Propagation de contraintes et ordonnancement de documents multimedia

Nicolas RIVIERE
LAAS-CNRS
F-31077 Toulouse Cedex 4
nriviere@laas.fr

Brigitte PRADIN-CHEZALVIEL
LAAS-CNRS et IUT A - UPS
F-31077 Toulouse Cedex 4
chezalvi@laas.fr

Robert VALETTE
LAAS-CNRS
F-31077 Toulouse Cedex 4
robert@laas.fr

Résumé

Le propos de cet article est de montrer que l'utilisation conjointe des réseaux de Petri p-temporel, des graphes AOA et des TCSP nous permet de trouver au moins une solution satisfaisante à un problème d'ordonnancement en temps contraint pour chaque ordre partiel sans avoir à construire le graphe de tous les états possibles de notre modèle. Nous présenterons comment la logique linéaire nous permet de générer un ordre partiel vérifiant les contraintes logiques sans avoir à générer tout le graphe d'état. Un cas d'étude de présentation de document interactif multimedia sera utilisé pour bien montrer la méthode utilisée afin de trouver une solution d'ordonnancement.

Mots clefs : Logique linéaire, Réseaux de Petri temporels, TSCP, Graphes potentiels événements

1 Introduction

La conception de documents multimedia interactifs est aujourd'hui un domaine d'application qui, avec la complexité grandissante des documents produits (animations graphiques, audio, video, audio-video), nécessite un ordonnancement (une synchronisation très précise entre les différents éléments qui composent les documents) compatible avec un ensemble de contraintes temporelles. La prise en compte de l'aspect temps réel dans ce type d'application oblige les concepteurs à traiter de manière explicite la présentation du document et ses contraintes : la conception dépend de la présentation attendue. Il apparaît donc essentiel d'avoir un ordonnancement des différents éléments du document respectant le mieux possible toutes les contraintes temporelles liées à sa présentation.

Afin de pouvoir ordonnancer correctement une présentation, il est nécessaire de connaître les différents ordres partiels qui permettent d'assurer une synchronisation correcte de tous les éléments qui composent le document multimedia.

Pour cela une technique d'ordonnancement basée sur une approche formelle pour la présentation de documents multimedia interactifs a été développée au LAAS-CNRS [9][8]. La méthodologie de conception formelle de ces documents comporte plusieurs étapes. La première étape est une description du comportement du document lors de sa présentation à l'aide d'un modèle auteur de haut niveau tel que SMIL (Synchronised Multimedia Integration Language), NCM (Nested Context Model), etc... La structure temporelle et logique qui a été définie est traduite en une spécification formelle RT-LOTOS. À partir de là est dérivé un graphe d'accessibilité minimale dont on vérifie les propriétés de cohérence. Lorsque les contraintes temporelles du document sont respectées il est possible de générer un ordonnancement de sa présentation ; le graphe obtenu contient tous les comportements vérifiant la cohérence du document. Dans le cas contraire, il faudra revoir la description jusqu'à satisfaire les contraintes. Cette méthodologie est donc fondée sur un graphe d'état énumérant tous les entrelacements

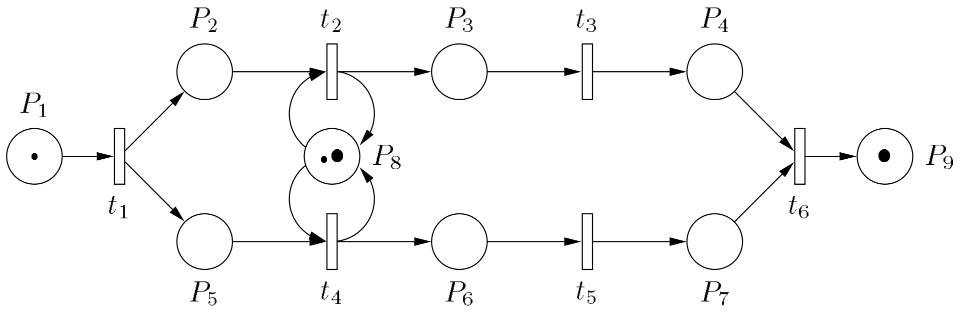

FIG. 1: Réseau de Petri avec conflit

des ordres partiels vérifiant une accessibilité logique mais pas toujours temporelle. Ensuite les branches dites *inconsistantes* afin d'obtenir un graphe *consistant* sont éliminées.

Nous nous proposons par notre méthode de générer les ordres partiels sans énumérer tous les entrelacements [7] puis de vérifier les contraintes temporelles du document multimedia. Pour cela, nous utilisons plusieurs outils : les réseaux de Petri temporels [1], la logique linéaire [4] et les TCSP (Temporal Constraint Satisfaction Problem) [10].

2 Utilisation de la logique linéaire pour trouver un ordre partiel

2.1 Rappels sur la logique linéaire

Un réseau de Petri définit des relations de précédence entre les transitions. L'analyse de l'accessibilité (entre deux marquages) est basée de façon classique sur des séquences de tir dans lesquelles le franchissement des transitions apparaît dans un ordre total. L'utilisation de la logique linéaire [4] nous permet d'obtenir des ordres partiels consistants avec les relations de précédence induites par le réseau [3]. Pour cela nous utilisons l'équivalence entre accessibilité dans un réseau de Petri et prouvabilité de séquent en logique linéaire [5] que nous allons présenter par la suite.

Une formule logique est associée à chaque marquage et à chaque instance de franchissement d'une transition. Un marquage M est un monôme en $\otimes$, *i.e.*, que l'on écrit sous la forme $A_1 \otimes A_2 \otimes ... \otimes A_k$, où les A_i sont les noms des places du réseau de Petri. Pour toute place A_k qui contient plusieurs jetons, la proposition A_k apparaît autant de fois que la place contient de jetons.

Une transition est une formule du type $M_1 \multimap M_2$ où M_1 et M_2 sont les marquages, soit les fonctions Pre et Post des transitions. Cette expression représente le franchissement d'une transition. Elle apparaitra autant de fois dans le séquent que cette transition doit être franchie. Sur la figure 1, le franchissement de t_1 sera traduit en logique linéaire par $P_1 \multimap P_2 \otimes P_5$.

Un séquent est associé à un scénario où le marquage initial et les franchissements des transitions sont les prémisses et le marquage final, la conclusion. Nous prouvons alors le séquent à l'aide des règles du calcul des séquents. Le fait que les atomes correspondant aux propositions logiques puissent être comptés, produits et consommés exactement comme des jetons de places de réseau de Petri montre bien que la prouvabilité d'un séquent est équivalente à l'accessibilité du marquage final depuis l'initial. L'ensemble des franchissements de transition montre clairement quelles transitions sont tirées. Durant la preuve, toutes les contraintes générées par le réseau et le marquage initial sont vérifiées.

Le séquent $M, \sigma \vdash M'$ représente un scénario où $\sigma = t_i, ..., t_n$ est la liste non ordonnée (monoïde commutatif) des différentes instances de tir des transitions concernées et M et M' sont les marquages initiaux et finaux.

La preuve est toujours matérialisée par un arbre qui se lit de bas en haut dans le cadre du calcul des séquents. La preuve démarre par le séquent à prouver et se termine lorsque les feuilles de l'arbre sont

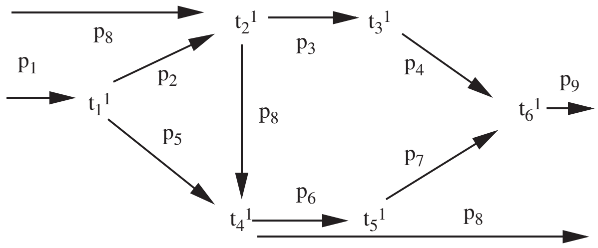

Fig. 2: Ordre partiel 1

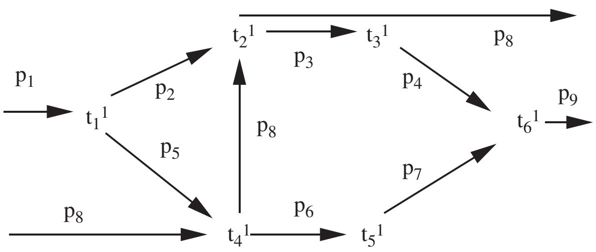

Fig. 3: Ordre partiel 2

des séquents *identité*, $A \vdash A$ par exemple. Plusieurs arbres de preuves sont possibles mais la preuve est construite de manière canonique. Dans cet article, nous nous plaçons dans le fragment ILL (Intuitionist Linear Logic) de la logique linéaire ce qui réduit l'utilisation des connecteurs au nombre de deux :

- le connecteur $\otimes$ représente la disponibilité simultanée de ressources,
- le connecteur $\multimap$ est l'implication linéaire qui représente la causalité car les atomes du côté gauche du connecteur doivent être consommés pour produire ceux du côté droit.

2.2 Dérivation d'un ordre partiel

Le début d'une preuve consiste à éliminer le connecteur $\otimes$ dans le marquage initial. Cela va nous permettre de travailler avec une liste de jetons (atomes logiques) logiquement indépendants mais pas nécessairement présents simultanément plutôt qu'avec des marquages. Durant la preuve, ces jetons correspondent à des relations de précédence. Soit t_i^j le $j^{\text{ème}}$ tir de la transition t_i. Si un jeton est produit par un tir de transition t_i^j et est consommé par t_k^l, cela signifie que t_i^j doit précéder t_k^l (noté $t_i^j \prec t_k^l$). Une paire (t_i^j, t_k^l) est associée à chaque jeton : ils représentent respectivement le tir de transition qui a produit le jeton et celui qui l'a consommé. Quand un tir n'est pas encore défini (encore inconnu), il est noté par un point. A la fin de la preuve, il suffit juste de collecter tous les atomes qui sont apparus. En effet chacun d'eux correspond à une relation de précédence et l'ensemble de ces derniers définit l'ordre partiel.

Durant la preuve, chaque pas permet l'élimination d'une formule correspondant à une transition. A chaque pas de la preuve, une transition t_i^j est éliminée de la liste des formules, les jetons consommés par t_i^j sont enlevés de l'ensemble des atomes avant franchissement et ceux produits ajoutés. Si à un pas donné de la preuve plus d'une élimination est possible et si les transitions et les jetons concernés sont disjoints, alors la preuve qui en découlera sera la même quel qu'en soit l'ordre d'élimination. Par contre si les transitions ou les jetons sont partagés (conflits jetons ou conflits transitions définis comme dans [7]), alors un arbre de preuve pour chaque ordre partiel devra être dérivé car chaque arbre caractérise un ordre partiel différent.

Exemple

Considérons le modèle de la figure 1 où il y a deux chemins distincts à exécuter et une ressource (H) partagée. Les événement initiaux et finaux sont t_1^1 et t_6^1. Les deux chemins contiennent deux actions chacun : t_2^1 et t_3^1 pour le premier et t_4^1 et t_5^1 pour le second. Les actions t_2 et t_4 partagent la même ressource. Le marquage initial est $P_1 \otimes P_8$ et le marquage final est $P_8 \otimes P_9$.

Après l'élimination de la transition t_1^1, il y a un conflit transition entre t_2 et t_4 car le marquage P_2, P_5, P_8 est atteint et que les deux transitions doivent consommer P_8. Deux ordres partiels sont ainsi

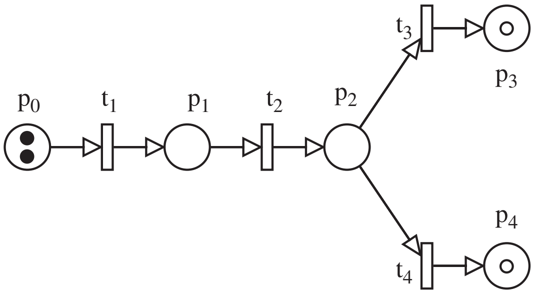

FIG. 4: Réseau de Petri de la preuve

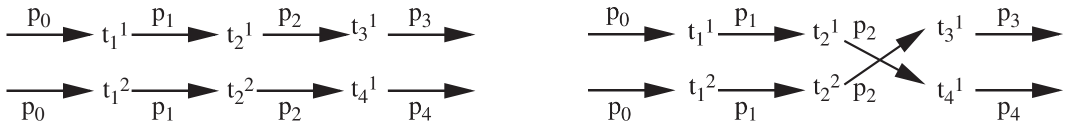

FIG. 5: Ordres partiels de la preuve

dérivés (figures 2 et 3). P_8 apparaît trois fois dans chaque ordre partiel car il en existe un exemplaire initialement qui est consommé et produit par t_2^1 puis reconsommé et reproduit par t_4^1.

2.3 Construction d'un arbre de preuves

Nous allons montrer par un exemple comment construire un arbre de preuves en utilisant les instances de tir des transitions comme estampilles. Si durant la construction d'un arbre de preuves nous n'utilisons jamais la règle *cut*, aucune mauvaise relation de précédence ne sera introduite. L'arbre contient toutes les contraintes de l'ordre partiel définies par le réseau de Petri entre le marquage initial et le marquage final et prises en compte pour la résolution du conflit.

La méthode et les résultats qui en découlent vont être montrés avec l'exemple de la figure 4. Les deux ordres partiels qui vérifient les contraintes définies par le réseau de Petri, pour le marquage initial M (composé de deux jetons dans la place p_0) et le marquage final M' (composé d'un jeton dans p_3 et d'un dans p_4), sont représentés sur la figure 5.

Cette preuve correspond à l'ordre partiel de gauche de la figure 5. Au vu de sa taille, l'arbre de preuve est décomposé en plusieurs pas. La formule $(p_0 \multimap p_1)^{\otimes 2}$ est une abréviation pour $(p_0 \multimap p_1) \otimes (p_0 \multimap p_1)$. Cela signifie que la transition correspondante sera tirée deux fois.

Pas 1 : La règle $\otimes_L$ transforme le marquage initial $p_0 \otimes p_0(.,.)$ en une liste d'atomes.

$$\frac{p_0(.,.), p_0(.,.), (p_0 \multimap p_1)^{\otimes 2}, (p_1 \multimap p_2)^{\otimes 2}, (p_2 \multimap p_3), (p_2 \multimap p_4) \vdash p_3 \otimes p_4}{p_0 \otimes p_0(.,.), (p_0 \multimap p_1)^{\otimes 2}, (p_1 \multimap p_2)^{\otimes 2}, (p_2 \multimap p_3), (p_2 \multimap p_4) \vdash p_3 \otimes p_4} \otimes_L \tag{1}$$

Pas 2 : La première transition traitée est t_1 $(p_0 \multimap p_1)$ par application de la règle $\multimap_L$.

$$\frac{p_0(., t_1^1) \vdash p_0 \quad p_0, p_1(t_1^1, .), (p_0 \multimap p_1), (p_1 \multimap p_2)^{\otimes 2}, (p_2 \multimap p_3), (p_2 \multimap p_4) \vdash p_3 \otimes p_4}{p_0(.,.), p_0(.,.), (p_0 \multimap p_1)^{\otimes 2}, (p_1 \multimap p_2)^{\otimes 2}, (p_2 \multimap p_3), (p_2 \multimap p_4) \vdash p_3 \otimes p_4} \multimap_L(t_1) \tag{2}$$

Les lignes supérieures contiennent deux séquents. Celui de gauche est une séquent identité et par conséquent une feuille de l'arbre. Le fait que le label de l'atome p_0 sur la gauche soit $(., t_1^1)$, cela signifie que nous avons une relation de précédence entre le point de départ et le tir t_1^1.

Il reste à prouver le séquent de droite.

Pas 3 : Pour ce pas, la règle $\multimap_L$ peut être appliquée soit à t_1 soit à t_2 : nous considérons le plus petit ordre lexicographique et traitons donc en premier t_1.

$$\frac{p_0(., t_1^2) \vdash p_0 \quad p_1(t_1^1, .), p_1(t_1^2, .), (p_1 \multimap p_2)^{\otimes 2}, (p_2 \multimap p_3), (p_2 \multimap p_4) \vdash p_3 \otimes p_4}{p_0, p_1(t_1^1, .), (p_0 \multimap p_1), (p_1 \multimap p_2)^{\otimes 2}, (p_2 \multimap p_3), (p_2 \multimap p_4) \vdash p_3 \otimes p_4} \multimap_L (t_1) \tag{3}$$

Le séquent identité de gauche porte le label $(., t_1^2)$ ce qui signifie qu'il existe une relation de précédence entre le point de départ et le tir de t_1^2.

Pas 4 : Ici, nous traitons t_2.

$$\frac{p_1(t_1^1, t_2^1) \vdash p_1 \quad p_1(t_1^2, .), p_2(t_2^1, .), (p_1 \multimap p_2), (p_2 \multimap p_3), (p_2 \multimap p_4) \vdash p_3 \otimes p_4}{p_1(t_1^1, .), p_1(t_1^2, .), (p_1 \multimap p_2)^{\otimes 2}, (p_2 \multimap p_3), (p_2 \multimap p_4) \vdash p_3 \otimes p_4} \multimap_L (t_2) \tag{4}$$

Le séquent identité de gauche porte le label (t_1^1, t_2^1) ce qui signifie qu'il existe une relation de précédence entre les tirs de t_1^1 et t_2^1 $(t_1^1 \prec t_2^1)$.

Pas 5 : Nouveau tir de t_2 (elle précède t_3 and t_4 dans l'ordre lexicographique).

$$\frac{p_1(t_1^2, t_2^2) \vdash p_1 \quad p_2(t_2^1, .), p_2(t_2^2, .), (p_2 \multimap p_3), (p_2 \multimap p_4) \vdash p_3 \otimes p_4}{p_1(t_1^2, .), p_2(t_2^1, .), (p_1 \multimap p_2), (p_2 \multimap p_3), (p_2 \multimap p_4) \vdash p_3 \otimes p_4} \multimap_L (t_2) \tag{5}$$

Maintenant, le séquent identité sur la gauche possède le label (t_1^2, t_2^2) qui indique la relation de précédence entre t_1^2 et t_2^2 $(t_1^2 \prec t_2^2)$.

Pas 6 : Il y a un conflit entre les deux transitions t_3 et t_4 car les deux consomment p_2 et bien qu'il y ait deux jetons dans p_2, ils ne sont pas identiques, leur label diffère. Les choix sont :

- $p_2 \multimap p_3$ consomme $p_2(t_2^1, .)$ et alors $p_2 \multimap p_4$ consomme $p_2(t_2^2, .)$;
- $p_2 \multimap p_3$ consomme $p_2(t_2^2, .)$ et alors $p_2 \multimap p_4$ consomme $p_2(t_2^1, .)$.

Chaque choix produit un autre partiel différent. Ici le premier choix est sélectionné (tir de t_3 en premier).

$$\frac{p_2(t_2^1, t_3^1) \vdash p_2 \quad p_2(t_2^2, .), p_3(t_3^1, .), (p_2 \multimap p_4) \vdash p_3 \otimes p_4}{p_2(t_2^1, .), p_2(t_2^2, .), (p_2 \multimap p_3), (p_2 \multimap p_4) \vdash p_3 \otimes p_4} \multimap_L (t_3) \tag{6}$$

Le séquent identité de gauche porte le label (t_2^1, t_3^1) : $t_2^1 \prec t_3^1$.

Pas 7 : Le dernier tir de transition est t_4.

$$\frac{p_2(t_2^2, t_4^1) \vdash p_2 \quad p_3(t_3^1, .), p_4(t_4^1, .) \vdash p_3 \otimes p_4}{p_2(t_2^2, .), p_3(t_3^1, .), (p_2 \multimap p_4) \vdash p_3 \otimes p_4} \multimap_L (t_4) \tag{7}$$

Le séquent identité de gauche porte le label $(t_2^2, t_4^1) : t_2^2 \prec t_4^1$.

Pas 8 : L'arbre se termine avec deux axiomes identité créés par l'utilisation de la règle $\otimes_R$.

$$\frac{p_3(t_3^1, .) \vdash p_3 \quad p_4(t_4^1, .) \vdash p_4}{p_3(t_3^1, .), p_4(t_4^1, .) \vdash p_3 \otimes p_4} \otimes_R \tag{8}$$

Deux séquents identité sont dérivés. Les labels des atomes expriment les relations de précédence entre t_3^1 et la fin du scénario pour le premier et entre t_4^1 et la fin du scénario pour le second.

Si les événements initiaux et finaux ne sont pas pris en compte, la solution correspond à assigner des valeurs binaires aux variables $x_{i,j,k,l}$ caractérisant les contraintes de précédence ; si $x_{i,j,k,l} = 1$ alors t_i^j doit précéder t_k^l :

$$\begin{array}{ll} x_{1,1,2,1} = 1 & x_{1,2,2,2} = 1 \\ x_{2,1,3,1} = 1 & x_{2,2,4,1} = 1 \end{array} \tag{9}$$

toutes les autres variables sont égales à zéro.

Ceci correspond à l'ordre partiel de gauche sur la figure 5. Celui de droite est dérivé quand le second choix est sélectionné pour le conflit au pas 6.

3 Ordonnancement temporel à l'aide de graphes AOA

3.1 Introduction des CSP et des graphes potentiels événements

Dans un travail précédent sur les relations entre les réseaux de Petri et les graphes de contraintes [6], nous avons vu qu'il était possible de traduire l'ensemble des contraintes générées par un réseau de Petri, un marquage initial et une liste de franchissement de transitions sous la forme d'un CSP (Constraint Satisfaction Problem) [11]. Toutefois, le CSP obtenu est une forme dégénérée avec une contrainte unique qui est la disjonction de l'ensemble des ordres partiels solutions des contraintes.

Un CSP est défini par un triplet (X, D, C) où :

- $X = x_1, ..., x_n$ est un ensemble de n variables,
- $D = d_1, ..., d_n$ est l'ensemble des domaines des variables,
- $C = c_1, ..., c_e$ est un ensemble de e contraintes, où chaque c_i est définie par l'ensemble des variables $X(c_i) \subset X$ impliqué dans la contrainte c_i et par une relation $R(c_i)$ entre les variables de $X(c_i)$.

Le fait de travailler avec des contraintes temporelles nous amène à travailler avec une certaine classe de CSP : les TCSP. Un TCSP (temporal constraints satisfaction problem) est une classe particulière des CSP où l'ensemble X des variables dénote un ensemble d'entités temporelles (dates, intervalles temporels, durées) et les contraintes C représentent les relations logiques ou temporelles possibles entre les variables. Toutes les contraintes de type TCSP sont binaires [10]. Il faut noter que comme nous ne travaillons qu'avec un seul intervalle temporel liant deux variables, nous nous plaçons dans le cadre d'un STP (Simple Temporal Problem). Pour chaque ordre partiel il y a un ensemble de contraintes suffisamment simple pour être résolu en un temps polynômial et pour permettre des procédures simples d'*arc consistence*.

Pour prendre en compte le temps et représenter de manière graphique un TCSP, nous utilisons les graphes AOA (activity-on-arc) [2]. Un graphe AOA est un graphe potentiel événement où les noeuds

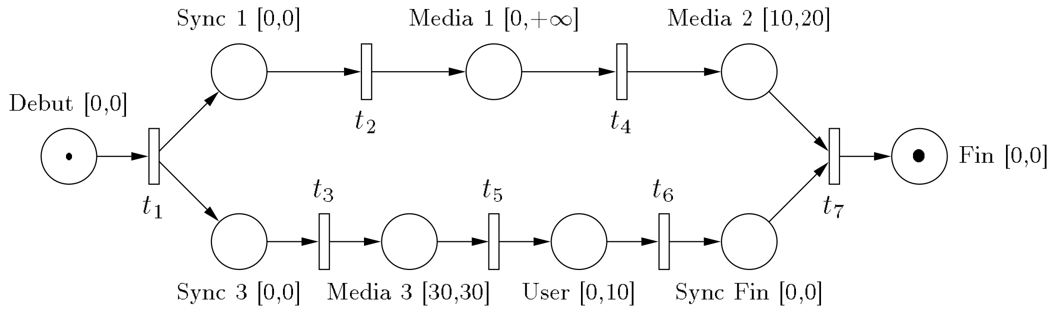

FIG. 6: Réseau de Petri P-temporel

représentent les variables X du TCSP (soit des événements) et les arêtes les contraintes temporelles entre deux événements. A chaque noeud $i \rightarrow j$ est associée une contrainte binaire c_{ij}. Les noeuds représentent des événements ou des pas relatifs à des débuts ou des fins d'activités. La longueur d'un arc représente la durée de l'activité attachée à cet arc. Etant donné que nous sommes capables de générer un CSP à partir d'un réseau de Petri, nous pouvons donc associer un graphe AOA à un réseau temporel ou temporisé.

3.2 Les réseaux de Petri et le temps

Ordonnancer un ensemble d'opérations est équivalent à trouver un ordre partiel parmi les tirs de transition correspondant aux opérations à faire et à fixer leur date de tir de façon à respecter toutes les contraintes. Dans notre contexte, le modèle réseau de Petri doit être complémenté par des contraintes temporelles (durées d'opérations, dates voulues, etc...). Il existe des réseaux de Petri, enrichis par des extensions temporelles, qui permettent de spécifier les contraintes complexes impliquant le temps et les ressources. Ce sont les modèles p-temporel, t-temporel, p-temporisé et t-temporisé.

Dans notre approche, nous utilisons le modèle p-temporel car il a été montré dans [6] que lors d'une traduction d'un réseau de Petri temporel en un graphe AOA, il était plus naturel d'associer des durées aux places plutôt qu'aux transitions car les contraintes de précédence sont générées par des places.

Un réseau de Petri p-temporel (figure 6) est un réseau de Petri P avec une fonction associant une durée minimale d_{imin} et une durée maximale d_{imax} à chaque place p_i. Chaque jeton doit rester au moins d_{imin} dans p_i et doit être consommé par le tir d'une transition avant d_{imax}.

Ce modèle associe deux contraintes à chaque relation de précédence, une pour la durée minimale et une pour la durée maximale. Il est possible d'avoir des circuits positifs ce qui signifie que l'ensemble de nos contraintes est inconsistent. En effet il n'est pas plausible qu'un circuit composé de durées minimales soit supérieur à un circuit composé de durées maximales. Les réseaux de Petri p-temporel sont capables de prendre en compte les échéances et sont plus généraux que les réseaux p-temporisé. La recherche d'une solution peut être faite en utilisant la programmation linéaire.

3.3 Calcul des domaines des dates de tir : exemple

Une fois la traduction de l'ordre partiel (obtenu par un arbre de preuve en logique linéaire) et des contraintes temporelles effectuée sous la forme d'un graphe AOA, les techniques classiques de propagation des contraintes permettent d'obtenir les dates de franchissement *au plus tôt* et les dates de franchissement *au plus tard* compatibles avec les contraintes.

Pour illustrer notre méthode, nous allons considérer le scénario multimedia qui décrit la présentation simultanée de deux médias (1 et 2) avec un média interactif 3. La présentation du média 1 a une durée de présentation indéterminée $[0, +\infty[$ en secondes. Le média 2, qui démarre après le média 1, a une durée comprise entre $[10, 20]$. 3 est un média interactif qui dure au maximum 40 secondes. L'aspect

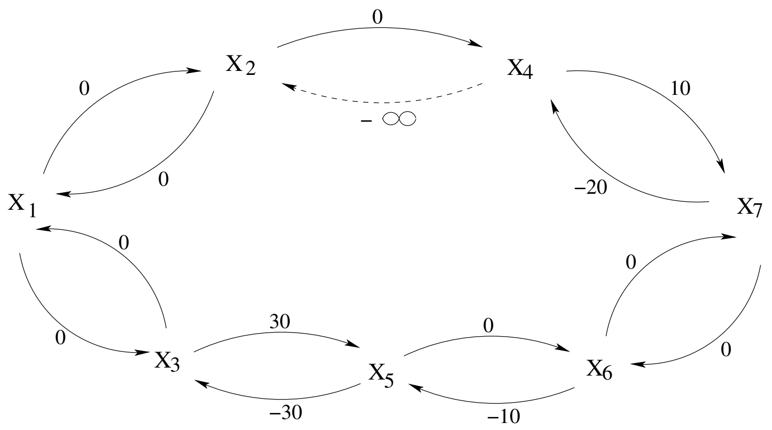

FIG. 7: Graphe AOA

interactif vient du fait que l'utilisateur a la possibilité d'interrompre ce média entre 30 et 40 secondes sans quoi sa présentation se termine automatiquement lorsque la durée maximale du media 3 est atteinte. Les présentations des médias 1 et 3 débutent en même temps et celles des médias 2 et 3 doivent se terminer en même temps. D'après ces spécifications nous devons donc trouver une durée maximale de présentation du média 1 afin de pouvoir satisfaire les contraintes de synchronisation de notre scénario.

A partir des spécifications, un modèle réseau de Petri p-temporel est créé (figure 6). Il est à noter que les noms des places de ce réseau ne correspondent pas aux noms des médias spécifiés ci-dessus. La branche composée des places $Sync1$, $Media1$ et $Media2$ représente les présentations successives des médias 1 et 2. La branche composée des places $Sync3$, $Media3$, $User$ et $SyncFin$ représente la présentation du média 3 avec l'interaction de l'utilisateur modélisée par la place $SyncFin$: en effet, si la transition t_6 est franchie avant que la durée liée à la place $User$ n'atteigne sa valeur maximale (10 secondes), il faut immédiatement sensibiliser la transition t_7 d'où une attente nulle sur $SyncFin$.

En considérant le modèle de la figure 6, nous générons le graphe AOA de la figure 7. Dans notre exemple, étant donné que la durée maximale de $Media1$ (égale à $+\infty$) ne contraint pas l'activité liée à la place $Media1$, l'arc inverse (en pointillé sur la figure 7) reliant la variable X_4 à la variable X_2 est éliminé. Il nous reste maintenant à quantifier le domaine de l'intervalle lié à X_4. Pour cela, on résout un système linéaire d'inéquations déduit du graphe AOA de la figure 7.

La première étape consiste donc à énumérer toutes les inéquations du système linéaire.

$$\begin{array}{l|l|l}
X_2 - X_1 \geq 0 & X_3 - X_1 \geq 0 & X_6 - X_5 \geq 0 \\
X_1 - X_2 \geq 0 & X_1 - X_3 \geq 0 & X_5 - X_6 \geq -10 \\
X_4 - X_2 \geq 0 & X_5 - X_3 \geq 30 & X_7 - X_6 \geq 0 \\
X_7 - X_4 \geq 10 & X_3 - X_5 \geq -30 & X_6 - X_7 \geq 0 \\
X_4 - X_7 \geq -20 & &
\end{array}$$

D'après les premières inéquations, des simplifications peuvent être effectuées. Nous posons donc que X_{123} est la variable qui représente X_1 ou X_2 ou X_3 indifféremment. De même nous posons X_{67} comme la variable unique pour X_6 et X_7.

$$\left.\begin{array}{l} X_2 = X_1 = 0 \\ X_3 = X_1 = 0 \end{array}\right\} \Rightarrow X_{123} = 0$$

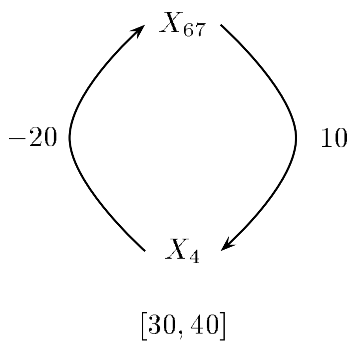

FIG. 8: Graphe AOA réduit

Nous obtenons donc le nouveau système suivant :

$$\begin{array}{rcr|rcr}
X_{123} & = & 0 & X_5 - X_{123} & \geq & 30 \\
X_4 - X_{123} & \geq & 0 & X_{123} - X_5 & \geq & -30 \\
X_{67} - X_4 & \geq & 10 & X_{67} - X_5 & \geq & 0 \\
X_4 - X_{67} & \geq & -20 & X_5 - X_{67} & \geq & -10
\end{array}$$

De nouvelles simplifications nous donnent le système suivant :

$$\begin{array}{rcr|rcr}
X_4 & \geq & 0 & X_5 & = & 30 \\
X_{67} - X_4 & \geq & 10 & X_{67} - X_5 & \geq & 0 \\
X_4 - X_{67} & \geq & -20 & X_5 - X_{67} & \geq & -10
\end{array}$$

Cela nous amène au système final :

$$X_{67} - X_4 \geq 10 \tag{10}$$
$$X_4 - X_{67} \geq -20 \tag{11}$$
$$\text{avec} \quad 30 \leq X_{67} \leq 40 \tag{12}$$

D'après ce dernier système nous obtenons le nouveau graphe AOA de la figure 8.

Cela correspond bien aux spécifications de notre document multimedia. En effet l'utilisateur peut stopper la présentation du média 3 entre 30 et 40, ce que montre l'encadrement de X_{67}. Cet encadrement correspond au domaine de valeurs que peut prendre cette variable.

A partir des inéquations (10) et (11) et du domaine incertain de X_{67}, nous avons deux cas possibles de valeurs pour X_4 :

- 1er cas : nous pouvons trouver un domaine de valeurs pour X_4 pour lequel il existe au moins une solution possible pour X_{67}.
- 2ème cas : nous pouvons trouver une valeur pour X_4 pour laquelle toutes les valeurs de X_{67} sont possibles.

En prenant la première inéquation et en posant $X_{67} = 30$, nous nous plaçons dans le cas le plus contraint pour les valeurs possibles de X_4 : nous obtenons $X_4 \leq 20$. En posant $X_{67} = 40$, nous sommes dans le cas le moins contraint et nous obtenons $X_4 \leq 30$.

Maintenant, avec la deuxième inéquation, si nous posons $X_{67} = 40$ nous sommes dans le cas le plus contraint et $X_4 \geq 20$. Et inversement avec $X_{67} = 30$ nous sommes dans le cas le moins contraint, ce qui nos donne $X_4 \geq 10$.

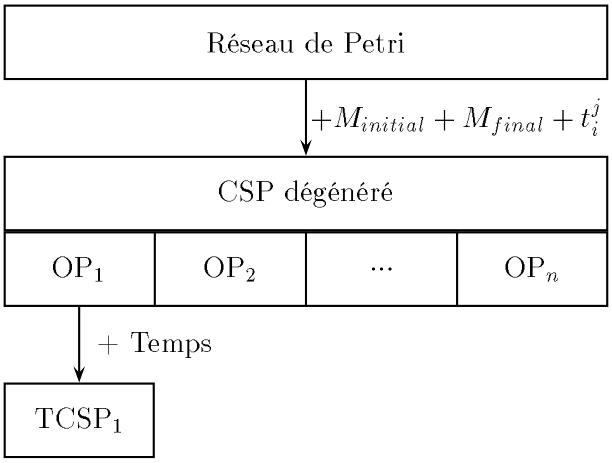

Fig. 9: Méthode globale

Nous pouvons dire que l'encadrement $20 \leq X_4 \leq 20$, soit $X_4 = 20$, qui correspond au cas le plus contraint, caractérise l'ensemble des valeurs de X_4 pour lequel toutes les valeurs de X_{67} satisfont les contraintes de synchronisation. Par contre l'encadrement $10 \leq X_4 \leq 30$, qui correspond au cas le moins contraint, caractérise l'ensemble des valeurs de X_4 pour lequel il existe au moins une valeur de X_{67} qui satisfait les contraintes de synchronisation du scénario. L'intervalle temporel de la place D du réseau p-temporel de la figure 6 ne doit donc plus être $[0, \infty]$ mais l'intervalle de l'un des deux cas caractérisés ci-dessus suivant la flexibilité que l'on veut laisser à l'utilisateur pour interagir. Pour le plus de flexibilité on choisira $[20, 20]$ sinon on choisira $[10, 30]$ sachant que l'on contraindra l'utilisateur à ne pouvoir éventuellemnt agir qu'en un instant donné.

4 Conclusion

Dans cet article nous avons montré que l'utilisation conjointe des réseaux de Petri p-temporel, des graphes AOA et des TCSP nous permettait de trouver au moins une solution satisfaisante à un problème d'ordonnancement en temps contraint pour chaque ordre partiel sans avoir à construire le graphe de tous les états possibles de notre modèle. La solution obtenue n'est bien sûr valable que pour l'ordre partiel généré par l'utilisation de la logique linéaire et dans lequel nous avons propagé les contraintes temporelles. Dans le cas où un conflit apparaît, autant d'ordres partiels que de chemins différents devront être générés pour lesquels nous pouvons trouver une solution qui sera différente pour chaque ordre partiel. La figure 9 montre bien comment on accède de n'importe quel réseau de Petri à un TSCP (associé à un ordre partiel).

En perspective, il serait intéressant de comparer les solutions obtenues pour chaque ordre partiel avec les solutions obtenues en se basant sur le graphe d'états complet car il n'est pas certain que les solutions trouvées à partir du graphe global soient l'union des solutions trouvées pour chaque ordre partiel. Pour cela, un outil est prévu afin d'automatiser la génération des ordres partiels pour faciliter notre travail, outil basé sur la réécriture des règles de logique linéaire à l'aide d'un langage de programmation. Il serait aussi intéressant de récupérer ces ordres partiels et de les traiter avec un outil qui utiliserait un algorithme optimal de programmation linéaire.

Références

[1] P. Bonhomme. *Réseaux de Petri p-temporels : contributions à la commande robuste*. PhD thesis, LAMII/CESALP/ESIA, Université de Savoie, Annecy, France, 12 Juillet 2001.

[2] S.E. Elmaghraby. *Activity networks : project planning and control by network models*. John Wiley & sons, New York, 1977.

[3] V. Gehlot. *A proof theoretic approach to semantics of concurrency*. Phd thesis, University of Pennsylvania, 1992.

[4] Jean-Yves GIRARD. Linear logic. *Theoretical Computer Science*, 50 :1–102, 1987.

[5] François GIRAULT. *Formalisation en logique linéaire du fonctionnement des réseaux de Petri*. PhD thesis, Université Paul Sabatier, Toulouse, France, Décembre 1997.

[6] C. Mancel, P.Lopez, N.Rivière, and R. Valette. Relationships between petri nets and constraint graphs : application to manufacturing. In *Rapport LAAS 01396*, Toulouse, France, September 2001.

[7] Brigitte PRADIN-CHEZALVIEL, Luis Allan KÜNZLE, and Robert VALETTE. Scenario durations characterization of t-timed Petri nets using Linear Logic. *PNPM*, 1999.

[8] P.N.M. Sampaio and J.P. Courtiat. Scheduling and presenting interactive multimedia documents. In *2001 IEEE International Conference on Multimedia (ICME'2001)*, pages 1224–1227, Tokyo, Japan, August 22-25 2001.

[9] P.N.M. Sampaio and J.P. Courtiat. A formal approach for the presentation of interactive multimedia documents. In *8th ACM International Conference on Multimedia*, pages 435–438, Los Angeles, USA, October 30 - November 4 2000.

[10] E. Schwalb and L. Vila. Temporal constraints : a survey. *Constraint : an International journal (Kluwer Academic)*, 2 :129–149, 1998.

[11] E.P.K. Tsang. *Foundations of constraint satisfaction*. Academic Press Ltd., London, 1993.