Spécifications modales de réseaux de Petri

Guillaume Feuillade — Sophie Pinchinat

IRISA / INRIA Rennes
Campus Universitaire de Beaulieu
Avenue du Général Leclerc
35042 RENNES Cedex - France
{guillaume.feuillade,sophie.pinchinat}@irisa.fr

RÉSUMÉ. *Nous étendons les résultats de Badouel & Darondeau sur la synthèse de réseaux de Petri non étiquetés à partir d'un langage régulier au cas d'une famille de langages solutions de formules de mu-calcul modal; ces formules sont traduites en spécifications modales, dont une restriction structurelle rend le problème décidable.*

ABSTRACT. *We present an extension of Badouel & Darondeau's results for unlabeled Petri net synthesis from regular languages. We study synthesis from families of languages defined through modal mu-calculus sentence formulas, which translate into modal specifications. A structural restriction makes this problem is decidable.*

MOTS-CLÉS : *réseaux de Petri, mu-calcul, synthèse*

KEYWORDS: *Petri nets, mu-calculus, synthesis*

1. Introduction

Le problème de synthèse de réseaux de Petri est celui de décider de l'existence d'un réseau isomorphe à un graphe fini donné. Ce problème à été résolu en utilisant la théorie des régions [BAD 99], dont le principe est de chercher des régions dans un graphe à partir desquelles on peut déduire les places du réseau synthétisé. Le but de la synthèse est ici de produire, à partir d'un objet séquentiel : le graphe, un objet concurrent : le réseau, dont le graphe des marquages est isomorphe au graphe initial. La synthèse à été étendue dans [BAD 02] à des objets infinis : les graphes automatiques (leurs sommets sont définis par un langage rationnel, et leurs arcs par des transducteurs rationnels). Ce résultat est plus puissant que le précédent car la donnée du problème est une famille de graphes infinis. Le problème de synthèse d'un réseau dont le graphe des marquages est isomorphe à un graphe d'une famille est récent. Nous étudions dans cet article le cas de familles décrites par une formule d'une logique modale du temps arborescent, le nu-calcul conjonctif.

Le mu-calcul modal est une logique du temps arborescent qui permet de spécifier les comportements de systèmes de transitions. Il a été introduit par Kozen [KOZ 83], et porte uniquement sur les comportements (l'arbre des exécutions) d'un système et non sur les états de ce système. Janin et Walukiewicz montrent dans [JAN 96] que les formules du mu-calcul sont exactement les formules de la logique monadique du second ordre qui ne distinguent pas des modèles bissimilaires. Le mu-calcul modal permet la spécification de systèmes et leur vérification (*model-checking*) ; ces deux points ont été abondamment étudiés dans le cadre des systèmes réactifs à évènement discrets (ou processus finis). Le problème du *model-checking* est, étant donné une formule et un système, de déterminer si le système est modèle de la formule ; le problème de synthèse est, étant donné une formule, de construire un modèle de cette formule. Ces deux problèmes sont décidables sur les processus réguliers à l'aide de la théorie des automates d'arbres, comme décrit dans [ARN 01]. De fait, si on considère des réseaux de Petri sans contrainte sur l'étiquetage des transitions, le problème de la synthèse est trivial, puisque tout processus (régulier) peut être décrit par un réseau de ce type. En revanche, le problème devient plus difficile si l'étiquetage des transitions est une fonction injective –on parle alors de réseaux *non étiquetés*–, puisqu'en général il n'est pas possible d'exprimer les comportement de tout processus fini dans ce cadre. L'objet de cet article est précisément d'étudier le problème de la synthèse dans le cas de réseaux non étiquetés.

Dans la première partie on rappelle la syntaxe et la sémantique du mu-calcul, on définit un fragment syntaxique, le nu-calcul conjonctif et une présentation alternative aux formules du nu-calcul conjonctif : les spécifications modales. La deuxième partie est consacrée à la synthèse de réseaux de Petri sur les langages rationnels avec l'approche équationnelle de [BAD 02]. La troisième partie traite de la décidabilité de la synthèse de réseaux pour certaines restrictions structurelles sur les spécifications modales.

2. Mu-calcul et spécifications modales

L'objet de cette partie est la présentation d'un fragment du mu-calcul et l'étude des modèles des formules de ce fragment. Dans un premier temps nous donnons une interprétation du mu-calcul dans les langages, puis nous introduisons le nu-calcul conjonctif en tant que fragment syntaxique. Nous proposons ensuite un autre façon présenter l'ensemble des modèles d'une formule du nu-calcul conjonctif : les spécifications modales. Nous étudions alors la structure des modèles d'une spécification modale. Nous établissons enfin l'équivalence entre ces spécifications et les formules du nu-calcul-conjonctif en montrant qu'elles définissent les mêmes ensembles de modèles.

2.1. *Notions utiles sur les langages*

On se fixe un alphabet fini $\Sigma = \{a_1, \dots a_n\}$. On considère les langages sur Σ, d'éléments typiques $L, R, ...$ avec les notations habituelles : lorsque u et v sont deux éléments de Σ^*, $u.v$ désigne la concaténation de u et v, et $u^* = \{u^k \mid k \in \mathbb{N}\}$, où u^k est la concaténation de k fois le mot u ; L^*, $L.a$ (pour $a \in \Sigma$), etc. $1 \in \Sigma^*$ désigne le mot vide.

Définition 1 (Clôture par préfixe) *Soit L un langage, on dit que L est clos par préfixe si et seulement si $1 \in L$ et tout mot $a_1 \dots a_n \in L$ vérifie $a_1 \dots a_{n-1} \in L$. La* clôture par préfixe *d'un langage L est le plus petit langage clos par préfixe contenant L, on la note $\overline{L}$. On notera également L_w l'ensemble $\{v \in \Sigma^* \mid w.v \in L\}$ des suffixes de w dans L.*

Remarquons que le langage vide n'est par définition pas clos par préfixe, nous le traiterons donc séparément lorsque c'est nécessaire ; en particulier, pour un langage clos par préfixe L, pour tout mot w, le langage L_w est soit clos par préfixe (si $w \in L$) soit vide. Par la suite L dénotera toujours un langage clos par préfixe sur l'alphabet Σ.

2.2. *Mu-calcul modal pour les langages*

Nous donnons les définitions relatives au mu-calcul et nous le dotons d'une interprétation sur les langages. Soit $Var = \{X, X_1, X_2, \dots\}$.

2.2.1. *Syntaxe du Mu-calcul*

Définition 2 (Syntaxe du Mu-calcul) *L'ensemble des formules du Mu-calcul modal, noté L_μ est défini par la grammaire suivante :*

$$(L_\mu \ni) \quad \beta_1, \beta_2 ::= \texttt{true} \mid X \mid \ <a> \beta_1 \mid \neg\beta_1 \mid \beta_1 \vee \beta_2 \mid \mu X.\beta_1(X)$$

où $a \in \Sigma$ et où toute variable X est sous la portée d'un nombre pair de symboles de négations $\neg$ dans $\beta_1(X)$ pour toute formule $\mu X.\beta_1(X)$ (pour assurer l'existence de points fixes).

On note classiquement par $\texttt{false}, [a]\beta_1, \beta_1 \wedge \beta_2, \rightarrow^a, \not\rightarrow^a$ *et* $\nu X.\beta_1(X)$ *les formules respectives* $\neg\texttt{true}, \neg\ <a>\ (\neg\beta_1), \neg(\neg\beta_1 \vee \neg\beta_2), <a>\ \texttt{true}, [a]\texttt{false}$ *et* $\neg\mu X.\neg\beta_1(\neg X)$.

On dit que la variable X *est* libre dans β *si elle n'est sous la portée d'aucun opérateur* $\mu.X$ *ou* $\nu.X$. *L'ensemble des variable libres de* β *est noté* $var(\beta)$. *Une formule* β *sans variable libre est appelée* sentence.

2.2.2. *Sémantique du Mu-calcul sur les langages*

Nous définissons une interprétation des formules du Mu-calcul modal sur les langages clos par préfixe de Σ^*. L'interprétation d'une formule du Mu-calcul sur un langage clos par préfixe L est l'ensemble des mots de L qui satisfont la formule, relativement à une interprétation donnée val des variables libres de la formule ; cet ensemble n'est pas forcément lui-même clos par préfixe.

Définition 3 (Sémantique du Mu-calcul) *Une formule* $\beta \in L_\mu$ *est interprétée sur un langage* $L \subseteq \Sigma^*$ *clos par préfixe, relativement à une valuation* $val : Var \rightarrow L$, *par l'ensemble* $[\![\ \alpha\ ]\!]_L^{[val]} \subseteq L$ *défini inductivement par :*

$$
\begin{array}{l}
[\![\ \texttt{true}\ ]\!]_L^{[val]} = L \\
[\![\ X\ ]\!]_L^{[val]} = val(X) \\
[\![\ \neg\alpha\ ]\!]_L^{[val]} = L \setminus [\![\ \alpha\ ]\!]_L^{[val]} \\
[\![\ \beta_1 \vee \beta_2\ ]\!]_L^{[val]} = [\![\ \beta_1\ ]\!]_L^{[val]} \cup [\![\ \beta_2\ ]\!]_L^{[val]} \\
[\![\ <a>\ \beta_1\ ]\!]_L^{[val]} = \{w \in L \mid w.a \in [\![\ \beta_1\ ]\!]_L^{[val]}\} \\
[\![\ \mu X.\beta_1(X)\ ]\!]_L^{[val]} = \bigcap\{V \subseteq L \ \mid\ [\![\ \beta_1\ ]\!]_L^{[val(V/X)]} \subseteq V\}
\end{array}
$$

où $val(V/X) : Var \rightarrow \mathcal{P}(L)$ *est la valuation définie par* $val(V/X)(X') = V(X')$ *pour toute variable* $X' \in Var$ *différente de* X *et* $val(V/X)(X) = V$.

L'interprétation $[\![\ \mu X.\beta(X)\ ]\!]_L^{[val]}$ (respectivement $[\![\ \nu X.\beta(X)\ ]\!]_L^{[val]}$) est ainsi le plus petit point fixe (resp. plus grand point fixe) de la fonction $V \mapsto [\![\ \beta\ ]\!]_L^{[val(V/X)]}$. La sémantique des sentences du Mu-calcul étant indépendante de la valuation, on notera simplement $[\![\beta]\!]_L$ l'interprétation de la sentence β relativement à n'importe quelle valuation. On dit que "le langage L *satisfait* la sentence β", noté $L \models \beta$, si et seulement si $1 \in [\![\ \beta\ ]\!]_L$.

2.3. *Nu-calcul conjonctif*

Nous exhibons un fragment syntaxique de L_μ pour lequel nous donnons une présentation alternative sous forme de langages, plus adaptée à la synthèse de réseaux.

2.3.1. *Syntaxe et sémantique*

Définition 4 (Nu-calcul conjonctif) *L'ensemble des formules du Nu-calcul conjonctif, noté L_ν est un fragment syntaxique de L_μ défini par restriction de la grammaire de L_μ (où $a \in \Sigma$) :*

$$(L_\nu \ni) \quad \beta_1, \beta_2 ::= \texttt{true} \mid X \mid \ \rightarrow^a \ \mid [a]\beta_1 \mid \ \not\rightarrow^a \ \mid \beta_1 \wedge \beta_2 \mid \nu X.\beta_1(X)$$

Une formule $\beta \in L_\nu$ est interprétée sur un langage $L \subseteq \Sigma^*$ clos par préfixe, relativement à une valuation $val : Var \rightarrow L$, suivant la même sémantique que L_μ :

$$\begin{array}{l}
[\![\texttt{true}]\!]_L^{[val]} = L \\
[\![X]\!]_L^{[val]} = val(X) \\
[\![\rightarrow^a \beta]\!]_L^{[val]} = \{w \in L \mid w.a \in L\} \\
[\![\not\rightarrow^a]\!]_L^{[val]} = \{w \in L \mid w.a \notin L\} \\
[\![[a]\beta]\!]_L^{[val]} = \{w \in L \mid w.a \in [\![\beta]\!]_L^{[val]}\} \cup \{w \in L \mid w.a \notin L\} \\
[\![\beta_1 \wedge \beta_2]\!]_L^{[val]} = [\![\beta_1]\!]_L^{[val]} \cap [\![\beta_2]\!]_L^{[val]} \\
[\![\nu X.\beta(X)]\!]_L^{[val]} = \bigcup\{V \subseteq L \ \mid \ [\![\beta]\!]_L^{[val(V/X)]} \supseteq V\}
\end{array}$$

où $val(V/X) : Var \rightarrow \mathcal{P}(L)$ est la valuation définie par $val(V/X)(X') = V(X')$ pour toute variable $X' \in Var$ différente de X et $val(V/X)(X) = V$.

L'opérateur $<a>\ \beta$ de L_μ s'exprime alors $[a]\beta\wedge\ \rightarrow^a$ dans L_ν. Les opérateurs suivants ne peuvent être exprimés dans L_ν : $\beta_1 \vee \beta_2$, $\mu X.\beta(X)$ et `false`. La disjonction $\vee$ est alors seulement présente implicitement dans le cas particulier $[a]\beta$, que l'on pourrait écrire $<a>\ \beta\vee\ \not\rightarrow^a$ dans L_μ.

Soit E_β l'ensemble des langages qui satisfont la formule β de L_ν. Afin de caractériser E_β, on se munit dans la suite d'une description de cet ensemble sous la forme de spécifications modales. On établit dans un premier temps la structure des modèles d'une spécification modale puis on montre que les spécifications modale et L_ν ont la même expressivité.

2.3.2. *Spécifications modales*

Définition 5 (Spécification modale)
*Une spécification modale est un tuple $S = \langle \{C_a\}_{a\in\Sigma}, I\rangle$ où pour tout $a \in \Sigma$, C_a est un langage rationnel des mots qui doivent être suivis de l'action a et I est un langage rationnel d'actions interdites. On définit $C_S : \mathcal{P}(\Sigma^*) \rightarrow \mathcal{P}(\Sigma^*)$, l'*opérateur de complétion *associé à la spécification modale S par $C_S(L) = \bigcup_{a\in\Sigma}(L \cap C_a).a$.*

On définit

$$mod(S) = \{L \subseteq \Sigma^* \mid C_S(L) \subseteq L \wedge L \cap I = \emptyset\}$$

On dit que "S est satisfiable" si $mod(S) \neq \emptyset$, et que "L satisfait S" si $L \in mod(S)$.

Les langages satisfaisant une spécification modale ne sont pas tous nécessairement rationnels.

Exemple 1 *Soit* $S = \langle \{C_a\}_{a\in\Sigma}, I\rangle$ *sur l'alphabet* $\Sigma = \{a, b, c\}$ *avec* $C_a = (1 + b.c(a.c.b^*)^*)$, $C_b = \{1\}$, $C_c = \emptyset$ *et* $I = (c + a.a + b.(c.b^*.a)^*.b)$.

– *le langage* $L_1 = \overline{((b.c.a)^*)}$ *ne satisfait pas* S *car* $a \in C_S(L_1)$ *mais* $a \notin L_1$;
– *le langage* $L_2 = \overline{((b.c.a)^* + a)}$ *satisfait* S,
– *et le langage* $L_3 = \overline{\{(b.c.b^n.a.c.b^n.a \mid n \in \mathbb{N}\} \cup \{a\}}$ *satisfait* S.

Soit $S = \langle \{C_a\}_{a\in\Sigma}, I\rangle$ une spécification modale. Sans perdre en généralité, on suppose que si S est satisfiable, alors $I \cap C_S(\Sigma^*) = \emptyset$ (si ce n'est pas le cas, alors la spécification $S' = \langle \{C'_a\}_{a\in\Sigma}, I\rangle$ où pour tout $a \in \Sigma$, $C'_a = C_a \cap \{w \in \Sigma^* \mid w.a \in I\}$ vérifie $mod(S') = mod(S)$ et $I \cap C_S(\Sigma^*) = \emptyset$).

Nous donnons quelques définitions relatives à S.

Définition 6 (S-clôture) *La S-clôture d'un langage L, notée $L\uparrow^S$, est le plus petit langage $L' \supseteq L$ tel que $L' \in mod(S)$. On note $L^S_\bot = \{1\}\uparrow^S$ et $L^S_\top$ le langage $\Sigma^* \setminus I.\Sigma^*$.*

Lemma 1 *La S-clôture d'un langage L est la plus petite solution L' de l'équation $R = L \cup C_S(R)$.*

Lemma 2 *Ces quatre propositions sont équivalentes :*
1. S est satisfiable *2. $L^S_\bot \in mod(S)$* *3. $L^S_\bot \cap I = \emptyset$* *4. $L^S_\top \in mod(S)$*

Preuve On ne montre que $3 \Rightarrow 2$, $1 \Rightarrow 4$ et $4 \Rightarrow 3$ (les cas $2 \Rightarrow 1$, $4 \Rightarrow 1$ et $2 \Rightarrow 3$ sont immédiats et permettent de conclure). $3 \Rightarrow 2$: par le lemme 1 nous avons $L^S_\bot = \{1\} \cup C_S(L^S_\bot)$, d'où $C_S(L^S_\bot) \subseteq L^S_\bot$ et par hypothèse $L^S_\bot \cap I = \emptyset$, donc $L \in mod(S)$; $1 \Rightarrow 4$: $C_S(L^S_\top) \subseteq C_S(\Sigma^*)$, or $C_S(\Sigma^*) \cap I = \emptyset$, par conséquent $C_S(L^S_\top) \subseteq L^S_\top$ et $L^S_\top \cap I = \emptyset$, d'où $L^S_\top \in mod(S)$; $4 \Rightarrow 3$: $L^S_\top \cup C_S(L^S_\top) = L^S_\top$ donc $L^S_\bot \subseteq \{1\} \cup C_S(L^S_\top) \subseteq L^S_\top$ et $L^S_\top \cap I = \emptyset$, d'où $L^S_\bot \cap I = \emptyset$. □

Proposition 1 *Lorsque S est satisfiable, alors il existe un modèle rationnel de S, en particulier $L^S_\top$ et $L^S_\bot$ sont rationnels.*

Preuve Le langage $L_\top$ est rationnel par définition.
On montre que la S-clotûre d'un langage rationnel est rationnelle. En effet, pour calculer $L\uparrow^S$, nous pouvons procéder de la manière suivante : (1) construire l'automate $\mathcal{A}$ reconnaissant $L \cup \bigcup_{a\in\Sigma} C_a.a$, (2) retirer de $\mathcal{A}$ tous les états non terminaux –cette opération produit un nouvel automate $\mathcal{A}'$ qui reconnaît le plus grand sous-langage clos par préfixe de $L \cup \bigcup_{a\in\Sigma} C_a.a$–, et (3) poser $L\uparrow^S = \mathcal{L}(\mathcal{A})$. □

Théorème 1 *Si S est satisfiable alors $(mod(S), \subseteq)$ est un treillis complet et distributif de plus grand élément $L^S_\top$ et de plus petit élément $L^S_\bot$.*

Preuve Tout d'abord, on peut facilement montrer que toute partie d'une specification modale admet une borne supérieure et une borne inférieure dans $mod(S)$. Si S est satisfiable, alors d'après le lemme 2, $L^S_\top \in mod(S)$, or $L^S_\top = \Sigma^* \setminus I.\Sigma^*$, donc $L^S_\top = max(mod(S))$. De plus $L^S_\bot \in mod(S)$, or d'après la définition 6, $L^S_\bot = min(mod(S))$. La distributivité est celle de $\cap$ sur $\cup$. □

2.3.3. *Opérations sur les spécifications modales*

Nous nous dotons maintenant d'opérations de construction et de combinaison de spécifications modales, puis nous donnons une décomposition des spécifications modales à l'aide d'un ensemble de spécifications atomiques.

Définition 7 (Opérateurs)
L'union *de deux spécifications $S_1 = \langle \{C^1_a\}_{a\in\Sigma}, I^1\rangle$ et $S_2 = \langle \{C^2_a\}_{a\in\Sigma}, I^2\rangle$, est la spécification $S_1 \cup S_2 = \langle \{C^1_a \cup C^2_a\}_{a\in\Sigma}, I^1 \cup I_2\}$.*

Le préfixage *d'une spécification S_1 par un langage $R \subseteq \Sigma^*$ est la spécification $R.S = \langle \{R.C_a\}_{a\in\Sigma}, R.I\rangle$.*

L'union de deux spécifications est une spécification dont les modèles sont l'intersection de modèles des deux spécifications, elle correspond au 'et' logique.

Lemma 3 *Soient S_1 et S_2 deux spécifications modales, $S_1 \cup S_2$ vérifie $mod(S_1 \cup S_2) = mod(S_1) \cap mod(S_2)$.*

Lemma 4 *Pour tout $L \subseteq \Sigma^*$,*

$$L \in mod(R.S) \Leftrightarrow \forall w \in R, L_w = \emptyset \text{ ou } L_w \in mod(S)$$

où L_w est l'ensemble des suffixes de w dans L (voir Définition 1).

Preuve Pour prouver ce lemme on utilise sans le préciser le fait que $L \cap v.\Sigma^* = v.L_v$ et donc $L \cap v.L' = v.(L_v \cap L')$.
$\Rightarrow$) Soit $L \in mod(R.S)$, par construction de $R.S$:

$$C_{R.S}(L) = \bigcup_{a\in\Sigma} (L \cap R.C_a) = \bigcup_{a\in\Sigma} \bigcup_{w\in R} (L \cap w.C_a) = \bigcup_{a\in\Sigma} \bigcup_{w\in R} w.(L_w \cap C_a)$$

Puisque $C_{R.S}(L) \subseteq L$, pour tout $w \in R$ et pour tout $a \in \Sigma^*$, nous avons $w.(L_w \cap C_a).a \subseteq L$, d'où $(L_w \cap C_a).a \subseteq L_w$. De la même manière, puisque $L \cap R.I = \emptyset$, $L \cap w.I = \emptyset$ pour tout $w \in R$, c.à-d. $L_w \cap I = \emptyset$. On en déduit que $L_w = \emptyset$ ou $L_w \in mod(S)$.

$\Leftarrow$) Montrons que pour tout $a \in \Sigma$, $(L \cap R.C_a).a \subseteq L$. Soit $v \in L \cap R.C_a$, il existe donc $w \in R$ et $u \in C_a$ tels que $v = wu$. Ainsi $u \in L_w \cap C_a$. et $L_w \neq \emptyset$. Par hypothèse, $L_w \in mod(S)$, donc $(L_w \cap C_a).a \subseteq L_w$, et en particulier $u.a \in L_w$. On en déduit que $v.a = w.u.a \in L$. Montrons maintenant que $L \cap R.I = \emptyset$: pour tout $w \in R$, si $L_w = \emptyset$, alors $L_w \cap RI = \emptyset$ donc $L \cap R.I = \emptyset$, sinon $L_w \in mod(S)$ mais alors $L_w \cap I = \emptyset$, d'où $L \cap w.I = \emptyset$. □

Définition 8 *On définit les spécifications atomiques suivantes :*
$S_{\texttt{true}} = \langle \{\emptyset\}_{a \in \Sigma^*}, \emptyset \rangle$, $S_{\not\rightarrow^b} = \langle \{\emptyset\}_{a \in \Sigma}, \{b\} \rangle$, *et*
$S_{\rightarrow^b} = \langle \{C_a\}_{a \in \Sigma}, \emptyset \rangle$, *avec* $C_a = \emptyset$ *pour* $a \neq b$ *et* $C_b = \{1\}$.

Nous avons alors :

$$\begin{array}{lcl} mod(S_{\texttt{true}}) & = & \mathcal{P}(\Sigma^*) \\ mod(S_{\not\rightarrow_b}) & = & \{L \subseteq \Sigma^* \mid b \notin L\} \\ mod(S_{\rightarrow_b}) & = & \{L \subseteq \Sigma^* \mid b \in L\} \end{array}$$

Théorème 2 *Toute spécification modale peut être exprimée à l'aide des spécifications atomiques et des opérateurs d'union et de préfixage.*

Preuve Soit $S = \langle \{C_a\}_{a \in \Sigma}, I \rangle$ une spécification modale, pour tout a dans Σ nous définissons l'ensemble $I^a = \{u \in \Sigma^* \text{ tel que } u.a \in I\}$. Soit $S' = \langle \{C'_a\}_{a \in \Sigma}, I' \rangle$ la spécification définie par

$$S' = \bigcup_{a \in \Sigma} C_a.S_{\rightarrow^a} \cup \bigcup_{a \in \Sigma} I^a.S_{\not\rightarrow^a}$$

D'après les définitions 7 et 8, nous avons $S = S'$. □

2.3.4. *Équivalence entre les spécifications modales et le Nu-calcul conjonctif*

Nous énonçons le théorème suivant :

Théorème 3 *Pour tout ensemble de langages clos par préfixe E, il existe une spécification modale S telle que $mod(S) = E$ si et seulement si E est l'ensemble des modèles d'une formule de L_ν.*

Par soucis de concision, nous ne reportons pas ici la preuve de ce théorème. Cependant, nous donnons la transformation qui permet de traduire une formule L_ν en une spécification modale équivalente.
On introduit dans un premier temps la notion de chemins de variable :

Définition 9 (Chemin de variable) *Soit β un formule de L_ν, on définit une fonction $P_\beta : var(\beta) \rightarrow \mathcal{P}(\Sigma^*)$, par induction sur β : pour tout $X \in var(\beta)$,*

– $\beta \in \{\texttt{true}, \rightarrow^a, \not\rightarrow^a\}$, *alors* $P_\beta(X) = \emptyset$,
– $\beta = Y$ *et* $Y \neq X$, *alors* $P_\beta(X) = \emptyset$,
– $\beta = X$, *alors* $P_\beta(X) = \{1\}$,
– $\beta = [a]\alpha$, *alors* $P_\beta(X) = a.P_\alpha(X)$,
– $\beta = \beta_1 \wedge \beta_2$, *alors* $P_\beta(X) = P_{\beta_1}(X) \cup P_{\beta_2}(X)$,
– $\beta = \nu Y.\alpha(Y)$, *alors* $P_\beta(X) = P_\alpha(Y)^*.P_\alpha(X)$.

Le langage $P_\beta(X)$ *est alors l'ensemble des* chemins de la variable X *dans* β.

Exemple 2 *Quelques exemples de chemins de variables :*

– *si* $\beta = [a]X$, *alors* $P_\beta(X) = \{a\}$,
– *si* $\beta = [a][b]X \wedge [c]X$, *alors* $P_\beta(X) = (a.b + c)$,
– *si* $\beta = \nu Y.([a][b]Y \wedge [c]X)$, *alors* $P_\beta(X) = (a.b)^*.c$

Les chemins de variables correspondent aux mots qui "amènent" à la variable X *: lorsque* $w \in [\![\ \beta\ ]\!]_L^{[val]}$, $P_\beta(X)$ *indique les mots* v *tels que* $w.v \in [\![\ X\ ]\!]_L^{[val]}$, *ou de manière équivalente* $w.v \in val(X) \subseteq L$.

Nous pouvons maintenant définir la spécification modale associée à une formule $\beta \in L_\nu$:

Définition 10 (Spécification modale associée à une formule) *On définit* S_β *la spécification modale associée à la formule* $\beta \in L_\nu$ *inductivement sur* β *:*

– $\beta \in \{\texttt{true}, \rightarrow^a, \not\rightarrow^a\}$, S_β *est donnée par la Définition 8,*
– $\beta = X$, $S_\beta = S_{\texttt{true}}$,
– $\beta = [a]\alpha$, $S_\beta = a.S_\alpha$,
– $\beta = \beta_1 \wedge \beta_2$, $S_\beta = S_{\beta_1} \cup S_{\beta_2}$,
– $\beta = \nu Y.\alpha(Y)$, $S_\beta = P_\alpha(Y)^*.S_\alpha$.

Exemple 3 *Soit* $\beta = [a]\nu X.([b]X \wedge \rightarrow^a \wedge \not\rightarrow^c)$, *la spécification modale associée à* β *est* $(a.b^*).(S_{\rightarrow^a} \cap S_{\not\rightarrow^c})$, *il s'agit de la spécification* $S_\beta = \langle \{C_a\}_{a\in\Sigma}, I\rangle$ *avec :*

$$C_a = (a.b^*), C_b = \emptyset, C_c = \emptyset, I = (a.b^*)$$

La proposition suivante établit que l'on peut sans perdre de généralité étudier le problème de la synthèse d'un réseau de Petri non étiqueté pour le cas des familles décrites par des specifications modales.

Proposition 2 *Pour tout* L, *pour tout* $w \in L$, *et pour toute sentence* β *de* L_ν,

$$L \models \beta \text{ si et seulement si } L \in mod(S_\beta)$$

3. Réseaux de Petri et synthèse

3.1. *Réseaux de Petri*

Définition 11 (Réseau de Petri)
Un Réseau de Petri *est un triplet $N = (P, T, F)$ où P et T sont deux ensembles finis disjoints respectivement de* places *et de* transitions *et F est une fonction, $F : (P \times T) \cup (T \times P) \to \mathbb{N}$. Un* marquage *de N est une application $m : P \to \mathbb{N}$. Une transition t est dite* active *au marquage m si et seulement si pour toute place $p \in P$, $m(p) \geq F(p,t)$; dans ce cas, le* tir *de t transforme le marquage m en un marquage m' de N tel que pour toute place $p \in P$, $m'(p) = m(p) - F(p,t) + F(t,p)$ et on note $m[t\rangle m'$. On notera également $m[w\rangle m'$ pour une séquence de transitions $w \in T^*$ lorsque m' est le produit des tirs successifs des transitions de w à partir de m.*

Un réseau de Petri $N = (P, T, F)$ augmenté d'un marquage initial m_0 sera noté $N = (P, T, F, m_0)$. Par la suite tous les réseaux que l'on considère seront dotés d'un marquage initial.

Définition 12 (Langage d'un réseau de Petri)
Soit N un réseau et m_0 un marquage de N, le langage *de N à partir de m_0, noté $\mathcal{L}(N, m_0)$ (ou $\mathcal{L}(N)$ lorsque N est initialisé) est l'ensemble des mots w de T^* tels que la succession de transitions w puisse être tirée à partir de m_0, c.à-d. il existe un marquage m tel que $m_0[w\rangle m$.*

3.2. *Synthèse de réseaux de Petri : langages rationnels et approche équationnelle*

Nous présentons ici de manière succincte la méthode de synthèse de réseaux de Petri proposée par Badouel et Darondeau dans [DAR 98] pour les spécifications automatiques. La spécification pour la synthèse est ici un langage rationnel, ce qui est un cas particulier de spécification automatique. Pour plus détail nous recommandons au lecteur de se reporter à [DAR 98]
On se donne un alphabet fini $\Sigma = \{a_1, \ldots, a_n\}$ qui est fixé pour la suite de cet article. Les réseaux de Petri que l'on utilise ont alors tous Σ comme ensemble de transitions (on confond les transitions et leurs images par un étiquetage injectif sur Σ).

Définition 13 (Places implicites d'un langage) *Soit L un langage rationnel, une* place implicite *de L est un réseau de Petri $p = (\{p\}, \Sigma, F, m_0)$ possédant une unique place p et telle qu'il existe une application $\sigma : L \to \mathbb{N}$ des mots de L vers les valuations de p, vérifiant $\sigma(1) = m_0(p)$ et pour tout mot $u.a \in L$, $\sigma(u)[a\rangle\sigma(u.a)$. L'application σ est unique et nommée l'*application associée *à p.*

Informellement, les places implicites d'un langage L sont les places qui permettent le tir de tous les mots du langage. Elles sont alors les seules places possibles pour tout

réseau dont le langage contient L.
Soit $N = (P, \Sigma, F, m_0)$ un réseau de Petri doté d'un marquage initial m_0. On identifie toute place $p \in P$ avec avec le vecteur de taille $(2n + 1)$

$$p = \langle m_0(p), F(p, a_1), \dots, F(p, a_n), F(a_1, p), \dots F(a_n, p)\rangle$$

Dans la suite nous fixons un langage L. Sous cette représentation, un vecteur $x = \langle x_0, x_1, \dots, x_n, x_{n+1}, \dots x_{2n}\rangle$ désigne une place implicite de L si et seulement si toutes ses composantes sont des entiers naturels et il existe $\sigma : L \to \mathbb{N}$ tel que

$$\sigma(1) = x_0 \text{ et } \forall u.a_i \in L, \sigma(u) \geq x_i \text{ et } \sigma(u.a_i) = \sigma(u) - x_i + x_{(n+i)} \quad [1]$$

Pour tout mot $w \in L$, on note $|w|_i$ le nombre d'occurrences de la lettre a_i dans w, l'application σ se définit alors par :

$$\sigma(w) = x_0 + \sum_{i=1}^{n} |w|_i \times (x_{(n+i)} - x_i) \quad [2]$$

En combinant [1] et [2] pour éliminer σ, nous obtenons alors un ensemble infini d'équations dont les solutions dans $\mathbb{N}^{2n+1}$ sont les places implicites de L. Nous donnons maintenant la méthode pour obtenir un système fini équivalent.

Définition 14 *L'*image commutative *d'un mot $w \in \Sigma^*$ est le vecteur $\psi(w)$ de dimension n tel que pour tout $1 \leq i \leq n$, $\psi(w)[i] = |w|_i$.*

On note $\Psi(R)$ l'ensemble $\bigcup_{w \in R} \psi(w)$ pour tout langage rationnel R. On partitionne L tel que $L = \{1\} \cup \bigcup_{j=1}^{n} (L^j).a_j$ et par conséquent $\Psi(L) = \bigcup_{j=1}^{n} (\Psi(L^j)$. Par construction $\Psi(L^j)$ est un sous-ensemble rationnel de $\mathbb{N}^n$. La méthode utilise alors le fait que les ensembles rationnels coïncident avec les ensembles semi-linéaires. Ceci ce traduit par une expression de $\Psi(L^j)$ sous la forme d'un union finie de sous-ensembles linéaires de $\mathbb{N}^n$: $\Psi(L^j) = \bigcup_{k=1}^{K} y_k + (Z_k)^*$ où $y_k \in \mathbb{N}^n$, Z_k est un sous ensemble fini de $\mathbb{N}^n$ et $(Z_k)^*$ désigne l'ensemble des combinaisons linéaires de vecteurs de Z_k avec coefficients dans $\mathbb{N}$. Nous pouvons maintenant écrire le système (1) sous la forme de l'ensemble fini d'inéquations suivantes, pour tout $1 \leq j \leq n$, $1 \leq k \leq K$, et $z_k \in Z_k$:

$$\sum_{i=1}^{n} y_k[i] \times (x_{n+i} - x_i) \geq x_j - x_0 \text{ et } \sum_{i=1}^{n} z_k[i] \times (x_{n+i} - x_i) \geq 0$$

On nomme $\mathcal{E}$ le système d'équations obtenu. Les solutions de $\mathcal{E}$ forment un cône rationnel possédant un nombre fini de rayons extrémaux. On nomme $\mathcal{C}(L)$ le cône ainsi obtenu à partir du langage L. L'ensemble $\{q_1, \dots q_m\}$ des rayons extrémaux de $\mathcal{C}(L)$ peut être calculé en utilisant l'algorithme de Chernikova. Les solutions de $\mathcal{E}$ sont alors toutes les combinaisons linéaires de $\{q_1, \dots q_m\}$ à coefficients dans $\mathbb{Q}$.
Les solutions de $\mathcal{E}$ possédant leurs composantes dans $\mathbb{N}$ (ou de manière équivalente,

toute solution de $\mathcal{E}$ dont les composantes sont ramenées sur $\mathbb{N}$ par multiplication du vecteur par un entier) définissent alors les places implicites de L.
Le réseau composé de l'ensemble des rayons extrémaux de $\mathcal{C}(L)$ (dont les coefficients sont ramenés dans $\mathbb{N}$) est noté $PN(L)$, son langage est noté $\mathcal{L}PN(L)$.

Théorème 4 *[DAR 98] Le réseau $PN(L)$ est minimal au sens où, pour tout réseau de Petri N,*

$$L \subseteq \mathcal{L}(N) \Rightarrow \mathcal{L}PN(L) \subseteq \mathcal{L}(N)$$

Il existe alors une solution à la synthèse de réseau pour un langage rationnel L si et seulement si $\mathcal{L}PN(L) = L$. Dans le cas contraire $PN(L)$ est le réseau de Petri possédant le plus petit langage contenant L.
Par la suite nous étendons la méthode de synthèse à certaines classes de spécifications modales. Nous utiliserons pour cela, parmi les notions évoquées ici, celles de place implicite ainsi que $PN(L)$ et $\mathcal{L}PN(L)$ pour un langage L.

4. Synthèse à partir de spécifications modales

Nous présentons ici une extension inédite à la synthèse de Réseaux à partir d'une famille de langages. La famille de langages est donnée comme l'ensemble des modèles d'une spécification modale. La problématique diffère alors des problématiques de synthèse de [BAD 99, DAR 98, BAD 02] par le fait que l'ensemble des états du système à synthétiser n'est pas connu à priori.

Définition 15 Les langages Σ_0 et Σ_1 *On définit deux sous ensemble de Σ^* : Σ_0 désigne les langages finis de mot de Σ^*, et par conséquent les langages qui peuvent être définis par une expression régulière sans employer l'étoile ; Σ_1 désigne les langages de mots de Σ^* qui sont définis par une union finie de langages du type $(u.v^*.w)$ où u, v, w sont des mots de Σ^*. On dit d'une spécification $S = \langle \{C_a\}_{a\in\Sigma}, I\rangle$ qu'elle est Σ_0 (resp. Σ_1, $(\Sigma_1 \cup \Sigma_0)$) si ses composantes C_a pour tout $a \in \Sigma$ et I sont dans Σ_0 (resp. Σ_1, $(\Sigma_1 \cup \Sigma_0)$). Lorsque $L \in \Sigma_1$, on nomme* taille *d'un langage le nombre de langages du type $(u.v^*.w)$ qu'il contient. On nomme également* taille *d'une spécification Σ_1, la somme des tailles de ses composantes.*

On se donne $S = \langle \{C_a\}_{a\in\Sigma}, I\rangle$, avec l'hypothèse que S est satisfiable.

Définition 16 Σ_1-triplet modal *Un* Σ_1-triplet modal *est un triplet (T, L, J) où T est une spécification modale Σ_1, L est un langage rationnel et J est un langage rationnel tel que $J \subseteq L.\Sigma$ et $J \cap L = \emptyset$. On dit qu'un langage L' clos par préfixe satisfait le triplet (T, L, J) si et seulement si $L' \in mod(T)$, $L \subseteq L'$ et $L' \cap J = \emptyset$. On note $mod(T, L, J)$ l'ensemble des langages satisfaisant (T, L, J).*

Nous utilisons maintenant la notion de Σ_1-triplet modal pour nous abstraire des composantes Σ_0 de S. Le lemme suivant permet de passer d'une spécification $\Sigma_0 \cup \Sigma_1$ à une union finie de Σ_1-triplets modaux.

Lemma 5 *Si S est une spécification modale dans $(\Sigma_0 \cup \Sigma_1)$ alors il existe un ensemble fini et un ensemble de Σ_1-triplets (T_k, L_k, J_k) pour tout $k \in K$ tels que J_k est un langage fini et $mod(S) = \bigcup_{k \in K} mod(T_k, L_k, J_k)$.*

Le principe de décomposition du lemme 5 est le suivant :

1) on exprime la spécification S comme l'union de deux spécifications S^0 et S^1 respectivement Σ_0 et Σ_1,

2) on montre que l'ensemble des modèles d'une spécification Σ_0 forme une union finie d'intervalles de langages. On construit alors un ensemble fini de couple de langages $\{(L_k, J_k)\}_{k \in K}$ tels que $mod(S_0) = \{L \subseteq \Sigma^* \text{ tel que } \exists k \in K, L_k \subseteq L \subset J_k\}$,

3) on construit les Σ_1-triplet (T_k, L_k, J_k) en posant pour tout $k \in K$, $T_k = S^1$.

Exemple 4 *Soit S la spécification sur l'alphabet $\Sigma = \{a, b\}$ définie par $C_a = \{b.b + a.(b.a)^*\}$, $C_b = \{1\}$, $I = \{b.b.b + b.(a+b).a.(a+b) + a.(b.a)^*.a\}$, la décomposition s'effectue la manière suivante :*

1) S^1 est définie par $C_a^1 = \{a.(b.a)^*\}$, $C_b^1 = \emptyset$, $I^1 = \{a.(b.a)^*.a\}$ et S^0 est définie par $C_a^0 = \{b.b\}$, $C_b = \{1\}$, $I = \{b.b.b + b.(a+b).a.(a+b)\}$,

2) nous obtenons deux couples de langages (L_1, J_1) et (L_2, J_2) avec :

$$\begin{array}{lcllcl} L_1 & = & \{1 + b\} & J_1 & = & \{b.a + b.a.a.(a+b)\} \\ L_2 & = & \{1 + b + b.b + b.b.a\} & J_2 & = & \{b.b.b + b.(a+b).a.(a+b)\} \end{array}$$

3) au final, nous obtenons deux Σ_1-triplets modaux : (S^1, L_1, J_1) et (S^1, L_2, J_2).

De fait, d'après le lemme 5, il existe un réseau N tel que son langage est modèle d'une spécification modale S si et seulement si il existe un Σ_1-triplet modal dont $\mathcal{L}(N)$ est modèle.

Nous somme maintenant en mesure d'énoncer le résultat principal de cet article.

Proposition 3 *Soit S une $(\Sigma_0 \cup \Sigma_1)$ spécification modale, l'existence d'un réseau de Petri N tel que $\mathcal{L}(N) \in mod(S)$ est décidable.*

Cette proposition nous permet de construire un réseau de Petri N tel que $\mathcal{L}(N)$ soit modèle d'une $(\Sigma_0 \cup \Sigma_1)$ spécification modale S, s'il existe, avec l'algorithme suivant :

1) calculer les (T_k, L_k, J_k) en utilisant le résultat du lemme 5 ;

2) pour chaque (T, L, J), Σ_1-triplet modal avec $T = \langle \{C_a\}_{a\in\Sigma}, I\rangle$, on calcule les places nécessaires à la synthèse d'un réseau. Une condition suffisante est l'existence d'un ensemble de places vérifiant

- pour tout $u \in J$, il existe une place p telle que $u \notin \mathcal{L}(p)$ ($L(p)$ est le langage du réseau de Petri composé de l'unique place p),
- pour tout $(v.w^*.v') \subseteq I$, il existe une place p telle que $\mathcal{L}(p) \cap (v.w^*.v') = \emptyset$,
- pour toute place p de N, $C_T(\Sigma^*) \subseteq \mathcal{L}(p)$

si ces conditions sont requises, la synthèse est effective et l'algorithme s'achève.

3) sinon, nous transformons chaque Σ_1-triplet (T, L, J) en un ensemble eventuellement vide $\{(T_k, L_k, J_k)\}_{k\in K}$ tel que pour tout $k \in K$, T_k est de taille strictement inférieure à T ; de sorte qu'il existe un réseau de Petri N tel que $\mathcal{L}(N) \in mod(T, L, J)$ si et seulement si il existe un réseau de Petri N' tel que $\mathcal{L}(N') \in mod(\bigcup_{k\in K}(T_k, L_k, J_k))$.

4) si l'ensemble des Σ_1-triplets modaux est vide, la synthèse n'a pas de solutions ; sinon reprendre l'étape 2.

Cet algorithme termine car la taille des T dans chaque triplet (T, L, J) décroît strictement à chaque itération de l'algorithme.

Ce résultat peut enfin être étendu à :

Théorème 5 *Soit S une $(\Sigma_0 \cup \Sigma_1)$ spécification modale et R un langage de Σ_0, l'existence d'un réseau de Petri N tel que $\mathcal{L}(N) \in mod(R^*.S)$ est décidable et effective.*

Ce théorème repose sur le fait que pour tout mot w du langage R, si $w^* \in R$, alors soit $w \notin \mathcal{L}(N)$, soit $w^* \in \mathcal{L}(N)$. Ce résultat étend strictement les travaux de [BAD 99] dès qu'une spécification modale possède une composante Σ_1. En effet, dans ce cas la spécification possède une infinité de modèles, et la synthèse au cas par cas n'est alors pas possible ; ceci n'est pas vrai dans le cas des spécification Σ_0 qui définissent un ensemble fini d'intervalles de langages pour lesquels il est possible de décider de la synthèse. Ce résultat diffère également de celui de [BAD 02] dès qu'une spécification est Σ_1, puisque dans ce cas, d'une part l'ensemble des états du système à synthétiser n'est pas initialement connu, d'autre part il n'est pas nécessairement régulier.

5. Conclusion

Nous avons étudié la synthèse de réseaux de Petri non étiquetés pour des familles de langages décrites par des sentences du nu-calcul conjonctif. La synthèse est montrée décidable pour un fragment caractérisé par une propriété structurelle des spécifications modales associées aux sentences. La simplicité de cette caractérisation suffit à justifier l'intérêt des spécifications modales. Nos résultats étendent les résultats de synthèse pour les langages réguliers dans une direction qui diffère de celle de [BAD 02]

pour les spécifications automatiques. En effet, les spécifications modales et les spécifications automatiques définissent des familles de langages incomparables.
Nous travaillons actuellement sur une extension des résultats présentés ici : soit à une classe plus large de spécifications modales, soit à un fragment plus expressif de la logique. Pour ce dernier cas, nous savons déjà que la synthèse de réseaux pour des formules quelconques du mu-calcul est indécidable.

6. Bibliographie

[ARN 01] ARNOLD A., NIWINSKI D., *Rudiments of mu-calculus*, North-Holland, 2001.

[BAD 99] BADOUEL E., DARONDEAU P., « Theory of regions », *Lectures on Petri Nets I: Basic Models*, vol. 1491 de *Lecture Notes in Computer Science*, p. 529–586, Springer, 1999.

[BAD 02] BADOUEL E., DARONDEAU P., « The Petri net synthesis problem for automatic graphs », rapport n^{o} 4661, December 2002, INRIA Rennes.

[DAR 98] DARONDEAU P., « Deriving Unbounded Petri Nets from Formal Languages », *CONCUR '98: Proceedings of the 9th International Conference on Concurrency Theory*, Springer-Verlag, 1998, p. 533–548.

[JAN 96] JANIN D., WALUKIEWICZ I., « On the Expressive Completeness of the Propositional mu-Calculus with Respect to Monadic Second Order Logic », MONTANARI U., SASSONE V., Eds., *CONCUR '96: Concurrency Theory, 7th International Conference*, vol. 1119, Pisa, Italy, 26–29 1996, Springer-Verlag, p. 263–277.

[KOZ 83] KOZEN D., « Results on the propositional mu-calculus », *Theoretical Computer Science*, vol. 27, 1983, p. 333–354.