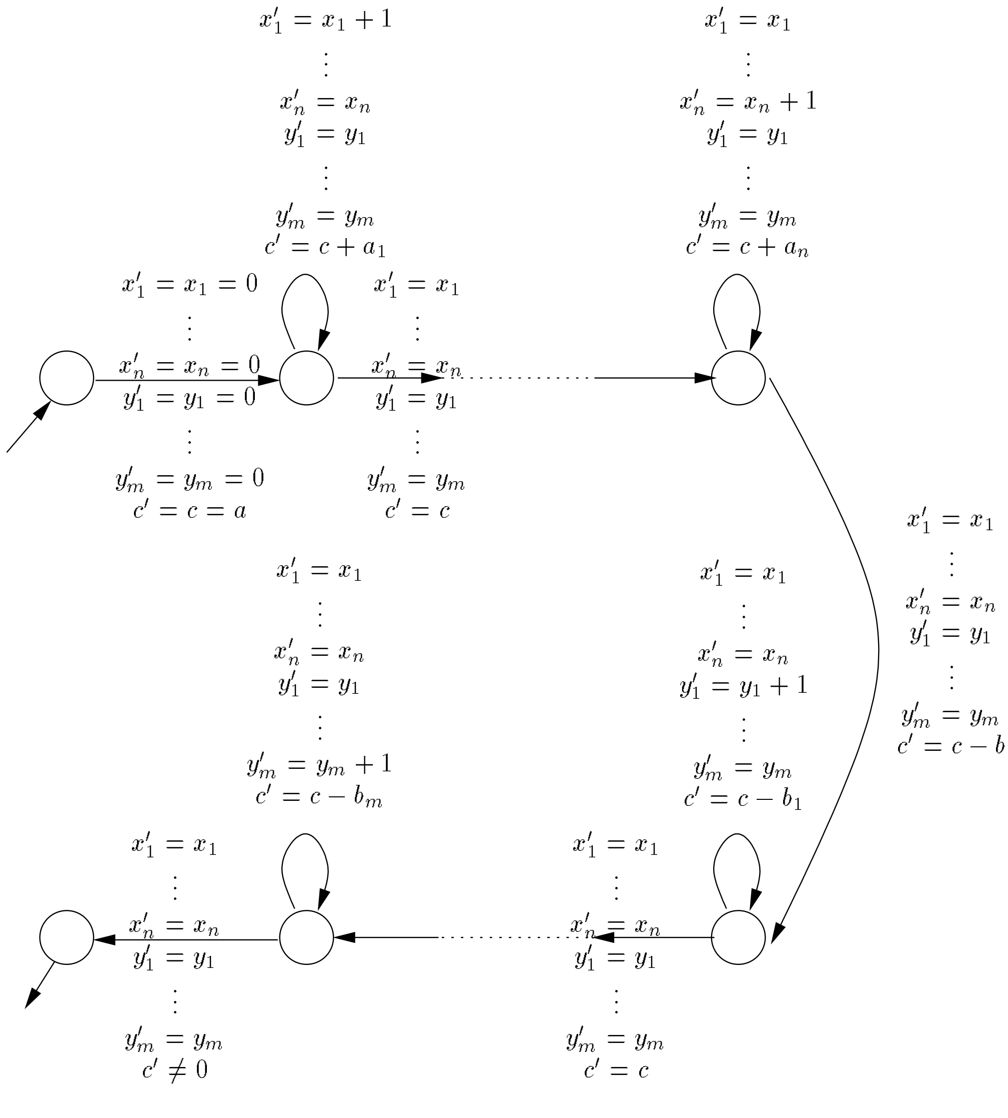


FIG. 2.17 – *Automate reconnaissant les mêmes mots que la formule 2.2 sur ses n+m premières variables*

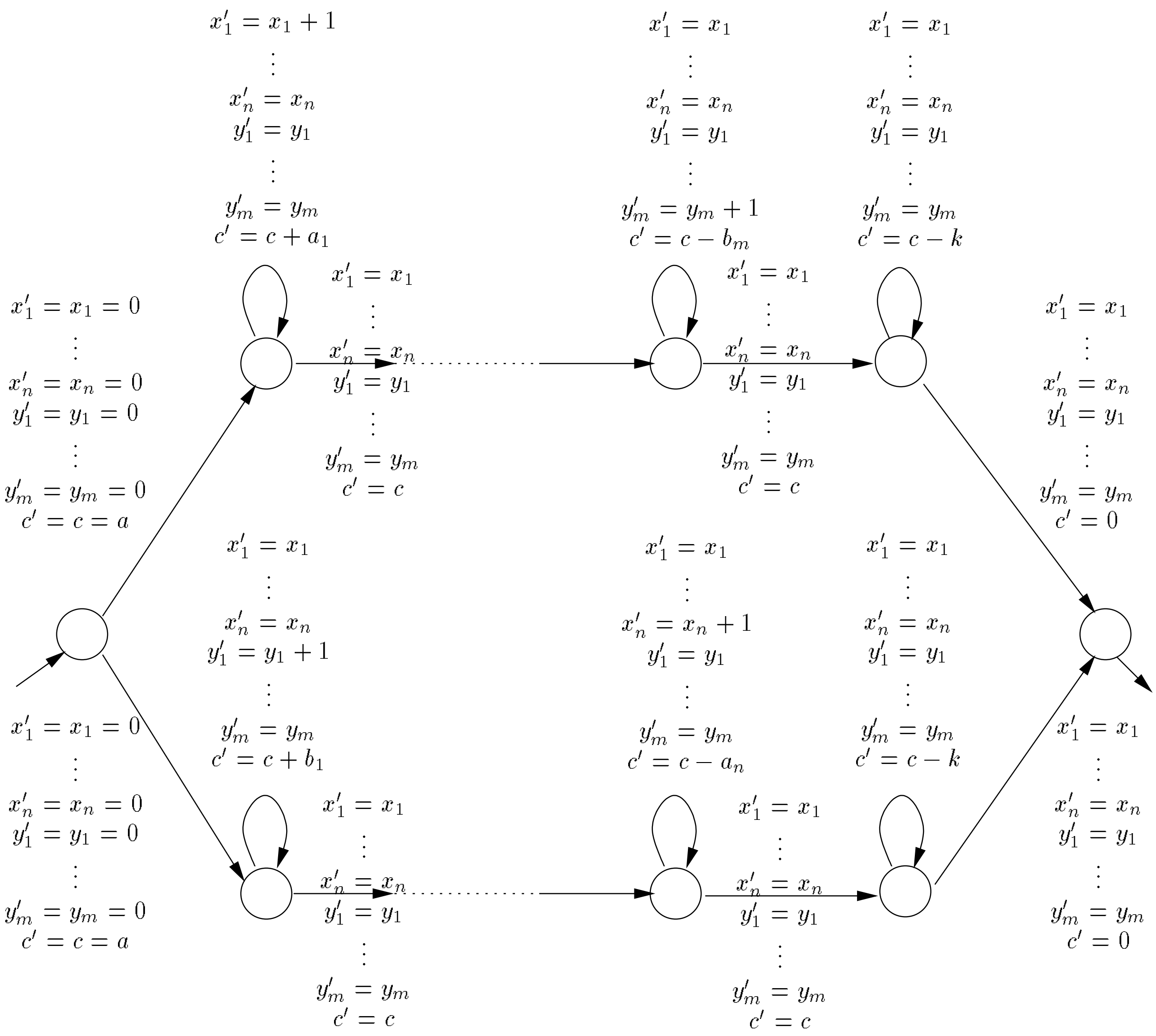


FIG. 2.18 – *Automate reconnaissant les mêmes mots que la formule 2.3 sur ses n+m premières variables*

Conclusion

Ce mémoire s'est articulé autour de la vérification de systèmes à compteurs (donc potentiellement infinis) et plus particulièrement autour de la notion de platitude. En effet, cette notion se retrouve dans beaucoup de problèmes très différents de l'informatique comme nous l'avons vu dans l'introduction de la partie 2.

Dans la première partie, nous avons présenté un système paramétré dont l'accessibilité restreinte, bien que le système soit plat, est indécidable dans le cas général :
étant données $f_1, \ldots, f_p$, p fonctions affines telles que $f_i : \mathbb{N}^k \to \mathbb{N}^k$, $f_i(X) = A_iX + B_i$, étant donnés $U, V \in \mathbb{N}^k$, alors $V \in \{f_p^{n_p} \cdots f_1^{n_1}(U) | n_1, \ldots, n_p \in \mathbb{N}\}$ est indécidable.

Nous avons montré que le problème reste tout de même décidable si on itère une ou deux fonctions uniquement : $V \in \{f^n(U) | n \in \mathbb{N}\}$, $V \in \{g^m f^n(U) | n, m \in \mathbb{N}\}$ est décidable. Mais nous savons qu'il existe une borne K au-delà de $V \in \{f_p^{n_p} \cdots f_1^{n_1}(U) | n_1, \ldots, n_p \in \mathbb{N}\}$ est indécidable ($p \geq K$).

En compléments, nous avons donné quelques conditions suffisantes pour que
$V \in \{f_p^{n_p} \cdots f_1^{n_1}(U) | n_1, \ldots, n_p \in \mathbb{N}\}$ soit décidable. En outre, nous avons présenté un deuxième algorithme qui décide si $V \in \{f^n(U) | n \in \mathbb{N}\}$ et qui permet de montrer que la complexité de ce problème est élémentaire mais la borne obtenue est très élevée : une tour d'exponentielle en k.

Dans la deuxième partie, nous sommes partis une nouvelle fois d'un système plat : les automates plats à compteurs qui présentent la propriété suivante d'après les travaux de H. Comon et Y. Jurski : l'accessibilité est expressible par une formule de Presburger. Nous avons caractérisé la notion de platitude pour un automate à compteurs par une logique temporelle linéaire : nous avons défini une logique $\mathcal{L}_p$ telle que l'ensemble de ses modèles soit exactement l'ensemble des automates plats à compteurs. Il en découle alors que la satisfaisabilité d'une formule de $\mathcal{L}_p$ est décidable et que le model-checking de la négation d'une formule de $\mathcal{L}_p$ sur des automates plats à compteurs est décidable.

De plus, cette logique $\mathcal{L}_p$ s'étend en une logique $\mathcal{L}_p^+$ qui étend LTL. On obtient donc une logique qui contient tout LTL et donc la satisfaisabilité reste décidable. En outre, le model-checking de négation de formules de $\mathcal{L}_p^+$ sur des automates à compteurs plats par morceaux (en particulier les automates plats à compteurs et les automates de mots) est décidable.

En compléments, nous avons trouvé une nouvelle caractérisation de l'arithmétique de Presburger.

L'intérêt principal de cette deuxième partie est d'avoir étendu strictement LTL en une logique temporelle linéaire qui reste décidable et qui permet de prendre en compte des systèmes paramétrés.

De plus, nous avons exposé une construction simple qui permet de passer d'un automate plat à compteur à une formule dont il est le modèle et vice-versa.

Notre travail présente cependant encore un point faible : le model checking a été montré décidable pour la *négation* de formules plates, mais, notre logique n'étant pas close par complémentaire, nous ne savons pas si le model checking est décidable pour des formules arbitraires de notre logique. Cette question revient à décider de l'universalité des automates plats à compteurs, qui est un problème ouvert.

L'universalité des automates temporisés, par exemple, est quant à elle indécidable (voir [2]) et par ailleurs, les automates temporisés peuvent être émulés par des automates plats (voir [7]). Cependant, ce résultat n'entraine pas immédiatement l'indécidabilité de la validité d'une formule de notre logique pour deux raisons : l'indécidabilité de l'universalité des automates temporisés utilise la densité des réels et ne s'applique donc pas à des automates à compteurs entiers. D'autre part, l'émulation ne préserve pas l'universalité.

Nous avons donc montré que notre logique plate est décidable, mais la question importante qui reste à résoudre est celle de la décision de la validité, question qui constitue un prolongement naturel du travail décrit dans ce mémoire.

Bibliographie

[1] Parosh Aziz Abdulla, Ahmed Bouajjani, et Bengt Jonsson, On-the-fly Analysis of Systems with Unbounded, Lossy FIFO Channels. In Proceedings of the 10th Computer Aided Verification (CAV), Vancouver, BC, Canada, 1998. pp. 304-318.

[2] R. Allur and D.L. Dill, A theory of timed automata, pp. 183-235, vol. 126 Theoritical Computer Science, 1994.

[3] M.-P. Béal and D. Perrin, Symbolic Dynamics and Finite Automata, Handbook of Formal Languages, vol 2, pp. 463-506, 1997.

[4] Bernard Boigelot, Symbolic Methods for Exploring Infinite Sate Spaces, Thèse, Université de Liège, 1997-1998.

[5] Ahmed Bouajjani, Peter Habermehl, Symbolic Reachability Analysis of FIFO-Channel Systems with Nonregular Sets of Configurations. In Proceedings of the 24th International Colloquium on Automata, Languages, and Programming (ICALP) , Bologna, Italy, 1997. LNCS, Springer-Verlag.

[6] H. Comon and Y. Jurski, Multiple counters automata, safety analysis and Presburger arithmetic, Research Report LSV-98-1, Mar. 1998.

[7] H. Comon and Y. Jurski, timed automata and the theory of real numbers. Proc. Conf. on Concurrency Theory, Aug. 1999.

[8] Dennis R. Dams, Flat Fragments of CTL anc CTL* : Separating the Expressive and Distinguish Powers, IGPL, Vol. 7 No. 1, pp. 55-78 1999.

[9] L.E. Dickson "Finiteness of the odd perfect and primitive abundant numbers with n distinct prime factors", Amer. J. Math. 35pp. 413-422, 1913.

[10] Kousha Etessami, Thomas Wilke, An Until Hierarchy for temporal logic, pp. 108-117, Proc., 11th IEEE Symp., Logic in Computer Science (LICS'96), New Brunswick NJ, USA, July 1996.

[11] Alain Finkel, Olivier Marcé, Verification of infinite regular communicating automata. Internal report 96-4, LIFAC, École Normale Supérieure de Cachan, April 1996.

[12] A. Finkel, P. McKenzie and C. Picaronny, A Well-Structured Framework for Analysing Petri Nets Extensions, Research Report LSV-99-2, Feb. 1999.

[13] P. Gastin, R. Meyer, A. Petit, "A (non-elementary) modular decision procedure for LTrL", Actes de MFCS'98, Lecture Notes in Computer Science 1450, p. 356-36, 1998.

[14] M. Yu. Matijacevitch Martin David and J. Robinson, Hilbert's Tenth Problem., ch. Diophantine Equations, proc. Symphosia Pure Math., 1976.

[15] M. Minsky, Computation, Finite and Infinite Machine, Prentice Hall, 1967.

[16] Orna Kupferman, Moshe Y. Vardi and Pierre Wolper, An Automata-Theoretic Approach to Branching-Time Model Checking, Proc. Computer Aided Verification, 1994.

[17] J.P. Queille, J. Sifakis. Specification and verification of concurrent systems in Cesar. In Int. Symp. on programming, Proc. 5th Int. Symposium, volume 137 of Lecture Notes in Computer Science, pages 337-351. Springer-Verlag, Berlin, 1981.

[18] A.P. Sistla, E.M. Clarke. The complexity of propositional linear temporal logic. Journal of the Association for Computing Machinery, 32:733-749,1985.

[19] Moshe Y. Vardi, An Automata-Theoretic Approach to Linear Temporal Logic, Logic for concurrency : structure versus automata, vol. 1043, Lecture Notes in Comp. Science, Springer Verlag, 1996.

[20] Ann Yasuhara, Recursive Function Theory & Logic, pp. 285-295, Academic Press, Inc, 1971.