Validation d'un modèle basée sur les RdPT pour la spécification et l'analyse des systèmes temps réel complexes : Application aux systèmes multimédias

Abdelkrim Abdelli

Université des Sciences et Technologies Houari Boumedienne

Faculté d'informatique et d'électronique
Département d'informatique

Validation d'un modèle basé sur les *RdPT* pour la spécification et l'analyse des systèmes temps réel complexes : Application aux systèmes multimédias

Abdelkrim Abdelli

Docteur es Informatique

Date de soutenance : 03 Octobre 2007

Directeur de thèse:

Prof. Nadjib Badache	Rapporteur	USTHB

Devant le jury composé de:

Pr Malika Ioualalen	Président	USTHB
Pr Hanifa Boucheneb	Examinateur	EP Montréal Quebec
Dr Faiza Bellala	Examinateur	USC Constantine
Dr Abdelkader Belkhir	Examinateur	USTHB
Dr Larbi Sakhri	Examinateur	UST Oran

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction Générale

1.1 Domaine d'interêt

De nos jours, les systèmes parallèles sont de plus en plus répandus dans le monde informatique ; ils sont caractérisés par leur puissance et leur complexité. Une classe importante des systèmes informatiques est celle des systèmes temps réel.

Les systèmes temps réel sont des systèmes dont le comportement doit satisfaire des contraintes temporelles strictes. Ils sont en général caractérisés par des interactions complexes avec l'environnement ; ils doivent par conséquent nécessairement être fiables. Il comprennent les systèmes hautement réactifs dont les décisions peuvent avoir des conséquences immédiates et irréversibles sur des biens et des hommes. Leurs domaines d'application est large, allant de la simple utilisation domestique tels que les systèmes de contrôle des chauffages et des cuisinières, jusqu'aux systèmes les plus complexes et les plus critiques tels que les systèmes de commande de l'avionique, le nucléaire, l'industrie et les télécommunications.

Pour vérifier la fiabilité de tels systèmes, les méthodes formelles ont été largement utilisées et constituent une étape importante dans leur processus d'élaboration. Cela commence par la proposition d'un modèle mathématique capable de décrire fidélement leurs comportements temporels et s'en suit la proposition de techniques d'analyse adaptées à ce modèle, permettant de vérifier qualitativement et quantitativement les propriétés des systèmes modélisés.

Par la suite, nous inscrivons notre travail dans un cadre particulier relatif à la spécification et à la vérification des applications multimédias dont les exigences résument en partie les besoins observés dans les systèmes temps réel. Les propositions que nous développons dans cette thèse pourraient trouver écho au près des applications temps réel au sens large, du fait qu'elles permettent de définir un cadre formel unifié permettant la prise en charge d'une grande partie des problématiques posées dans ce type de systèmes.

1.2 Problématique et Motivation

Récemment, la demande de preuves de correction des systèmes dit *faiblement synchrones* croît rapidement. Le concept des systèmes faiblement synchrones [1], couvrent la grande famille

1. Les systèmes où les durées nominales des actions sont potentiellement soumises à des gigues

des systèmes incluant aussi bien les systèmes synchrones (où la *gigue*[2] est assumée nulle) que les systèmes asynchrones (où la *gigue* est assumée non bornée). Par example, les applications multimédia distribuées comme la vidéo-conférence, ou la vidéo-à la demande sont une partie de ces systèmes. De là, la problématique pour ces systèmes consiste à satisfaire les contraintes intra *flux*[3] et inter *flux*. Satisfaire les contraintes intra *flux* consiste à contrôler la *gigue* de chaque média audio ou vidéo dans le but de la maintenir sous un seuil acceptable. Satifaire les contraintes de synchronisation inter *flux* consiste à contrôler par exemple les *gigues* entre les flux audio et vidéo dans le respect d'une valeur seuil (e.g synchronisation du son avec un flux vidéo par rapport au mouvement des lèvres [69]).

Par ailleurs, certaines applications multimédia complexes peuvent être fortement réactives ; elles admettent des comportement préemptifs où un *flux* peut être suspendu momentanément pour être repris ultérieurement. Ce type de comportements permettant d'exprimer les interactions des utilisateurs (i.e, Arrêt, Pause, ou Lecture d'un flux vidéo ou audio), est très important à maîtriser, notamment pour l'élaboration de mécanismes adaptés dans les protocoles de transport.

A d'autres égards, la conception d'applications multimédia distribuées complexes à grande échelle peut être grandement facilitée avec l'apport de modèles formels permettant de spécifier les caractéristiques fondamentales de ces applications, et de vérifier leurs propriétés. Cela permet d'exprimer simultanément avec la même méthodologie des contraintes de sûreté, de temps et d'ordonnancement associées à une application multimédia ou hypermédia. Au fait, ces modèles formelles permettent d'exprimer de manière plus ou moins abstraite les plus importantes exigences de qualité de service des applications multimédia.

Dans ce contexte, plusieurs travaux se sont intéressés ces dernières années au développement des documents multimédia interactifs basés sur différents standards : $SMIL$ [90], $HyTime$[58], $MHEG$ [75] , etc....
En effet, les documents multimédia sont devenus un puissant moyen de communication intégrant différents types de média tels que la vidéo, l'audio, l'animation, le texte et l'image, caractérisés aussi bien par des contraintes temps réel, que par des contraintes de synchronisation strictes. Ces contraintes sont en général difficiles à décrire et à traiter, particulièrement lors de l'élaboration d'architectures multimédia distribuées à grand échelle. Par conséquent, la problématique majeure rencontrée lors de la phase de description est la capacité à définir correctement leurs structures logiques et temporelles de telle sorte que la spécification permette la supervision des contraintes imposées lors de l'exécution de leurs différents composants. La capacité à définir cette structure implique la disponibilité d'un modèle adéquat capable de spécifier de manière précise et clair les contraintes du système.

Etant au fait de la complexité des documents multimédia/hypermédia en terme de contraintes spatiales et temporelles, les méthodes formelles ont été largement utilisées pour la spécification, et la validation des contraintes multimédia (e.g. contraintes de synchronisation, interactions utilisateurs, etc...). Par exemple, les réseaux de Petri [79] et les spécifications algébriques ont été souvent d'un grand intêret. L'extension de ces formalismes pour le traitement des exigences multimédia a été entrepris à travers différents travaux [61][44][53][70][41][14][95]. De plus, l'inter-

2. La *gigue* est l'écrat observé avec la durée nominale.
3. Un *flux* est un processus élémentaire contraint temporellement, et ayant des contraintes de synchronisation avec d'autres *flux*.

pretation de ces modèles avec des paramètres spatiaux et des descripteurs de ressources, rend possible la génération de codes exécutables fonctionnant sur des plate-formes spécifiques [4][59].

Plus particulièrement, les extensions temporisées des réseaux de Petri [74][53][87][95][2] [83][70][14] sont des modèles puissants pour la spécification des systèmes temps réel, principalement car ils permettent de modéliser la concurrence et les contraintes de temps naturellement. Ils offrent une syntaxe graphique permettant une édition accessible et des techniques formelles de validation pour l'analyse de ces documents pour la détection des inconsistances temporelles et des erreurs de synchronisation. Cependant, pour être applicable à la description d'une large frange des systèmes multimédia plus ou moins complexes, le modèle doit être capable de satisfaire le cahier de charge suivant [34][61][44] :

- La capacité de décrire hiérarchiquement les structures des documents aussi bien que les niveaux de synchronisation.
- La capacité de spécifier une temporisation non exhaustive.
- La capacité d'exprimer les synchronisations basées sur les interactions des utilisateurs.
- La capacité d'exprimer un média objet comme une unité logique, dont l'abstraction du contenu permet néanmoins de préserver les relations temporelles se référant à une partie ou la totalité de cet object.
- La capacité d'exprimer un large panel de mécanismes de synchronisations.
- La disponibilité d'une sémantique formelle suppléee par des techniques de vérification pour la détection des inconsistances potentielles présentes dans les documents multimédia complexes.
- et finalement, la simplicité et la manière intuitive des concepts de modélisation fournis à l'utilisateur.

Toutefois, du fait de la complexité des exigences des applications multimédia, les modèles existants et particulièrement ceux basés sur les réseaux de Petri ne traitent qu'un sous ensemble de ces contraintes :

- Les contraintes de temps et d'ordonnancement [94][32] ;
- Les contraintes de synchronisation [41][53][70][49] ;
- Les mécanismes de préemption [42][83].

De plus, la non prise en charge d'un certain nombre de problématiques liées notamment aux mécanismes complexes de gestion des ressources, occulte d'un ascpet important lors de la phase d'analyse. Partant du constat qu'aucun de ces modèles ne permet de saisir de manière exhaustive tous les besoins des applications multimédia, il en résulte que l'analyse menée au moyen de ces méthodologies reste incomplète et ne permet pas d'apporter des solutions efficaces aux problématiques soulevées par ces applications. Par conséquent la définition d'un modèle capable de synthétiser de manière cohérente ces différentes contraintes, serait un premier pas important pour la prise en charge complète des problématiques de ce type d'application.

Un tel modèle une fois défini, pourrait être exploité à la vérification formelle et à l'analyse quantitative des systèmes multimédia, gràce aux techniques d'analyse par énumération permettant de dériver le graphe d'accessibilité correspondant, décrivant les comportements possibles du système modélisé. Plus concrètement, cette méthodologie rendra possible :

- Dans un outil d'édition, la correction des incohérences par formattages temporel et spacial du code source d'un document multimédia ;

– Au niveau du lecteur multimédia, la programmation des scénarios appropriés consistants lorsqu'ils existent, et l'implémentation des mécanismes de délivrance et de préchargement adaptés et plus efficaces [22][12][45] ;
– Au niveau d'un protocole de transport, le contrôle du respect des contraintes de qualité de service, notamment celles liées à la synchronisation des flux [52],...

1.3 Objectif

Nous proposons dans cette thèse une méthodologie de vérification permettant la spécification et l'analyse des systèmes temps réel en général et des applications multimédia en particulier. A cet effet, un nouveau modèle permettant de prendre en charge en plus des systèmes *faiblement synchrones*, d'autres systèmes plus complexes, faisant intervenir les problématiques suivantes :

a) Le modèle doit être capable de modéliser les besoins des médias en terme de ressources, grâce à des mécanismes de préemption adaptés, permettant de résoudre les conflits par la définition d'un ordre de priorité. Plus concrètement, dans les applications multimédia/hypermédia, certaines ressources dites *non critiques* (e.g. canal audio, espace écran) sont facultatives pour l'exécution d'une présentation multimédia mais néanmoins ayant un incident prépondérant sur la qualité du service perçue par l'utlisateur. En outre, la gestion des schémas d'allocation de ces ressources étant très complexe et flexible [4] ; le modèle doit être pourvu de mécanismes d'un pouvoir expressif permettant de spécifier n'importe quel fonctionnement.

b) Le modèle doit pouvoir traiter les mécanismes de préemption de flux. Nous pensons particulièrement au mécanisme de suspension de temps rencontré lors des interactions des utilisateurs, où une présentation peut être suspendue pour une durée indéfinie pour être reprise par la suite.

Au vue de ce constat, nous introduisons une nouvelle extension de $RdPT$ [74] appelée $STPTPN$ (*Preemptive Time Petri Nets with Synchronizing Transitions*) [16] dédiée à la modélisation des exigences multimédia, et utilisant comme modèle de base un $RdPT$ augmenté de mécanismes spécifiques et pourvu de sémantiques adaptées pour cet objectif :

1. Considérant la problématique énoncée en (a), les ressources *non critiques* sont modélisées par des places spéciales en adaptant la sémantique de la règle de tir. Ainsi, puisque la disponibilité de ces ressources ne conditionne pas le tir d'une transition (i.e., la transition peut tirer même si les ressources sont non disponibles). Pour dénoter des différentes évolutions possibles du système, nous introduisons trois types d'événements associés au tir d'une transition t :
   – Un *événement* $fort$ noté t, dénotant que t est tirée, avec acquisition de toutes ses ressources *non critiques* ;
   – Un *événement de violation passive*, noté t^* dénotant que t est tirée sans acquérir l'une des ressources *non critiques* requises ; et
   – Un *événement de violation active* noté $^*t$, indiquant que t a été tirée, en procédant à la violation des ressources *non critiques* manquantes à partir des transitions moins prioritaires qui les détenaient.

4. Dans le contexte des documents multimédia, les fonctionnements diffèrent d'un lecteur à un autre.

Pour ce faire, deux nouveaux mécanismes sont introduits :

– L' *hyperarc préempteur* [14] est introduit pour déterminer quel événement doit être associé lors du tir d'une transition.

– L' *arc* de *privation de ressource [20]* est introduit pour spécifier un mécanisme de préemption de ressources entre deux transitions conflictuelles.

2. Pour conditionner l'occurrence des événements des *interactions des utilisateurs*, en fonction de la disponibilité des ressources *non critiques,* un arc appelé "*hyperarc de franchissement[20]* est introduit.

3. Dans le modèle que nous proposons, un chronomètre [46] est associé à chaque transition, permettant d'initier, d'arrêter, et de reprendre le décompte du temps. Ces actions sont inférées grâce à utilisation de la sémantique des arcs standards et des arcs inhibiteurs [26]. Ce mécanisme permet de modéliser par exemple la suspension de l'exécution d'un média suite à l'interaction d'un utilisateur et sa reprise ultérieure.

4. Pour modéliser les différents schémas de synchronisation comme ceux introduits dans les réseaux de petri à flux $TSPN$ [53], nous considérons le tir simultané et indivisible d'un ensemble de transitions (appelé ici $rendez-vous$), orchestré par différentes règles de synchronisation.

5. Finallement, dans le but de spécifier la notion de priorité entre rendez-vous concurrents, nous appliquons le mécanisme de "*priorité en temps réel" [11][10][21]* permettant de résoudre le *non déterminisme* entre rendez-vous *inclusifs* franchissables pour les mêmes dates ; un rendez-vous r_1 est plus prioritaire que le rendez-vous r_2 s'il l'inclut strictement [5]. En clair, lorsque r_1 et r_2 sont franchissables pour la même date, seul le rendez-vous r_1 est réalisé à cette date car son schéma de synchronisation intègre en plus, les transitions synchronisant pour r_2.

Toutefois, la proposition d'un tel cadre général serait inutile, sans sa dotation de techniques d'analyses permettant l'exploration de son espace d'état et de procédures permettant l'évaluation de ses performances. Cet aspect nécessite de construire le graphe d'accessibilité préservant les propriétés linéaires du système. A cet effet, la technique la plus usitée pour les $RdPT$ [74] est la méthode du "*graphe des classes*"[32]. Cette dernière représente chaque noeud du graphe par une classe formalisée par le couple (M, D), où M est le marquage du réseau et D est un système de contraintes dénotant le domaine des instants de tirs encodé sous forme [6] DBM (*Difference Bound Matrix*)[54].

Cependant, l'application de la méthode du graphe des classes pour l'énumération de l'espace d'état d'un $STPTPN$ soulève plusieurs problématiques majeures :

- L'introduction de la sémantique des chronomètres soulève une problématique non encore résolue par des algorithmes polynômiaux. Plus concrètement, cette sémantique rend impossible l'expression dans le cas général du système D sous forme DBM ; Au fait le système D est donné

5. Les transitions synchronisant pour r_2 sont partie prenantes dans r_1.

6. Cette forme exprime chaque inéquation soit par $\underline{t_i} - \underline{t_j} \preceq c_{ij}$ ou bien par $c_{i\bullet} \preceq \underline{t_i} \preceq c_{\bullet i}$ avec $c_{ij} \in Q \cup \{\infty\}$, $c_{\bullet i} \in Q^+ \cup \{\infty\}$ et $c_{i\bullet} \in Q^+$

par la conjonction de deux sous-systèmes, l'un ayant une forme *DBM* noté $\widetilde{D}$ et l'autre de forme quelconque dite *polyédrique* noté $\widehat{D}$. Par conséquent, les tests de franchissement et d'équivalence sont déterminés par la résolution en un temps exponentiel du système D de forme général polyédrique. Cependant, des approches ont été proposées exprimant la sur-approximation *DBM* de D obtenue en éliminant les contraintes $\widehat{D}$ [42][83][72]. Ces techniques permettent de maintenir la complexité de calcul au même degré que pour un $RdPT$, mais en produisant un graphe grossier pouvant inclure des comportements additionnels non induits par la spécification de base. Pour ce faire, des heuréstiques agissant sur le graphe ainsi obtenu [42] ont été proposées pour détecter ces comportements afin de pouvoir les éliminer lors de la phase d'analyse. Toutefois, cette procédure n'est pas sans induire un coût supplémentaire qui dans certains cas pourrait être prohibitif.

- L'introduction des mécanismes de synchronisation en associant les sémantiques temporelles *faible* et *forte* dans un même modèle, génère des classes inconsistantes dont l'identification est un prérequis pour l'énumération de l'espace d'état. De plus, l'addition du concept de rendez-vous dans le modèle, en plus du concept standard de transition, rend complexe la formalisation des contraintes de franchissement. Au fait, l'expression de ces dernières devrait permettre inférer aussi bien les contraintes des transitions que celles des rendez-vous.
- L'introduction du mécanisme de "*la priorité en temps réel*" rend complexe la formalisation du test de franchissement, car impliquant différentes règles de tir. Toutefois, l'application de ce mécanisme de priorité, permet de résoudre le non déterminisme entre rendez-vous concurrents, engendrant un graphe plus compact par suppression des sémantiques d'entrelacement.

Etant au fait de ces problématiques, nous proposons d'abord une méthode d'énumération exacte basée sur la formulation de chaque classe accessible E par un couple (M, D) où D est un ensemble de formules de *forme polyédrique* déterminant les contraintes des transitions sensibilisées. A partir de là, les tests de consistance, de franchissement ainsi que le calcul des classes accessibles nécessitent l'extension du système D à un système additif capturant les contraintes des rendez-vous. Donc, le modèle évolue ici, en franchissant à chaque pas un rendez-vous impliquant le tir simultané de ses transitions synchronisantes et générant un ensemble d'événements étiquetant l'arc du graphe. Cependant, comme la complexité de calcul d'une classe exacte d'un $STPTPN$ est exponentielle aux nombres de variables, l'approche en question ne peut être efficace pour l'analyse des modèles complexes car s'avérant très couteuse en temps de calcul et en espace mémoire. Pour palier à cet inconvénient, la sur-approximation *DBM* peut s'avérer une solution alternative pour l'énumération et l'analyse de l'espace d'état des $STPTPN$ complexes et permet un algorithme efficace en terme de complexité de calcul comparé à la première solution. Pour ce faire, de la même manière que pour les $RdPT$ à chronomètres, nous envisageons une approche d'énumération basée sur le relâchement de toutes contraintes sous forme polyédriques pour ne garder que celles sous forme *DBM,* $\widetilde{D}$. Cela permet de déterminer efficacement des conditions suffisantes pour les tests de consistance, de franchissement et enfin d'équivalence, et de calculer un graphe sur-approximé moins riche mais cependant moins couteux. A cet effet, le calcul d'une classe sur-approximée notée $\widetilde{E}$ pourrait être obtenue en appliquant l'algorithme de $Floyd - Warshall$, dont la complexité est estimée dans le meilleur des cas à $o((m+l)^3)$ où l et m sont respectivement le nombre de transitions sensibilisées et le nombre de rendez-vous fortement sensibilisés pour la classe. Toutefois,

l'application de cet algorithme reste encore relativement coûteuse. Pour y remédier, nous proposons une technique permettant de réduire la complexité de calcul d'une classe accessible en un temps polynômial d'ordre deux, estimé à $o(l^2 + l \times m + m)$ [15][17]. Au fait, cet algorithme permet d'éliminer les calculs redondants dans le même esprit de ce qui a été entrepris auparavant pour les $RdPT$[37][94].

Par ailleurs, l'approche de sur-approximation que nous proposons permet de calculer l'espace d'état *exact* d'un $STPTPN$ lorsque ce dernier ne contient pas d'arc inhibiteur, et un espace *sur-approximé* dans le cas contraire. Chaque noeud du graphe est une paire $(M, \widetilde{D})$ où $\widetilde{D}$ correspond au sous-système de contraintes DBM relatif aux transitions sensibilisées, à partir du quel nous pouvons déduire celui relatif aux rendez-vous noté DR. Ce dernier est exprimé par une matrice des bornes particulière permettant de réaliser le test de franchissement de manière efficace et flexible. Le processus d'énumération munie d'une relation d'équivalence basée sur l'égalité des classes permet de construire le graphe contractant l'espace d'état [7] du $STPTPN$. Cette contraction permet de préserver les propriétés linéaires du modèle. Toutefois, en raison de l'introduction des sémantiques des chronomètres et de synchronisation, les graphes exact et sur-approximé obtenus peuvent être infinis même si le réseau est borné, et ce au contraire d'un $RdPT$ dont le graphe des classes est prouvé fini si le modèle de base est borné.

Les graphes ainsi calculés pourraient être utilisés si finis, pour l'analyse des propriétés générales du système, plus précisément des applications multimédia pour lesquelles le $STPTPN$ est dédié. De plus, l'introduction de la classification d'événements (i.e, *fort, de violation active et passive*) permet de raffiner l'analyse et d'inférer de nouvelles propriétés liées à la fiabilité et à la correction du système modélisé. D'ailleurs, nous discutons dans cette thèse de l'extension aux contraintes ressource du concept standard de consistance des documents multimédia, auparavant introduit dans [85]. Nous inférons de cela un nouveau paradigme que nous appelons "*consistance qualitative*" [19][20].

Par ailleurs, concernant les propriétés temps réel, elles pourront être déduites du graphe grâce à l'application d'une procédure calculant des sur-approximations des temps minimaux et maximaux de n'importe quelle séquence de franchissement. Ces informations permettent l'analyse quantitative et de là, l'évaluation des performances du système modélisé.

1.4 Contribution

Les contributions scientifiques des travaux entrepris durant cette thèse, peuvent être énumérées comme suit :

1. Proposition d'un modéle formel riche basé sur les *RdPT (*appelé $STPTPN$), permettant de concentrer plusieurs sémantiques. Ce modèle rend possible la modélisation des systèmes temps réel complexes et en particulier des systèmes multimédia.
2. Proposition d'un processus de modélisation des documents multimédia par l'exploitation du modéle défini. Pour ce faire, les fonctionnements des principaux lecteurs[27][81] ont été étudiés et considérés.
3. Proposition d'une approche exacte pour la contraction de l'espace d'état d'un $STPTPN$. Cette contraction dite technique du *graphe des classes* nécessite pour ce cas la représentation

7. Dans le cas d'une sur-approximation *DBM,* le graphe obtenu inclut l'espace d'état du *STPTPN.*

de chaque classe E par une paire (M, D) où D est un système de contraintes sous forme polyédrique. La complexité du calcul de chaque classe est exponentielle au nombre de transitions et rendez-vous sensibilisés pour la classe.

4. Proposition d'une approche implémentant la sur-approximation de l'espace d'état d'un $STPTPN$. Cette technique est basée sur la représentation de chaque classe par le sous système de contraintes sous forme DBM $\widetilde{D}$. Le graphe construit est exact si le modèle ne contient pas d'arcs inhibiteurs et peut être sur-approximé dans le cas contraire ; préservant un sous ensemble de propriétés. La complexité du calcul de chaque classe est polynômiale d'ordre deux au nombre des transitions sensibilisées.
5. Proposition d'une technique pour le calcul de la réponse du temps borné (i.e, temps minimal et maximal d'une séquence). Les valeurs calculées sont exactes si le graphe ne contient pas d'arcs inhibiteurs et sur-approximées sinon
6. Finallement, exploitation du graphe pour l'analyse de la cohérence des documents multimédia. Le concept de *consistance qualitative* est défini étendant celui introduit dans [85] à la détection des conflits de ressources et à l'amélioration de la qualité des présentations multimédia.

1.5 Organisation du document

La présente thèse est organisée comme suit : Le *Chapitre* 2 revient sur les définitions des réseaux de Petri, nous y énumérons ses principales extensions temporisées en mettant en exergue certaines des sémantiques introduites telles : la priorité, la préemption, et la synchronisation.

Dans le *Chapitre* 3, nous explorons une extension des $RdPT$ dédiée à la spécification des applications multimédia à large echelle. Pour ce faire, prenant en considération les contraintes de temps et de synchronisation et les besoins en termes de ressources de ces systèmes, nous définissons progressivement un cadre général usant de mécanismes diverses tels que la priorité, la préemption de ressource, l'inhibition temporelle, et enfin la synchronisation, et ce dans le but de modéliser le plus correctement et le plus fidèlement possible ces besoins. Le *Chapitre* 4 présente la syntaxe et la sémantique fomelles du modèle proposé, et finit par le comparer avec des modèles existants. Nous montrons aussi comment exploiter ce modèle pour la modélisation des contraintes des documents $SMIL$.

Le *Chapitre* 5, introduit deux approches pour le calcul de l'espace d'état d'un $STPTPN$. La première basée sur la représentation polyédrique calculant le graphe exact et la seconde implémentant sa sur-approximation DBM. Un exemple est donné pour expliciter les approches proposées. Le *Chapitre* 6 montre comment exploiter les graphes obtenus pour l'analyse qualitative et quantitative du système modélisé. Nous discutons d'une approche pour la vérification de la *consistance* des documents multimédia [20] ; un nouveau paradigme est introduit appelé "*consistance qualitative*" permettant d'étendre les concepts introduits dans [84][86][85], à la détection et la gestion des conflits de ressources.

Enfin, deux annexes sont rajoutées, l'une contenant les preuves formelles des propositions et théorèmes énoncés dans cette thèse. L'autre présente des résultats primaires des mise en oeuvres des algorithmes proposés, expérimentés sur des modèles réduits. Une comparaison non exhaustive avec d'autres approches est menée.

Chapitre 2

Généralités sur les réseaux de Petri

2.1 Introduction

Les réseaux de Petri (RdP) [78] sont des outils graphiques et mathématiques permettant de modéliser le comportement dynamique des systèmes à événement discret comme les systèmes manufacturiers, les systèmes de télécommunications, les réseaux de transport, ... etc.

Leur représentation graphique permet de visualiser d'une manière naturelle le parallélisme, la synchronisation, le partage de ressources, les choix (conflit), ... etc. Leur représentation mathématique permet d'établir les équations d'états, à partir des quelles il est possible d'apprécier les propriétés du modèle et de les comparer avec le comportement du système modélisé. Le modèle de base des réseaux de Petri [79] permet donc d'étudier les propriétés logiques des systèmes, c'est-à-dire les propriétés qui dépendent de l'ordre des événements, mais sans faire référence aux instants où ils occurrent.

Depuis l'introduction des RdP, de nombreuses extensions ont été apportées. Ces extensions ont été effectuées soit pour améliorer la concision du modèle (il s'agit des réseaux de Petri colorés) [63], soit pour augmenter la puissance de spécification de cet outil (les réseaux de Petri temporisés [80], les réseaux de Petri temporels [74], les réseaux de Petri stochastiques [29], ...)

Dans notre cas, nous nous intéressons aux extensions temporelles déterministes du modèle à savoir les réseaux de Petri temporels $RdPT$ [74]. Ces extensions ont été établies dans le but de prendre en compte le facteur temps dans l'évolution du modèle.

Nous consacrons la première partie du présent chapitre aux réseaux de Petri autonomes ; Leur définition et leurs propriétés logiques et dynamiques les plus importantes sont rappelées. Une discussion sur les méthodes pour vérifier et étudier leurs comportements et leurs propriétés sera menée.

La deuxième partie se verra consacrée à la présentation d'un modèle où le temps prends la forme d'un intervalle temporel. Une attention particulière sera alors portée au modèle défini par Merlin ($T - RdPT$ [74]). Un rappel des définitions, des règles de fonctionnement et des méthodes d'analyses de cette extension des RdP fera l'objet de cette seconde partie. Enfin, nous terminons le présent chapitre par présenter quelques extensions de $RdPT$ élargies à certains mécanismes permettant de capturer des sémantiques plus complexes. Nous focalisons notamment sur des modèles introduisant des mécénismes de priorité [10], de préemption [67] et enfin de synchronisation [53].

2.2 Les réseaux de Petri

Les réseaux de Petri ont été introduits par *Carl Adam Petri* dans sa thèse intitulée *Communication avec Automates* en **1962** [78].

2.2.1 Définitions de base

Condition et événement :

- Une **condition** et la description de l'état d'une ressource du système modélisé, elle peut être vraie comme elle peut être fausse.
  - une machine est au repos.
  - une machine est en réparation.
  - une commande est en attente.
- Un **événement** est une action qui se déroule au sein du système et dont la réalisation dépend de l'état du système.
  - début de traitement sur une machine.
  - panne sur une machine.
  - début de traitement d'une commande.

Place et transition :

Dans le formalisme des réseaux de Petri :

- une place modélise une condition ou une ressource du système.
- une transition modélise un événement (action, synchronisation).

Place d'entrée et place de sortie :

Chaque transition peut être reliée à des places d'entrée et des places de sortie. Les places d'entrée sont les places d'où sont issues les arcs orientés vers la transition ; les places de sortie sont les places où aboutissent les arcs orientés issus de la transition.

Transition source et transition puits :

Une transition source est une transition qui ne comporte aucune place d'entrée ; une transition puits est une transition qui ne comporte aucune place de sortie.

Définition d'un réseau de Petri :

Un réseau de Petri peut être assimilé à un système composé de deux parties distinctes : Une partie statique (structurelle), qui se constitue d'un graphe orienté, comportant :

- Un ensemble fini de places, symbolisées par des cercles.
- Un ensemble fini de transitions, symbolisées par des tirets ou rectangles.
- un ensemble d'arcs orientés qui assurent la liaison d'une place à une transition ou d'une transition à une place.

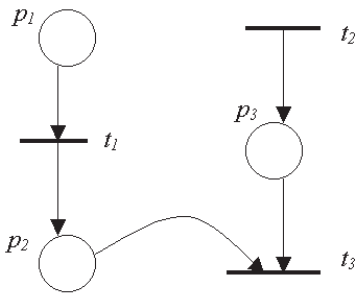

FIGURE 2.1 – Exemple d'un RdP

La *Figure* 2.1 représente un exemple de réseaux de Petri avec p_1, p_2, p_3 (resp. t_1, t_2, t_3) les places (resp. les transitions) du réseau. p_1 est la seule place d'entrée de la transition t_1 ; p_2 et p_3 sont les places d'entrées de la transition t_3 ; p_2 est aussi place de sortie de la transition t_1 ; p_3 est une place de sortie de la transition t_2 ; t_2 est une transition source et t_3 est une transition puits.

Définition 1 ***(Réseau de Petri)****. Un RdP est un quadruplet* $R = (P, T, Pre, Post)$*, où :*

- P*, est un ensemble fini non vide de places ;*
- T*, est un ensemble fini non vide de transitions ;*
- $Pre : P \times T \longrightarrow N$*, est l'application d'incidence avant, correspondant aux arcs directs reliant les places aux transitions ;*
- $Post : P \times T \longrightarrow N$*, est l'application d'incidence arrière, correspondant aux arcs directs reliant les transitions aux places.*

Graphiquement, un arc relie une place p à une transition t (resp. une transition t à une place p) si $Pre(p,t) \neq 0$ (resp. $Post(p,t) \neq 0$).

2.2.2 Marquage d'un *RdP*

Le marquage d'un *RdP* est précisé par la présence à l'intérieur des places d'un nombre fini (positif ou nul), de marques ou de jetons. Une place est donc vide ou marquée. Lorsque la place représente une condition logique (e.g. : machine au repos, ...), la présence d'un jeton indique que cette condition est vraie ; fausse dans le cas contraire. Lorsque la place représente une ressource (au sens le plus large, e.g. un stock, ...), elle peut contenir plusieurs jetons (e.g. le nombre de pièces en stock). Ainsi, le marquage initial d'un *RdP* correspond à la distribution initiale des jetons dans chacune des places du *RdP*, précisant l'état initial du système retenu pour l'analyse.

Définition 2 ***(Réseau de Petri marqué).*** *Un RdP marqué est un couple* $< R, M >$ *où :*

- R *est un réseau de Petri ;*
- M *est une application qui associe à chaque place* p *de* R *un nombre de marques* p *à* $M(p)$*. $M : P \longrightarrow N$. On désigne par $M(p)$ le nombre de marques (jetons) contenues dans la place p.*

Cette partie statique est complétée par le marquage initial noté M_0. La partie dynamique du réseau consiste alors à faire évoluer ce marquage, c'est ce que nous allons voir dans le paragraphe qui suit.

2.2.3 Règles de fonctionnement

Une transition t est dite sensibilisée (validée, franchissable ou encore tirable) par le marquage M, si : $\forall p \in P, \quad M(p) \succeq Pre(p,t)$. Par la suite, nous notons par *Ts(M)* l'ensemble des transitions sensibilisées pour M.

Le tir (le franchissement) d'une transition t a pour conséquences :

- De retirer $Pre(p,t)$ jetons de chaque places d'entrée p de la transition t.
- De rajouter $Post(p,t)$ jetons à chaque place de sortie p de la même transition t.

Si le marquage avant le tir de la transition t est M, son tir conduira donc au marquage M' vérifiant : $\forall p \in P, M'(p) = M(p) - Pre(p,t) + Post(p,t)$.

L'ensemble des marquages accessibles en partant du marquage M_0 en franchissant une séquence de transitions S sera noté $Acc(R)$. Si $M_n \in Acc(R)$, alors il existe au moins une séquence de transitions $S = (t_1, ..t_n)$ franchissable qui permet de passer du marquage M_0 au marquage M_n.

Remarque : Lorsqu'une transition est validée, cela n'implique pas qu'elle sera immédiatement franchie ; cela ne représente qu'une possibilité de franchissement ou d'évolution du RdP. Pour les RdP il y a un seul franchissement à la fois. Ces remarques impliquent que lorsque plusieurs transitions sont sensibilisées dans un même marquage (les transitions sont franchissables en parallèle), l'ensemble des marquages suivants sera obtenu en considérant toutes les possibilités de franchissement des transitions les unes après les autres. Ainsi, n transitions sensibilisées conduiront à n marquages permettant ainsi d'étudier l'ensemble des comportements possibles. Cependant, ce mode de fonctionnement conduit à une explosion combinatoire des marquages pour les RdP complexes.

2.2.4 Transitions conflictuelles et nouvellement sensibilisées

- Deux transitions t_1 et t_2 sensibilisées pour M sont dites en conflit pour le marquage M, si la condition suivante est vérifiée : $\exists p \in P, \quad Pre(p,t_1) + Pre(p,t_2) \succ M(P)$.
  Le tir de l'une des deux transitions à partir du marquage M désensibilisera l'autre.
- Soit le tir de la transition t à partir du marquage M pour aboutir au marquage M'. Une transition t' sensibilisée pour M' est dite *nouvellement sensibilisée* pour M' si elle est sensibilisée seulement dans M', oubien elle est sensibilisée pour M et M' et en conflit avec la transition franchie t pour le marquage M.

$$\begin{cases} (t' \in Ts(M')) \wedge (t' \notin Ts(M)) & \vee \\ (t' \in Ts(M) \cap Ts(M')) \wedge (\exists p \in P, \quad Pre(p,t) + Pre(p,t') \succ M(P)) \end{cases}$$

Autrement, la transition t' est dite *anciennement sensibilisée* ou *persistante*.

2.2.5 Propriétés des *RdP*

Les principales propriétés des RdP peuvent être classées en deux groupes :

· Les propriétés structurelles.

· Les propriétés dynamiques.

Les propriétés structurelles sont indépendantes du marquage initial et sont liées à la topologie du réseau. Leur analyse repose essentiellement sur les techniques d'algèbre linéaire et leur but est de bâtir une passerelle entre la structure du réseau étudié et son comportement.

Les propriétés dynamiques, quant à elles dépendent du marquage initial et sont liées à l'évolution de ce dernier. Leur vérification nécessite bien souvent la construction du graphe de marquages accessibles. Elles permettent d'apporter des réponses aux questions concernant l'accessibilité d'un marquage particulier, la bornitude, la vivacité, ...

Propriété de bornitude :

Cette propriété permet de caractériser la possibilité pour une place d'accumuler une quantité bornée ou non de marques au cours de l'évolution du réseau.

- Une place p est bornée ou *k-bornée* pour un marquage initial M_0, s'il existe un entier naturel k tel pour tout marquage accessible à partir de M_0, le nombre de marques de p reste inférieur ou égal à k, i.e. : $\exists k \in N \ / \quad \forall\ M \in Acc(R),\ M(p) \preceq k$.
- Une place p est dite binaire si elle est *1-bornée*.
- Un *RdP* est *k-borné* pour un marquage initial M_0, si toutes les places sont k-bornées pour M_0, et on note : $\exists k \in N \ / \quad \forall\ M \in Acc(R), \forall\ p \in P,\ M(p) \preceq k$
- Un *RdP* est dit *Sauf*, s'il est *1-borné*.

Réseau vivant :

Un *RdP* marqué est vivant si et seulement si pour tout marquage accessible M ($M \in Acc(R)$), et pour toute transition t, il existe une séquence de franchissement partant de M et contenant t. La vivacité d'un réseau garantit le franchissement de toute transition quelque soit le marquage atteint. La propriété de vivacité est une propriété forte, souvent difficile à vérifier.

Réseau bloqué :

Un *RdP* est bloqué si aucune transition n'est validée pour un marquage donné M ; M est appelé marquage puits.

Réseau ré-initialisable :

Un *RdP* est réinitialisable pour un marquage initial M_0, si pour tout marquage accessible M, il existe une séquence de franchissement de transitions telle que l'on puisse accéder à M_0 (M_0 est alors appelé *état d'accueil*).

Remarque : Un *RdP* a un état d'accueil M' pour un marquage initial M_0, si pour tout marquage accessible, il existe une séquence de franchissement S tel que M' est accessible depuis M en franchissant S.

2.2.6 Graphe de couverture

La vérification des propriétés données ci-dessus se fait généralement en établissant le *graphe de couverture* qui est composé de nœuds qui correspondent aux marquages accessibles, et d'arcs correspondant aux franchissements de la transition qui fait passer d'un marquage à un autre. Le nombre de nœuds dans ce graphe est fini.

La construction du graphe de couverture permet de décider si un *RdP* est borné : on dit que la propriété de réseau borné est décidable. Dans ce cas, le graphe sera appelé le graphe des marquages accessibles et noté $GAcc(R, M_0)$. A partir de ce graphe, on peut vérifier toutes les autres propriétés (vivacité, accessibilité d'un marquage, ...). Cependant, pour un *RdP* non borné, établir l'arbre de couverture ne suffit pas pour résoudre le problème d'accessibilité et le problème de vivacité. Ces deux problèmes sont également décidables, mais par des algorithmes plus compliqués. Malgré ceci, le graphe de couverture reste le seul outil pour vérifier les propriétés dynamiques d'un *RdP* pour un marquage initial donné. Cette approche d'analyse est générale, elle est applicable à toute classe de réseaux. Mais, ses limites sont liées à l'explosion combinatoire du nombre d'états. De plus, à partir de ce graphe on peut savoir s'il existe des fonctionnements spécifiques comme par exemple l'existence d'un fonctionnement répétitif stationnaire. L'approche qui est basée sur le graphe de couverture pour étudier les *RdP* constitue l'analyse énumérative.

2.2.7 Exemple

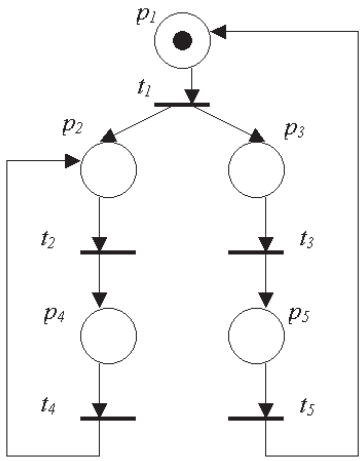

FIGURE 2.2 – Evolution d'un RdP

La *Figure* 2.2 représente un réseau de Petri $R = (P, T, Pre, Post, M_0)$ tel que :
$P = \{p_1, p_2, p_3, p_4, p_5\}$, $T = \{t_1, t_2, t_3, t_4, t_5\}$, $n =\mid T \mid = 5$, m $=\mid P \mid = 5$.
Le marquage initial M_0 est donné par le vecteur $M_0 = [1, 0, 0, 0, 0]$, où seule t_1 est sensibilisée par M_0.

- Le franchissement de la transition t_1 à partir de M_0 conduit au marquage M_1, on note : $M_0 \xrightarrow{t_1} M_1$ avec M_1=[0, 1, 1, 0, 0], où t_2, t_3 sont sensibilisées par M_1.
- Le franchissement de la transition t_2 à partir de M_1 conduit au marquage M_2, on note : $M_1 \xrightarrow{t_2} M_2$ avec M_2=[0, 0, 1, 1, 0], où t_3, t_4 sont sensibilisées par M_2.
- Le franchissement de la transition t_3 à partir de M_1 conduit au marquage M_3, on note :$M_1 \xrightarrow{t_3} M_3$ avec M_3=[0, 1, 0, 0, 1], où t_2 est sensibilisée par M_3.
- Le franchissement de la transition t_3 à partir de M_2 conduit au marquage M_4, on note : $M_2 \xrightarrow{t_3} M_4$ avec M_4=[0, 0, 0, 1, 1], où t_4, t_5 sont sensibilisées par M_4, et ainsi de suite,...

L'ensemble des marquages accessibles obtenu pour le *RdP* de la *Figure* 2.2 est donné par : $Acc(R) = \{M_0, M_1, M_2, M_3, M_4\}$ avec, M_0=[1, 0, 0, 0, 0], M_1=[0, 1, 1, 0, 0], M_2=[0, 0, 1, 1, 0], M_3=[0, 1, 0, 0, 1], M_4=[0, 0, 0, 1, 1]. Le graphe de couverture est illustré par la *Figure* 2.3.

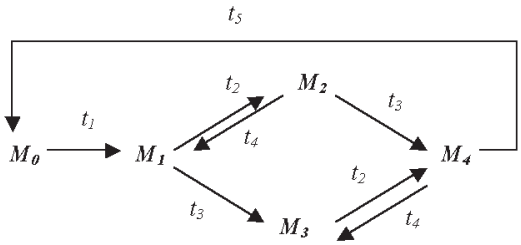

FIGURE 2.3 – Graphe de couverture

2.2.8 Extension des RdP aux arcs inhibiteurs

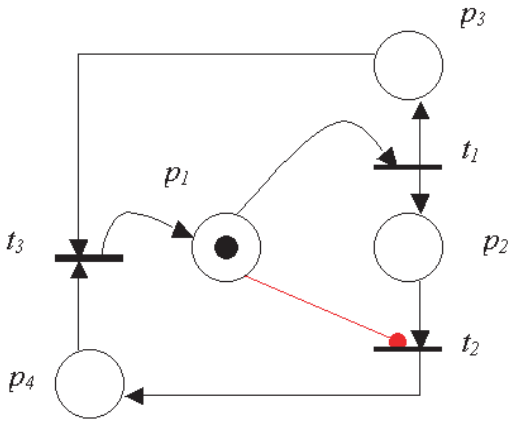

FIGURE 2.4 – RdP avec arc inhibiteur

Dans un *RdP* ordinaire une transition est validée lorsque chacune de ses places d'entrée contient au moins un jeton. Cette règle ne permet pas de conditionner le franchissement d'une transition à l'état vide de l'une de ses places d'entrées. Les *RdP* à arcs inhibiteurs, permettent de réaliser ce test.

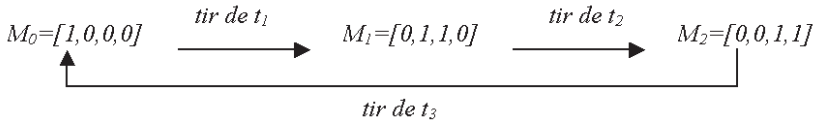

FIGURE 2.5 – Graphe des marquages du RdP à arc inhibiteur

Définition 3 *(RdP avec arcs inhibiteurs) Un RdP à arcs inhibiteurs est un couple* $<R, Inh>$ *où :*

- R : *est un RdP marqué ;*
- $Inh : P \times T \longrightarrow N$, *est la fonction arc inhibiteur.*

La représentation graphique des réseaux à arcs inhibiteurs ne diffère pas de celle des réseaux de Petri que par le fait que si $Inh(p,t) \neq 0$, on aura un arc de la place p vers la transition t représenté par une flèche avec un cercle à l'extrémité. La *Figure* 2.4 représente un exemple de réseaux de Petri avec arcs inhibiteurs. Dans cela , on a un arc inhibiteur reliant la place p_1 à la transition t_2 tel que $Inh(p_1, t_2) = 1$. Par conséquent, on dit que :

– t_2 est sensibilisée par le marquage M, si et seulement si $M(p_2) = 1$;
– t_2 est validée par le marquage M, si et seulement si : $M(p_2) = 1$ et $M(p_1) = 0$;
– t_2 est inhibée, si et seulement si : $M(p_2) = 1$ et $M(p_1) = 1$.

Pour mieux comprendre le fonctionnement d'un réseau à arcs inhibiteurs, on a le graphe des marquages suivant correspondant au réseau de la *Figure* 2.4 :

A l'état initial déterminé par M_0, seule la transition t_1 est validée, son franchissement conduit a un nouveau marquage M_1 tel que, $M_1(p_1) = 0$ et $M_1(p_2) = 1$. Dans ce cas la transition t_2 devient validée par ce marquage ; son tir conduit à un nouveau marquage et ainsi de suite, comme le montre le graphe des marquages de la *Figure* 2.5.

2.3 Extensions temporisées des réseaux de Petri

2.3.1 Etat de l'art

Si l'analyse qualitative des systèmes non temporisés est assez bien maîtrisée, la prise en compte et le traitement des valeurs explicites du temps soulèvent encore de nombreux défis . De plus, comme les comportements des systèmes ne dépendent pas seulement de l'ordre pour lequel ils sont réalisés mais aussi des instants pour lesquels ils se produisent. Par conséquent la modélisation et l'analyse de tels systèmes soulèvent de nouvelles problématiques dans l'informatique d'aujourd'hui. Dans ce contexte particulier, plusieurs modèles théoriques ont été proposés et étudiés parmi lesquels, les automates temporisés [24], et les différentes extensions temporisées des réseaux de Petri [2][14][31][65][95][87][70][83][89]. D'ailleurs, les réseaux de Petri furent parmi les premiers modèles théoriques étendus au facteur temps, et les premières approches ont été proposées dans [74][80]. Par exemple, dans le modèle de *Merlin* aussi appelé *RdPT* (*Réseau de Petri temporel*) [74] ; un intervalle $[tmin, tmax]$ est associé à chaque transition. Si une transition t est sensibilisée à la date δ, alors t peut être tirée instantanément après $\delta + tmin$ et pas avant $\delta + tmax$, en assumant que t reste continuellement sensibilisée. Le modèle de *Merlin* permet d'exprimer ainsi, aussi bien des spécifications de délai ou de durée de tir. Ses domaines d'applications sont larges, et il a été principalement utilisé pour l'étude des protocoles et des problèmes sous-jacents. Cependant, en raison de la densité du temps, l'analyse exhaustive du modèle fut ardue en raison de la non finitude de son graphe d'atteignabilité. D'ailleurs, les *RdPT* furent le support théorique des premières techniques d'énumération des espaces d'état des systèmes temporisés. Pour cet effet, l'approche principale pour la construction de l'espace d'état d'un *RdPT* est la méthode des graphes des classes [32][73]. Les noeuds du graphe sont des ensembles d'états [1] du *RdPT*, et ses arcs sont étiquetés avec les noms des transitions. De plus, cette approche nécessite l'encodage du système D sous une forme équivalente dite forme *DBM (Difference Bound Matrix)*, dont le calcul se fait en appliquant un algorithme du plus court chemin [2]. Toutefois, *Lilius* [66] a raffiné cette technique de telle sorte qu'il soit possible d'appliquer des procédés de réduction par ordre partiel, auparavant développées pour les systèmes non temporisés. Des travaux plus récents [94][37] ont proposé des techniques

1. Une classe d'état est une paire (M, D) consistant en un marquage M et un système de contraintes D définissant l'espace de tir.

2. Le plus célèbre de ces algorithmes est celui de $Floyd/Warshall$ dont la complexité est estimé à $o(l^3)$ où l étant le nombre de variables présentes dans le système.

permettant de réduire la complexité du calcul d'une classe à $o(l^2)$, offrant aussi la possibilité de calculer les temps minimaux et maximaux d'une séquence en un temps estimé à $o(k \times l)$ où k est la longueur de la séquence. En assumant ces procédés d'énumération, on peut constater que si L est le langage accepté par le graphe et L' est le langage non temporisé accepté par le $RdPT$, alors on a $L = L'$ et cela ne préserve que les propriétés linéaires du réseau.

Dans le but de vérifier les propriétés de branchements, une approche alternative, dite méthode du "*graphe des classes atomiques*" a été proposée définissant une nouvelle équivalence de classes permettant la vérification avec des logiques telles que (CTL) [97]. Par la suite, *Berthomieu* et $Vernadat$ ont proposé dans [48] une construction similaire applicable à un large ensemble de réseaux, produisant un graphe plus compact que celui obtenu dans [97]. Cette dernière proposition fut encore discutée par *Boucheneb et autres* qui proposèrent de contracter davantage le graphe obtenu en réduisant sa complexité de calcul [38]. Dans le même contexte, d'autres travaux ont proposé des techniques [67] permettant de traduire les $RdPT$ vers des automates temporisés [25], et ce dans le but d'exploiter les outils et les algorithmes précédemment définis pour l'analyse des propriétés de branchement des automates temporisés [92] [25] [98].

Toutefois, malgrè leurs succès, les $RdPT$ restent inadaptés et incapables de modéliser un certain nombre de systèmes aux comportements complexes. Par conséquent, pour accroître leur puissance d'expression, des travaux ont cherché à associer les intervalles de temps aux autres emplacements du réseau, sur les places [65] [89] ; sur les arcs [95] [53], et en considérant différentes sémantiques du temps (faible et forte) [50]. Ces nouveaux modèles ont permi d'accroître l'expression temporelle des $RdPT$, mais en posant de nouvelles problématiques quand-t-à leurs analyses. La plupart de ces modèles souffrent du manque d'algorithmes efficaces pour le calcul de leur espace d'état et pour l'analyse de leur propriétés temps-réel. Par exemple, dans [95], $Walter$ définit un modèle appelé $TLPN$ *(Time Link Petri Net)*, associant des intervalles aux arcs ; ainsi, un jeton dans la place p peut être consommé pour le le tir d'une transition t seulement si le temps écoulé dans la place est inclus dans l'intervalle de l'arc (p, t).

Afin d'établir une comparaison exhaustive entre modèles, $Cerone$ *et autres* [50] ont proposé une première tentative, définissant une extension appelée "*Statically Timed Petri Net*" associant des intervalles aux places, aux transitions, et aux arcs. Ce modèle basé sur les inclusions de langages temporels et atemporels, considére des temps discret et dense, tout en associant des sémantiques forte et faible. Dans cela, il a été démontré qu'avec une sémantique faible tous les modèles sont équivalents, et que les modèles les plus expressifs sont ceux utilisant un temps dense associé à une sémantique forte. Cette comparaison fut raffinée dans [39][40], et il a été prouvé qu'en considérant une sémantique forte et un temps dense, le modèle de *Merlin [74]* est moins expressif qu'un $TLPN$ [95].

Par ailleurs, dans le but d'étendre les $RdPT$ aux contraintes de synchronisations, *les réseaux de Petri à flux ou* $TSPN$ *(Time Stream Petri Net)* [53] ont été proposé pour la modélisation des systèmes complexes dits *faiblement synchrones*, comme une extension des $RdPT$ [74] et des $OCPN$ *(Object Composition Petri Net)*[70]. Ces systèmes englobent un large panel allant des systèmes fortement synchrones aux systèmes asynchrones. Au fait, un $TSPN$ est un RdP augmenté d'intervalles sur les arcs, comme dans un $TLPN$ [95], mais avec deux principales différences :

- Un $TSPN$ offre neuf types de synchronisation pouvant être associées à une transition, alors qu'un $TLPN$ en offre seulement une ;

– Plus encore, dans un $TLPN$, la sémantique du temps est faible ; une transition n'est jamais forcée de tirer. Au fait, un $TLPN$ ne définit pas d'état spécial pour la violation temporelle, tandis que dans un $TSPN$ la violation des contraintes de temps peut se produire, si l'écoulement du temps surpasse les contraintes d'une transition ; menant ainsi à des *états inconsistants.*

Dans le but perpétuel d'étendre les *RdPT* à la prise en charge des sémantiques complexes, d'autres extensions de *RdPT* ont été proposées pour modéliser la suspension et la reprise d'actions dans les systèmes temps réel. Parmi ces extensions, les *Scheduling-TPN (Scheduling Time Petri Nets)*, introduits dans [68]. Ils étendent les *RdPT* en incluant dans la sémantique du modèle le comportement des ordonnanceurs temps réel. Cela implique la suspension et la reprise de l'exécution des tâches. Ce modèle s'appuie sur le concept de *chronomètre* (ou *stopwatch*) [46] ; horloge pour laquelle le temps peut être arrêté. La même approche est développée dans [42], avec les $PTPN$ *(Preemptive Time Petri Nets)* qui associent aux transitions des ressources et des priorités d'accès à ces dernières. Les auteurs proposent en outre une méthode intéressante permettant une analyse temporelle quantitative du réseau. De là, ils peuvent en déduire des propriétés temporelles de l'application modélisée. Dans le même contexte, *O.Roux* et *D.Lime* ont proposé dans [83], les $IHTPN$ *(Time Petri Nets with Inhibitor Hyperarcs)*, un modèle plus général de réseaux de Petri fusionnant les deux modèles classiques des *RdPT* et des réseaux de Petri à hyper arcs inhibiteurs [62]. Au fait, les deux premières extensions ajoutent ressources et priorités au modèle *RdPT*, les $IHTPN$ introduisent des arcs inhibiteurs qui contrôlent la progression des transitions.

2.3.2 Réseaux de Petri temporels

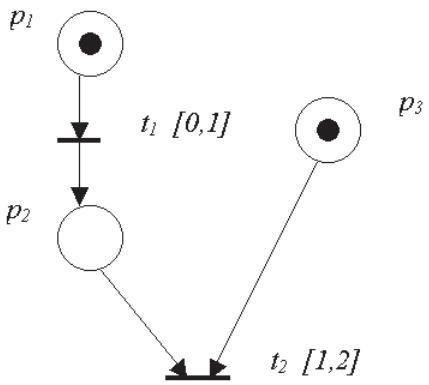

FIGURE 2.6 – Réseau de Petri temporel

Les réseaux de Petri temporels *RdPT* (TPN pour *Time Petri Nets* en anglais) [74] sont une extension des réseaux de Petri introduit par Merlin en 1974. Le modèle de Merlin, généralement appelé réseau de Petri t-temporel, a été conçu pour l'étude des problèmes de recouvrement pour les protocoles de communication. Dans ce modèle, à chaque transition est associée une contrainte temporelle de type intervalle. L'intervalle associé à la transition t est relatif au moment où la transition devient sensibilisée. Soit $[EFT(t), LFT(t)]$ cet intervalle. Supposons que t soit validée a l'instant θ, alors t peut être tirée seulement entre $EFT(t) + \theta$ et $LFT(t) + \theta$, sauf si elle est désensibilisée par le tir d'une autre transition avec laquelle elle était en conflit.

La *Figure* 2.6 représente un exemple de *RdPT* ; dans cela, seule la transition t_1 est sensibilisée car elle possède un jeton dans sa place d'entrée p_1, elle doit être tirée à une date comprise entre 0 et 1. Le tir de t_1 amène à un nouvel état dans lequel p_2 et p_3 (places d'entrées de la transition t_2) sont marquées, alors t_2 devient à son tour sensibilisée.

Nous donnons ci-dessus la définition formelle d'un *RdPT*.

Définition 4 *(Réseau de Petri t-temporel). Un RdPT est un doublet* $RT = (R, FI)$ *où :*

- *R, est un réseau de Petri marqué ;*
- $FI : T \rightarrow Q^+ \times (Q^+ \cup \{\infty\})$*, est la fonction intervalle statique où* Q^+ *représente l'ensemble des nombres rationnels non négatifs. L'application* FI *associe à chaque transition* t *du réseau son intervalle statique de franchissement* $FI(t) = [EFT(t), LFT(t)]$ *tel que :*
  - $EFT(t) \in Q^+$, $LFT(t) \in Q^+ \cup \{\infty\}$ *et* $0 \preceq EFT(t) \preceq LFT(t)$.
  - $EFT(t)$ *(resp.* $LFT(t)$ *) représente l'instant de tir au plus tôt (resp. au plus tard) de la transition* t.
  - *Une transition* t *doit être sensibilisée et atteindre son délai minimum* $EFT(t)$ *avant de pouvoir être tirée(franchie) et ne peut rester validée au-delà du délai maximum* $LFT(t)$ *sans être tirée. Le tir d'une transition est de durée nulle.*

Règle de fonctionnement

A partir de l'instant initial, le modèle séjourne dans son marquage initial jusqu'au franchissement d'une transition, ce franchissement conduit vers un nouveau marquage (i.e., les jetons utilisés disparaissent et ceux produits apparaissent). Le modèle va ensuite séjourner dans le nouveau marquage jusqu'au prochain franchissement et ainsi de suite. Donc le modèle évolue et son évolution est due soit à la progression du temps, soit au franchissement d'une transition.

Remarque : Dans ce qui suit, nous excluons le cas de la multi-sensibilisation des transitions [3] (nous considérons des réseaux *T-sauf* modélisant les systèmes mono-serveur). i.e pour tout marquage accessible M, aucune transition du réseau n'est sensibilisée simultanément plus d'une fois : $(\forall t \in T,\ 2 \times Pre(p,t) \succ M)$. Cette hypothèse peut être levée en considérant les transitions multi-sensibilisées pour un même marquage comme étant des transitions différentes (chaque sensibilisation de la transition a sa propre identification et ses propres paramètres temporels). Ensuite, la multi-sensibilisation peut être traitée selon les stratégies dites « *non déterministe* » ou « *première sensibilisée – première tirée* » développée dans [30]. Dans la première stratégie, les transitions multi-sensibilisées sont considérées comme indépendantes (choix aléatoires de celle à franchir et de celle à désensibiliser en cas de conflit). Par contre, dans la seconde stratégie, les transitions multi-sensibilisées sont ordonnées de la plus récente à la plus vieille. La plus vieille transition est franchie en premier. En cas de conflit, par exemple, la transition la plus récente est désensibilisée.

Le comportement d'un *RdPT* peut être caractérisé par la notion d'état. De manière intuitive un état sera représenté par un couple composé :

– du marquage courant ;

3. Ceci permet de modéliser les systèmes multiserveur.

– de l'intervalle de tir pour chaque transition validée (au sens des *RdP* classiques).

De manière plus formelle, nous avons la définition suivante.

Définition 5 *(Un état). L'état d'un RdPT est défini par une paire* $e = (M, I)$*, telle que :*
- *M est l'application marquage, assignant à chaque place p du réseau un certain nombre de marques ;*
- *I est l'application intervalle dynamique de tir, associant à chaque transition sensibilisée du réseau, l'intervalle de temps dynamique dans lequel elle peut être tirée :* $I : Ts(M) \longrightarrow Q^+ \times (Q^+ \cup \{\infty\}), \quad I(t) = [EFT_e(t), LFT_e(t)]$

Remarque : Les intervalles de tir dans l'état courant e peuvent différer de ceux assignés initialement (les intervalles statiques) aux transitions du réseau. Pour cela ils seront appelés intervalles dynamiques.

Condition de franchissement

La définition de l'état nous donne la possibilité de déterminer les conditions de tir des transitions depuis un état. Sachant que l'origine du temps pour θ est l'instant où l'état e a été atteint, une transition t est tirable à un instant θ depuis un état $e = (M, I)$ si et seulement si les deux conditions suivantes sont satisfaites :
- La transition t est sensibilisée par le marquage M au sens des réseaux de Petri classiques, $\forall p \in P, \quad M(p) \succeq Pre(p, t)$;
- θ est compris entre la date de tir au plus tôt de t et la plus petite des dates de tir au plus tard des autres transitions sensibilisées dans l'état e :

$(EFT_e(t) \preceq \theta \preceq LFT_e(t) \quad \wedge \quad \forall t' \in Ts(M), \quad \theta \preceq LFT_e(t')$.

A partir de la notion d'état et des conditions de tir d'une transition depuis un état, l'état suivant peut être calculé.

Algorithme de calcul de l'état suivant

Le tir d'une transition t franchissable depuis un état $e = (M, I)$ à un instant θ (θ étant relatif à l'instant d'apparition de l'état e), conduit au nouvel état $e' = (M', I')$, déterminé par :
- Le nouveau marquage M' : $\forall p \in P$, *M'(p) = M(p) - Pre(p, t) + Post(p, t) ;*
- Les nouveaux intervalles de tir I' :
  - pour toutes les transitions t' sensibilisées par le marquage M', on distingue :
    - pour toutes les transitions persistantes ; t' est sensibilisée par les marquages M et M' et non en conflit avec la transition t :
      $I'(t') = [MAX(0, EFT_e(t') - \theta), \quad LFT_e(t') - \theta]$, avec
      $I(t') = [EFT_e(t'), LFT_e(t')]$
    - les intervalles associés aux autres transitions (transitions nouvellement sensibilisées) sont les intervalles statiques, $I'(t') = [EFT(t'), LFT(t')]$.

Remarque : Les ensembles d'états peuvent être représentés par des paires (M, D), dans lesquelles M est un marquage et D un ensemble de vecteurs de dates appelé domaine de tir. La i-ème projection de l'espace D est l'intervalle de tir $I(t_i)$ associé à la i^{ieme} transition sensibilisée.

Les domaines de tir seront décrits par des systèmes d'inéquations linéaires avec une variable par transition sensibilisée (notée comme les transitions). Cette approche est décrite ci-après.

La méthode du graphe des classes d'états

La méthode la plus courante de calcul de l'espace d'état d'un *RdPT* est le *graphe des classes d'états [32]*. Puisque l'espace d'états est infini, il faut rassembler les états en un nombre fini de groupes. Dans cette méthode, les groupes sont appelés classes d'états. Formellement, les états d'une même classe sont tous les états accessibles après franchissement de la même séquence de tir depuis l'état initial, mais à des dates différentes.

Définition 6 ***(Classe d'états).*** *Une classe d'états est un couple $E = (M, D)$ dans lequel :*

- *M, est le marquage de la classe (marquage commun à tous les états de la classe) ;*
- *D, est le domaine de tir de la classe, défini par l'union des intervalles de tir associés à chaque état de la classe. La classe initiale E_0 contient seulement l'état initial e_0. D peut être représenté par l'ensemble des solutions d'un système d'inéquations linéaires, comportant une variable pour chaque transition sensibilisée par le marquage M de la classe. Les inéquations de D sont de deux types ; on parle alors d'un système mis sous forme DBM (Difference Bound Matrix) :*
  - $\forall t_i \in Ts(M), \quad 0 \preceq MIN_{e\in E}\{EFT_e(t_i)\} \preceq \quad \underline{t_i} \quad \preceq MAX_{e\in E}\{LFT_e(t_i)\}$;
  - $\forall t_j, t_i \in Ts(M) \wedge (t_i \neq t_j), \quad \underline{t_j} - \underline{t_i} \preceq MAX_{e\in E}\{LFT_e(t_j) - EFT_e(t_i)\}$.

    Où $\underline{t_j}$ et $\underline{t_i}$ sont des variables associées respectivement aux transitions t_j et t_i.

Transition entre classes d'états (Condition de tir d'une transition) : Une transition t_i est tirable depuis la classe $E = (M, D)$, si et seulement si :

(i) t_i est sensibilisée par M au sens des *RdP* ;

(ii) $D \wedge (\forall t \in Ts(M), \underline{t_i} \preceq \underline{t})$ est consistant.

La condition (ii) interdit le tir de toute transition validée (au sens des *RdP*) dont l'intervalle de tir (courant) est strictement précédé par celui d'une autre transition validée. Le tir de t_i conduit à la classe $E' = (M', D')$ suivante, caractérisée par :

- Le nouveau marquage $M' : \forall p \in P, M'(p) := M(p) - Pre(p, t_i) + Post(p, t_i)$;
- Le nouveau domaine D' est obtenu selon la procédure suivante (4 étapes) :

  a) Ajouter au système D, les conditions de tir de t_i exprimant qu'elle soit la première tirée parmi l'ensemble des transitions validées, i.e.
  $D \wedge (\forall t_j \in Ts(M), \underline{t_i} \preceq \underline{t_j})$;

  b) Supprimer dans le système obtenu les inéquations incluant les variables $\underline{t_j}$ associées aux transitions en conflit avec t_i, ces transitions sont désensibilisées par le tir de t_i ;

  c) Dans le système ainsi réduit, effectuer le changement de variable suivant :
  $\forall i \neq j, \quad \underline{t_j} = \underline{t'_j} - \underline{t_i}$, et éliminer par substitution toutes les occurrences de la variable $\underline{t_i}$, pour ne garder que les nouvelles t'_j ;

  d) Compléter ce dernier système par une variable supplémentaire pour chaque transition nouvellement sensibilisée. Associer à ces transitions leurs intervalles de tir statiques.

L'ensemble de solutions du système déterminé à l'étape (c) peut être vu comme le domaine de tir des transitions distinctes de t_i qui sont restées sensibilisées pendant le tir de t_i, exprimé avec pour nouvelle origine du temps la date à laquelle la transition t_i a été tirée. Les éliminations effectuées aux étapes (b) et (c) préservent les contraintes temporelles induites sur les variables restantes.

A partir de cette représentation finie des états accessibles et de la relation de transition entre classes, on peut aborder la notion de graphe des classes d'états.

Construction du graphe des classes d'états

La racine de ce graphe est la classe initiale E_0 ; pour chaque transition tirable dans une classe E, une nouvelle classe E' est créée. Afin de diminuer le risque d'explosion combinatoire, chaque nouvelle classe est comparée avec celles déjà produites, pour tester une possible égalité. Si c'est le cas, l'exploration de la branche concernée est abandonnée (existence de séquences répétitives). L'existence d'un chemin dans cet arbre, reliant la classe initiale à la classe E, prouve la possibilité d'existence d'un échéancier de tir réalisable, supporté par ce chemin. Deux points restent à préciser pour pouvoir construire efficacement cet arbre :

- Comment tester l'égalité de deux classes ?
- Comment savoir si l'énumération se termine ?

Condition d'égalité entre classes d'états : Deux classes d'états $E = (M, D)$ et $E' = (M', D')$ sont égales par définition si et seulement si : $M = M'$ et $D = D'$.

Il a été démontré dans [73] que le système d'inégalités particulières qui définit les domaines de tir, admet une représentation sous forme *DBM* unique. La complexité de calcul de la forme est polynômiale [32], précisément, égale à $o(l^3)$ où l étant le nombre de transitions sensibilisées pour M. Cependant, la complexité de calcul d'une classe peut être réduite à $o(l^2)$ en éliminant certains calculs redondants par la proposition d'un algorithme récursif de calcul des matrices des bornes préservant cette forme [94] [37]. Ces algorithmes permettent aussi de calculer les temps maximaux et minimaux d'un chemin du graphe en un temps polynômial estimé à $o(k \times l)$ où k est la longueur du chemin.

Lorsque la forme *DBM* des domaines de tir est établie, vérifier l'identité de leurs formes suffit pour conclure à l'égalité de deux domaines. Au vu des résultats précédents, l'égalité de deux classes d'états pourra donc être construite de manière incrémentale, en même temps que sont énumérées les classes. L'énumération est achevée lorsque toutes les branches de l'arbre ont été explorées ; cela suppose que le nombre de classes soit fini. Mais nous avons vu que la finitude de l'ensemble des marquages accessibles est indécidable. Comme la définition des classes d'états inclut le marquage, la finitude de l'ensemble des classes est aussi indécidable d'où l'objet du paragraphe suivant.

Finitude du graphe des classes : En assumant un *RdPT* T-Sauf, toute classe a un nombre fini de successeurs (au plus un par transition sensibilisée). Il reste à examiner les conditions sous lesquelles l'ensemble des classes est fini.

Théorème 1 *[32] Le nombre de classes d'un réseau de Petri temporel T-Sauf est fini si et seulement si le réseau est borné.*

Par conséquent, ce problème est indécidable car le problème de bornitude d'un *RdPT* est indécidable. *Berthomieu et autres* [32] ont démontré que si les bornes des intervalles initiaux sont des rationnels, le nombre des domaines de tirs possibles est fini. Puisqu'un état est déterminé par le marquage et le domaine de tir, à partir de ce résultat, il est très facile de montrer que si le réseau est borné alors le graphe est borné et vice-versa.

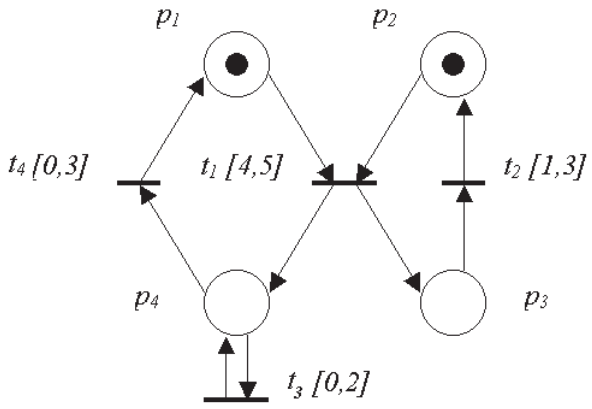

FIGURE 2.7 – Exemple d'application

Exemple A titre d'illustration, construisons les classes du réseau temporel représenté en *Figure* 2.7. Initialement seule la transition t_1 est sensibilisée.

La classe initiale $E_0 = (M_0, D_0)$ avec : $M_0 : \{p_1, p_2\} \to 1$. $\quad D_0 \left\{ 4 \preceq \underline{t_1} \preceq 5.$

– Le franchissement de t_1 depuis E_0 conduit à la classe $E_1 = (M_1, D_1)$, avec :

$$M_1 : \{p_3, p_4\} \to 1 \quad D_1 : \begin{cases} 1 \preceq \underline{t_2} \preceq 3 \\ 0 \preceq \underline{t_3} \preceq 2 \\ 0 \preceq \underline{t_4} \preceq 3 \end{cases}$$

– Les transitions sensibilisées par le marquage M_1 sont : t_2, t_3, t_4. Franchir t_2 depuis E_1 dans [1, 2] conduit à la classe $E_2 = (M_2, D_2)$, avec $M_2 : \{p_2, p_4\} \to 1$; et D_2 est calculé en quatre étapes, selon la règle définie dans le paragraphe précédent :

Étape (a) : D_2(a) est obtenu en ajoutant à D_1 les conditions de tirabilité de t_1 :

$$\begin{cases} 1 \preceq \underline{t_2} \preceq 3 & \underline{t_2} \preceq \underline{t_3} \\ 0 \preceq \underline{t_3} \preceq 2 & \underline{t_2} \preceq \underline{t_4} \\ 0 \preceq \underline{t_4} \preceq 3 \end{cases}$$

Étape (b) : Aucune transition n'étant en conflit avec t_2, on a D_2(b) = D_2(a).

Étape (c) : Le changement d'origine produit le système suivant :

$$\begin{cases} 1 \preceq \underline{t_2} \preceq 3 & 0 \preceq \underline{t_2} + \underline{t_4} \preceq 3 \\ 0 \preceq \underline{t_2} + \underline{t_3} \preceq 2 & \underline{t_2} \preceq \underline{t_2} + \underline{t_4} \\ \underline{t_2} \preceq \underline{t_2} + \underline{t_3} \end{cases}$$

Depuis lequel D_2(c) est obtenu par élimination de t_2 : $\begin{cases} 0 \preceq \underline{t_3} \preceq 2 \\ 0 \preceq \underline{t_4} \preceq 3 \\ \underline{t_4} - \underline{t_3} \preceq 1 \end{cases}$

Étape (d) : Aucune transition n'étant nouvellement sensibilisée, on a alors $D_2 = D_2(c)$.

- Franchir t_3 depuis E_1 dans [0, 2] conduit à une nouvelle classe $E_4 = (M_4, D_4)$ avec : $M_4 : \{p_3, p_4\} \rightarrow 1$

  $D_4 : \begin{cases} 0 \preceq \underline{t_2} \preceq 3 \\ 0 \preceq \underline{t_3} \preceq 2 \\ 0 \preceq \underline{t_4} \preceq 3 \end{cases}$
- Franchir t_4 depuis E_1 dans [0, 2] conduit à la classe E_5 avec : $M_5 : \{p_1, p_3\} \rightarrow 1$; $D_5 : \{0 \preceq \underline{t_2} \preceq 3$

Les transitions sensibilisées par le marquage M_2 sont : t_3, t_4.

- Franchir t_3 depuis E_2 dans [0, 1] conduit à la classe $E_3 = (M_3, D_3)$ avec : $M_3 : \{p_2, p_4\} \rightarrow 1$. $D_3 : \begin{cases} 0 \preceq t_3 \preceq 2 \\ 0 \preceq t_4 \preceq 1 \end{cases}$
- Franchir t_4 depuis E_2 dans [0, 1] conduit à la classe initiale E_0.

Les transitions sensibilisées par M_3 sont : t_3, t_4, telles que :

- Franchir t_3 depuis E_3 dans [0, 2] conduit à la même classe E_3.
- Franchir t_4 depuis E_3 dans [0, 2] conduit à la classe initiale E_0.

Depuis E_4 :

- Franchir t_2 depuis E_4 dans [0, 2] conduit à E_3.
- Franchir t_3 depuis E_4 dans [0, 2] conduit à la même classe E_4.
- Franchir t_4 depuis E_4 dans [0, 2] conduit à une nouvelle classe $E_5 = (M_5, D_5)$

Seule la transition t_2 est sensibilisée par M_5 :

- Franchir t_2 depuis E_5 dans [0, 3] conduit à la classe initiale E_0.

L'énumération précédente des classes accessibles produit le graphe des classes de la *Figure* 2.8.

2.4 Autres extensions

2.4.1 Réseau de Petri temporel à priorité temps réel [10]

Lorsque plusieurs transitions sont franchissables à partir d'un même marquage, le choix de la transition à franchir se fait de manière non déterministe. Une solution standard pour résoudre le non déterminisme est d'affecter des priorités statiques aux transitions. On parle alors de réseaux de Petri à priorité [60]. Dans ce type de réseau, une relation d'ordre partiel est définie sur l'ensemble des transitions, et une transition est franchissable si aucune autre transition plus prioritaire n'est

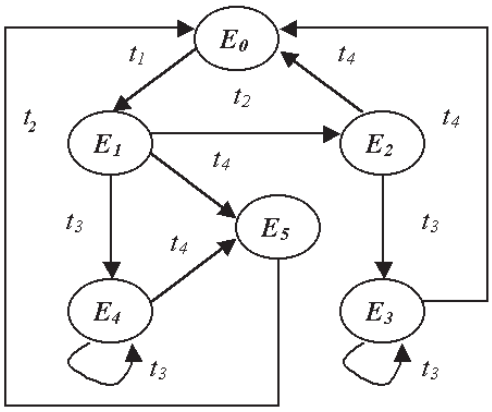

FIGURE 2.8 – Graphe des classes

sensibilisée pour le même marquage. Nous avons proposé dans [10] l'extension de cette sémantique pour les *RdPT* permettant de résoudre le non déterminisme en temps réel. Ce nouveau modèle permet d'éviter la génération des situations de blocage et en réduisant l'apparition des situations de famine. Ce mécanisme appelé « *mécanisme de priorité en temps réel* » énonce qu'une transition sensibilisée, est tirable à un instant donné si aucune autre transition (plus prioritaire et sensibilisée pour le même marquage) n'est franchissable au même instant.

Remarque : Cette sémantique permet de sélectionner la transition à franchir en temps réel i.e. Pour chaque marquage accessible à un instant donné, une seule transition parmi celles en conflit est au plus franchissable, alors que la sémantique traditionnelle restreint à une seule transition tirable au plus par marquage accessible.

Exemple :

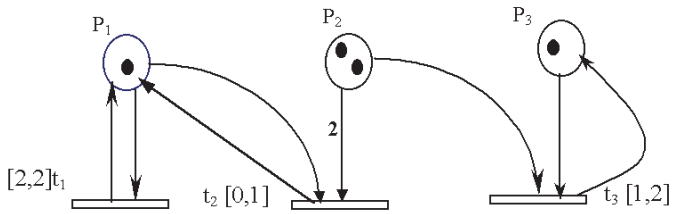

FIGURE 2.9 – Réseau de Petri temporel à priorité temps réel

Considérons l'exemple de la *Figure* 2.9 en assumant l'ordre de priorité suivant : $t_3 \succ t_2 \succ t_1$. Le graphe illustré dans la *Figure* 2.10.*a* correspond au graphe des classes obtenu du *RdPT* sous jacent. Nous pouvons remarquer que la transition t_1 est vivante et qu'il y a absence de famine à partir du marquage initial. L'application des priorités statiques sur le réseau produit le graphe de classes de la *Figure* 2.10.*b*. Dans cela, la transition la plus prioritaire est autorisée à franchir à partir de chaque classe, provoquant une situation de famine par rapport à la transition t_2. Enfin, le graphe décrit dans la *Figure* 2.10.*c* correspond au graphe calculé en appliquant la "*priorité en temps réel*". Initialement, la transition t_2 ne peut être tirée que durant $[0, 1[$ excluant les instants pour lesquels t_3 est franchissable (i.e,$[1, 1]$). Par contre, à partir de la classe E_{11} le tir de t_2 est impossible car ne pouvant se produire en dehors des instants de franchissement de la transition

t_3. Toutefois, la situation de famine constatée en appliquant les priorités statiques restreintes au marquage, est absente dans le cas de l'application de la priorité en temps réel.

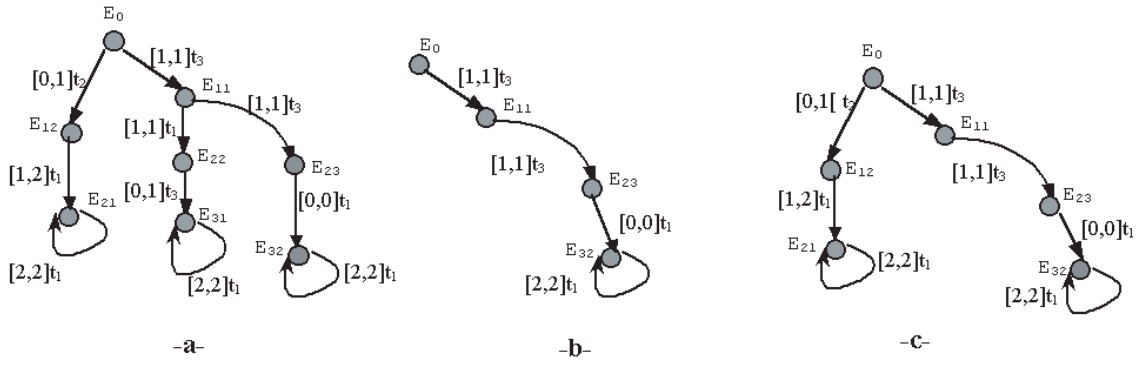

FIGURE 2.10 – Comparaison des graphes obtenus

2.4.2 Réseau de Petri temporel étendu aux chronomètres.

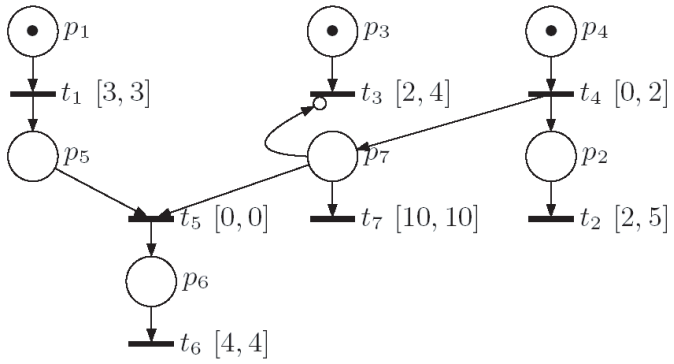

FIGURE 2.11 – Un réseau de Petri temporel à chronomètre étendu aux arcs inhibiteurs

Modéliser certains systèmes temps réel nécessite d'exprimer la suspension et la reprise d'actions. Pour répondre à ces besoins, plusieurs extensions des *RdPT* intégrant la notion de chronomètre ont été proposées pour permettre d'exprimer la suspension et de la reprise d'actions : les *Scheduling-TPN* [68], les *Preemptive-TPN* [42] et les $IHTPN$ [83]. Les deux premiers ajoutent ressources et priorités aux *RdPT* ; les $IHTPN$ introduisent des arcs inhibiteurs qui contrôlent la progression des transitions. Toutefois, les $IHTPN$ ont été prouvés plus expressifs que les autres modèles [83]. Notons que l'accessibilité d'états dans tous ces modèles est indécidable mais le problème reste ouvert pour des réseaux bornés (d'un grand intérêt pratique).

Pour toutes ces extensions, des semi-algorithmes calculant une abstraction de l'espace d'états en termes de classes d'états sont disponibles se faisant par l'utilisation de surapproximations qui caractérisent un ensemble d'états incluant l'espace exact mais pouvant être plus important. Ces méthodes consistent à approximer les polyèdres des classes par le plus petit représentable par une matrice de différences des bornes (DBM) le contenant [42]. La méthode est efficace, mais les surapproximations obtenues sont souvent grossières. Néanmoins, elles permettent de fournir des conditions suffisantes pour les propriétés de sûreté.

Considérons l'exemple de la *Figure*.2.11, présenté dans [83], décrivant un $IHTPN$. La sémantique de l'arc inhibiteur agit comme suit, lorsque la transition t_3 est inhibée (après le tir de t_4) son horloge est stopée et sera réactivée juste après l'arrêt de son inhibition (après le tir de t_5 ou t_7). Considérons par exemple le tir de la séquence t_4, t_1, t_5 on obtient la classe suivante : $E = (M, D)$

$M : \{p_2, p_3, p_6\} \rightarrow 1$

$$D : \begin{cases} 0 \preceq \underline{t_2} \preceq 4, & -4 \preceq \underline{t_2} - \underline{t_3} \preceq 4, \\ 0 \preceq \underline{t_3} \preceq 4, & -4 \preceq \underline{t_2} - \underline{t_6} \preceq 0, \\ 4 \preceq \underline{t_6} \preceq 4, & -4 \preceq \underline{t_3} - \underline{t_6} \preceq 0, \\ & 1 \preceq \underline{t_2} + \underline{t_3} \preceq 6 \end{cases}$$

On peut facilement remarquer que la transition t_6 est non tirable car t_2 ou t_3 doivent être tirées auparavant. Plus clairement, pour que t_6 soit tirable il faudrait que le système $D \wedge (\underline{t_6} \preceq \underline{t})$ soit consistant, c.-à-d nous devons vérifier que $\underline{t_2} = \underline{t_3} = \underline{t_6} = 4$ et que $\underline{t_2} + \underline{t_3} \preceq 6$. Cette dernière inéquation n'étant pas satisfaite, donc t_6 est non tirable. Le système D obtenu, est un polyèdre de forme quelconque qui ne peut être mis sous forme DBM en utilisant la technique des graphe des classes. La résolution de ce polyèdre implique un coût important d'ordre exponentiel. Par conséquent, pour maintenir la complexité au même ordre que pour les $RdPTs$, les algorithmes existants procèdent à l'approximation du polyhèdre en supprimant toutes les inéquations ne respectant pas la forme DBM. Dans notre cas, cela reviendrait à éliminer l'inégalité, $1 \preceq \underline{t_2} + \underline{t_3} \preceq 6$. Cependant, en supprimant cette dernière, t_6 devient tirable. Donc, nous avons ici une surapproximation du domaine D. En procédant de la sorte, nous rajoutons des nouvaux états dans la classe qui ne devraient pas être accessibles, c'est pourquoi qu'on parle de surapproximation. Néanmoins pour la vérification des propriétés de **sûreté**, la surapproximation n'est pas un handicap, car cette propriété exige que "*quelque chose de mauvais n'arrive jamais*". Pour ce faire, nous avons besoin de la vérifier sur l'espace calculé, avec le risque d'être pessimiste dans le cas où la propriété n'est pas vérifiée sur l'espace surapproximé.

2.4.3 Réseau de Petri à Flux (TSPN)

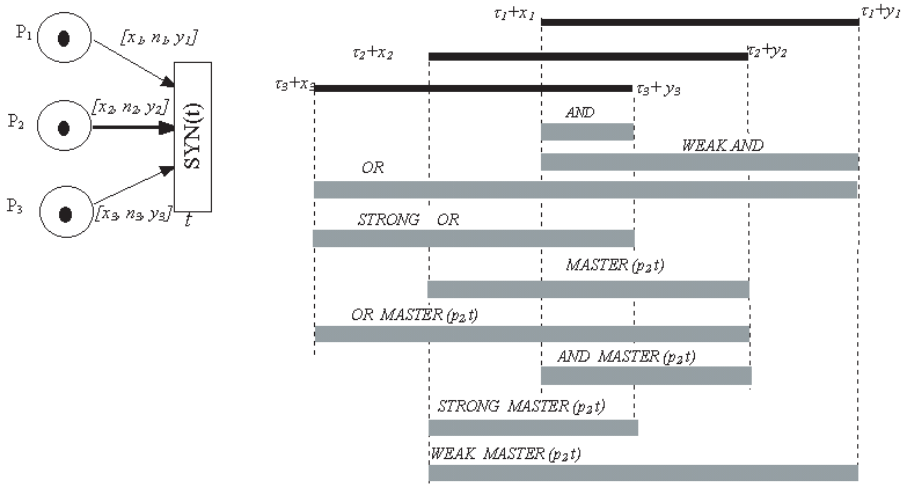

FIGURE 2.12 – Synchronisations dans un Réseau de Petri à Flux

Les réseaux de Petri à Flux ou *(Time Stream Petri Net)* [53] ont été introduit pour la

modélisation des contraintes de synchronisation dans les systèmes faiblement synchrones. Cette notion de faiblement synchrone offre un cadre générique qui couvre une grande palette de systèmes allant des systèmes synchrones aux systèmes purement asynchrones. Ce pouvoir d'expression des réseaux à flux a été exploité notamment dans la spécification des systèmes multimédia. Ce modèle étend les *Timed Link Petri Nets* [95] au sept schémas de synchronisations définis dans *OCPN (Object Composition Petri Net)* [70] tels que, les processus sont représentés comme des places, et leurs caractéristiques temporelles sont données par des d'intervalles associées aux arcs. Ces intervalles appelés *intervalles de validité temporelle (TVI),* sont définis comme un triplet $[x, n, y]$, où x, n et y sont respectivement, la *durée minimale*, *nominale* et *maximale* admissibles relatives au processus. Il y a trois stratégies fondamentales de synchronisation dans un $TSPN$ impliquant neuf règles obtenues à partir d'une combinaison consistante et complète des intervalles temporels de validité absolus des arcs associés à une transition marquée :

- Stratégies de synchronisation dynamiques {*'And', 'Weak And', 'And-Master'*} régies par le dernier processus, (i.e., Le dernier arc qui atteint la borne minimum de son TVI autorise le franchissement de la transition).
- Stratégies de synchronisation dynamiques, {*'Or', 'Strong Or', 'Or Master'* } régies par le premier processus, (i.e., le premier arc qui atteint la borne minimum de son TVI autorise le franchissement de la transition).
- Stratégies de synchronisation statiques {*'Master', 'Strong Master', 'Weak-Master'* } régies par un processus sélectif, (i.e., la transition peut être franchie seulement si son arc maître ait atteint la borne minimale de son TVI).

En conclusion, les $TSPN$ offrent neuf stratégies de synchronisation (voir *Figure*.2.12) qui peuvent être associées à une transition, introduites grâce à la fonction *Syn(t).* En particulier, il existe deux stratégies pour régir respectivement une synchronisation par le flux au plus tard et au plus tôt. Enfin, la synchronisation peut être régie par le flux maître dont les caractéristiques sont prédéfinies statiquement dans le modèle.

Fonctionnement

Soit $A_{in}(t_i)$, l'ensemble des arcs entrants de la transition t_i. On désigne par la notation a_{ij} l'arc (p_i, t_j). Supposons $\tau(a_{ij})$ la date absolue de la sensibilisation de l'arc a_{ij}, i.e, l'instant où le nombre de jetons dans la place p_i satisfait la précondition de l'arc a_{ij}. Par conséquent, l'intervalle $[x(a_{ij}), y(a_{ij})]$ associé à l'arc a_{ij} dénote l'intervalle des dates relatives (à l'instant $\tau(a_{ij})$), durant lesquelles les jetons dans la place p_i pourront être consommés lors du tir de la transition t_j. L'arc a_{ij} modélise ici un processus autonome dont les contraintes temporelles sont spécifiées par son intervalle de validité. La modélisation d'une synchronisation entre n processus (arcs) (i.e, a_{ij} $i = 1..n$), est réalisée par la transition t_j. La synchronisation est possible lorsque la transition t_j est sensibilisée (i.e, lorsque tous ses arcs le sont) et sa date de sensibilisation notée $\tau(t_j)$ est donnée par $\underset{\forall a_{ij} \in A_{in}(t_j)}{MAX} \{\tau(a_{ij})\}$. Par exemple, une évolution synchrone de tous les processus (Stratégie *And*) exigerait de consommer tous les jetons en respectant les contraintes des n processus. Plus formellement, t_j peut être tirée à l'instant relatif δ, si :

$$\tau(t_j) + \underset{\forall a_{ij} \in A_{in}(t_j)}{MAX} \{x(a_{ij})\} \preceq \tau(t_j) + \delta \preceq \tau(t_j) + \underset{\forall a_{ij} \in A_{in}(t_j)}{MIN} \{y(a_{ij})\}.$$

Une évolution asynchrone (Stratégie Or) exigerait qu'au moins l'un des procesus ait ses contraintes satisfaites. Donc, la transition t_j peut être tirée à l'instant δ, si : $\tau(t_j)+\underset{\forall a_{ij}\in A_{in}(t_j)}{MIN}\{x(a_{ij})\} \preceq \tau(t_j)+\delta \preceq \tau(t_j)+\underset{\forall a_{ij}\in A_{in}(t_j)}{MAX}\{y(a_{ij})\}$.

Par ailleurs, une évolution contrôlée (Stratégie $Master$) exigerait que le procesus sélectionné (appelé *Maître* ou *Master*) ait ses contraintes satisfaites. Donc si a_{mj} est l'arc représentant le processus maître, la transition t_j peut être tirée à l'instant δ, si : $\tau(t_j)+x(a_{mj}) \preceq \tau(t_j)+\delta \preceq \tau(t_j)+y(a_{mj})$.

De même, les autres schémas de synchronisation permettent de définir des variantes des trois stratégies décrites précédemment, permettant ainsi de modéliser une grande partie des relations de synchronisations introduites par l'algèbre d'*Allen* [23]

2.5 Conclusion

Depuis leur introduction en 1962 par C. A. Petri, les *Réseaux de Petri* ont largement démontré leur efficacité et leur facilité d'usage pour la modélisation et l'analyse des systèmes à événements discrets, offrant ainsi un modèle puissant pour la modélisation des systèmes parallèles et concurrents. Rapidement, la nécessite de la prise en compte du facteur temps comme une composante explicite de l'application modélisée, a conduit à l'émergence des différents modèles temporisés. Ces derniers ont été intensivement utilisés dans une optique d'évaluation des performances. Néanmoins, la modélisation de certains systèmes temps-réel complexes nécessitent d'exprimer plusieurs de ces sémantiques dans un même modèle. Partant de là, la définition d'un tel cadre de spécification permettrait d'envisager une caractérisation exhaustive des contraintes imposées et une analyse plus correcte des propriétés. Pour répondre à ces besoins, nous explorons dans le prochain chapitre la possibilité d'étendre les *RdPT* à différents mécanismes dans le but de capturer la majorité des comportements observés dans les systèmes temps réel complexes, plus particulièrement dans les applications multimédia.

Chapitre 3

Modélisation des contraintes multimédia par des RdPT

3.1 Introduction

Une présentation multimédia peut être vue comme un ensemble de présentations de médias de base (e.g. vidéo, image, texte, audio ou animation), organisées dans l'espace et dans le temps selon différents schémas (en parallèle ou en séquence), pouvant être contraintes temporellement, et fortement synchronisées. De plus, elles peuvent partager des ressources communes et autoriser des interactions externes des utilisateurs.

Le pouvoir d'expression de ces documents infère des sémantiques complexes menant le plus souvent à des disfonctionnements lors de leurs exécutions. Ces erreurs sont dues principalement à une mauvaise caractérisation des différentes contraintes lors de la phase d'édition. Une méthodologie formelle pourrait être utilisée pour prévenir l'apparition de ces problèmes et de les corriger en amont, à condition d'être capable de capturer le plus fidèlement possible les comportements de ces applications, et de les détecter lors de la phase d'analyse. La première phase de cette méthodologie consiste donc à retranscrire les contraintes de ces applications en une spécification formelle cohérente.

Partant de ce constat, nous cherchons dans ce chapitre à exploiter les *RdPT* pour la spécification des contraintes observées dans les applications multimédia complexes. Plus particulièrement nous focalisons sur la spécification des contraintes des documents hypermédia\multimédia. Nous montrons d'abord comment modéliser un objet de base et puis progressivement, nous considérons des présentations plus complexes où la sémantique des *RdPT* est étendue aux *hyperarcs préempteurs*[14], aux *arcs de privation* [20], aux *hyperarcs de franchissement*, aux *arcs inhibiteurs*, aux *chronomètres*, et aux *mécanismes de synchronisation.* Une attention particulière est dédiée à la modélisation des interactions des utilisateurs (confrontées à la disponibilté des ressources d'affichage), dont les occurrences sont à l'origine de la plupart des inconsistances observées dans les présentations multimédia.

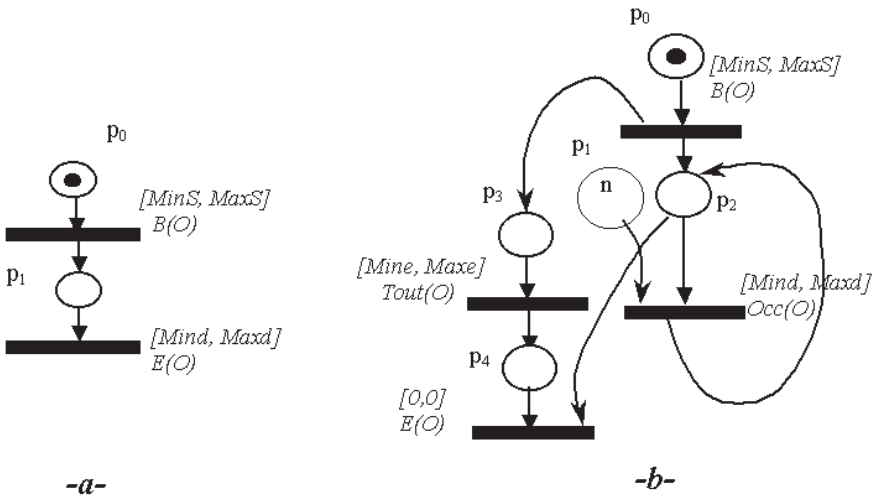

FIGURE 3.1 – Modélisation des contraintes temporelles d'une présentation de base

3.2 Modélisation des contraintes d'un média de base

3.2.1 Modélisation des contraintes de temps

Nous modélisons les contraintes de chaque média de base cité dans le document multimédia par une **unité** décrivant un *RdPT*. Ensuite, les différentes unités sont composées selon la structuration du document multimédia. Premièrement, la spécification des contraintes d'un média noté O, est caractérisée par deux événements clé ; l'événement de début $B(O)$ et l'événement de fin $E(O)$. De plus, les occurrences de ces événements peuvent être contraintes temporellement selon les exigences de la présentation. Nous modélisons les contraintes temporelles du média O par la spécification donnée en *Figure* 3.1.*a*, où $[MinS,\ MaxS]$ représente le délai à observer avant le début du traitement de O, et $[Mind,\ Maxd]$ délimite la durée totale d'une occurrence de la présentation du média O. Notons que le temps utilisé ici est relatif. De plus, la place p_1 reliant les transitions $B(O)$ et $E(O)$ dénote une fois marquée que le media est en cours de présentation. Nous appellons cette place, "*place centrale*" de l'unité O.

Par ailleurs, une caractérisation exhaustive des exigences d'un traitement plus complexe permet de considérer le cas où plusieurs occurrences d'une même présentation sont à réaliser. Pour ce faire, la *Figure* 3.1.*b* montre comment cette spécification peut être modélisée ; La place p_1 est utilisée pour représenter le nombre de répétitions à parfaire indiqué par le nombre de jetons (n) contenus dans la place p_1. La transition estampillée "$occ(O)$" indique, une fois tirée, la réalisation d'une occurrence parmi celles requises pour la présentation de O. L'intervalle $[Mine, Maxe]$ modélise l'intervalle de temps durant lequel l'ensemble des répétitions doivent se réaliser. Par conséquent, lorsque ce délai est atteint (i.e., tir de $tout(O)$), la présentation de O devra impérativement cesser (i.e., le jeton dans la place p_2 est consommé définitivement en tirant $E(O)$).

3.2.2 Modélisation des contraintes sur les ressources

Par ailleurs, la modélisation des exigences des applications multimédia nous amène à considérer (en plus des exigences temporelles et de synchronisation), les besoins en termes de ressources. Nous cherchons par la suite à étudier le comportement de ces applications relativement à la disponibilité

ou non de ces besoins et d'apprécier l'impact de ces derniers sur la qualité de la présentation. Cette étude nous permet d'envisager les voies et moyens idoines pour la modélisation de ce type de comportements.

Un premier examen des besoins des présentations multimédia en termes de ressources permet de démarquer deux grandes catégories, que sont :

1. Les ressources dites *critiques* qui sont obligatoires pour le déroulement de la présentation ; Celles-ci incluent entre autres les ressources système (e.g. Mémoire, CPU) ;
2. Les ressources dites *non critiques* qui peuvent être vues comme importantes mais non obligatoires [1] pour la réalisation d'une présentation multimedia (e.g. zone d'affichage, canal Audio).

Ainsi, les conséquences de la non disponibilité d'une ressource critique pour une présentation multimédia semblent être irrémédiables, dans le sens où il en résulterait un plantage général du lecteur, il demeure que ce type d'incidents (dépendant uniquement des capacités système de la machine), est assez rare et de surcroît non maîtrisable.

Par contre, la non acquisition d'une ressource non critique par une présentation est un phénomène très fréquent, impliquant la non perception d'un certain nombre d'informations véhiculées par le document, et donc influençant négativement la qualité de sa présentation. Ces disfonctionnements sont dus essentiellement aux situations de conflit qui peuvent être corrigés lors de la phase d'édition et mieux gérés lors de la phase de lecture, à condition de pouvoir les détecter préalablement.

Partant de ce premier constat, nous nous intéressons dans ce qui suit aux problèmes exclusivement engendrés par cette seconde catégorie de ressource. Cela passe par la prise en charge de manière précise de cette problématique lors de la phase de modélisation afin de permettre après analyse exhaustive des scénarios induits, la détection des erreurs potentielles.

Nous proposons de discuter ci-après tous les aspects relatifs à cette problématique, en considérant des présentations décrites au moyen du langage $SMIL$[90]. Ce dernier est devenu ces dernières années sans conteste la référence dominante sur Internet pour la caractérisation des présentations multimédia distribuées complexes. Ces documents offrent des mécanismes évolués permettant d'organiser spacialement et temporellement plusieurs présentations de média selon différents arrangements. De plus, ils peuvent nécessiter des ressources spécifiques pour être présentés.

Pour capturer le plus fidèlement possible ces comportements complexes, nous focalisons notre étude sur la gestion des conflits telle qu'elle est procédée par les principaux lecteurs compatibles $SMIL$ [27][81], et de là nous proposons des modélisations exhaustives adaptées aux fonctionnements des différents lecteurs.

Dans le contexte des présentations $SMIL$, on note principalement deux types de ressources non critiques :

- Le *canal audio*, permettant de délivrer le son associé à un média sonore.
- Une *région* déterminant une fenêtre de l'écran définie par des coordonnées spatiales, dans laquelle le média doit être visualisé. Ainsi, chaque média de type *texte*, *image* ou *vidéo* est associé à une des régions définies [2].

1. Un média peut être joué même si l'une des ressources requises est indisponible, mais en étant dénué d'un certain nombre de vecteurs d'information véhiculés par sa présentation.

2. Deux médias peuvent être associés à une même fenêtre.

Cependant, la disponibilité de ce type de ressources est non obligatoire pour l'exécution d'une présentation multimédia. Cette dernière peut se réaliser même si une de ces ressources est non disponible, mais en ne délivrant pas toutes les informations véhiculées. Ainsi, par exemple une vidéo pourrait être visualisée dans sa fenêtre d'affichage mais le son l'accompagnant peut ne pas être perçu si le canal audio est indisponible ; de même, le scénario inverse peut se produire si la zone d'affichage est utilisée par un autre média.

Par ailleurs, plusieurs médias pourraient être en conflit pour la même ressource. Dans ces cas, une politique de priorité opérant en temps réel est établie par le lecteur pour déterminer quel élément va obtenir cette ressource. Par exemple, le lecteur *Real one [81]* établit une priorité ascendante selon l'ordre d'insertion des déclaractions des médias dans le document $SMIL$. D'autres lecteurs comme *GRiNS* [56] ou *Ambulant* [27] adoptent l'ordre inverse. Par ailleurs, ces lecteurs [27][56][81] à travers leurs fonctionnements plus ou moins différents, autorisent tous la préemption des ressources. Ainsi, un média en possession d'une ressource non critique peut en être dépossédé si un autre média de priorité supérieure est joué entre temps. De plus, le lecteur *Real one* est assez restrictif dans son fonctionnement ; il ne permet pas à un média de récupérer la ressource manquante même si cette dernière est libérée avant la fin de son exécution. D'autres lecteurs comme *Ambulant* [27] sont plus flexibles puisqu'ils permettent à un média de recouvrer la ressource requise durant son traitement, si restituée dans les temps.

Remarque : Dans un souci de simplifier la sémantique du modèle, nous assumons qu'un média ne peur acquérir qu'une seule ressource non critique. Cette limitation est justifiée par un constat pratique en premier lieu basé sur l'observation des besoins de telles applications. Nous avons constaté durant nos investigations que dans la majorité des cas, les médias de base requièrent au plus une seule ressource non critique. Dans le cas d'un média nécessitant deux ressources non critiques, telle une vidéo sonore ; ses contraintes pourraient être modélisées en dissociant l'entité audio de l'entité vidéo, en les considérant comme deux médias de base disctincts évoluant en parallèle synchrone et ayant les mêmes caractéristiques temporelles

Représentation d'un schéma d'allocation d'une ressource non critique

Considérons O_i $(i = 1..p)$ p unités, telles que chacune décrit les exigences temporelles d'un média de base, et soit rs une ressource requise par O_i $(i = 1..p)$. Etant donné que la ressource rs est non indispensable pour le traitement des médias O_i $(i = 1..p)$; pour modéliser le schéma d'allocation de rs envers les unités conflictuelles, nous avons besoin d'introduire des places spéciales que nous appellons "*places ressource*" représentées par des cercles à double lignes. Plus précisément, le schéma d'allocation d'une ressource rs est modélisé par le couple $(fr(rs),\ bs_i(rs)\ i = 1..p)$, où :

- $bs_i(rs)\ i = 1..p$ est un ensemble de places ressource, tel que chaque place $bs_i(rs)$ est associée à l'unité O_i et dénote une fois marquée que rs est utilisée pour le traitement de O_i.
- $fr(rs)$ est une place ressource, n'appartenant à aucune unité, et dénote une fois marquée que rs est restituée ;

  Remarque : Pour un marquage donné, seulement une seule place parmi l'ensemble $\{fr(rs), bs_1(rs), ..bs_p(rs)\}$ doit être marquée.

Comme le jeton contenu dans une place ressource est *faible*[3] (comparé à celui d'une place standard), nous avons besoin d'utiliser des arcs spéciaux pour exprimer ces sémantiques. Pour cet effet, nous exploitons un nouveau mécanisme appelé "*hyperarc préempteur*" [14][16] permettant de distinguer la sémantique d'une place ressource de celle d'une place standard dans le sens qu'une transition ne nécessite pas la disponibilité du jeton dans sa place ressource d'entrée pour être sensibilisée (i.e., seuls les jetons dans les places standards sont requis). L'*hyperarc préempteur* permet de connecter en entrée un ensemble de places ressource à une seule transition. Il utilise deux types d'arcs : un *arc fort* représenté par un trait continu orienté ; et un *arc de violation* représenté par un trait orienté en pointillés. Par conséquent, étant donné un *hyperarc préempteur*, la place ressource connectée à travers un *arc de violation* est appelée "*place ressource de violation*", tandis que la place reliée par l'arc fort, est appelée "*place ressource forte*". La représentation graphique de ce mécanisme illustrée par la *Figure* 3.2, modélise le media O_1 requérant la ressource rs, conflictuellement avec p autres médias.

De là, dans le respect de ces observations, et dans le but de formaliser cette proposition, nous définissons de nouveaux concepts d'événements lors du tir d'une transition :

- Un *événement fort* noté t dénotant que la transition t est tirée et tel que s' il existait un *hyperarc préempteur* connecté, alors sa *place ressource forte* est nécessairement marquée.
- Un "*événement de violation passive*" noté t^* définissant la situation où la transition t est tirée telle qu'il y a un *hyperarc préempteur* dont toutes ses places ressource connectées sont non marquées.
- Un " *événement de violation active*" noté ${}^*t$ généré lorsque la transition est tirée telle qu'il y a un *hyperarc préempteur* dont sa *place ressource forte* est non marquée et au moins une *place ressource de violation* est marquée.

Ainsi, le tir d'une transition t nécessite quet t soit auparavant sensibilisée [4], et que ses contraintes de temps soient consistantes au regard des contraintes des autres transitions sensibilisées.

Remarque : En conformité avec les hypothèses émises précédemment stipulant qu'une présentation ne peut requérir au plus qu'une seule ressource, nous assumons que chaque *hyperarc préempteur* peut contenir au plus un arc fort et que chaque place ressource est 1-bornée. Par ailleurs, pour être cohérent avec ces mêmes hypothèses, nous admettons que l'occurrence d'un *événement de violation passive* t^* ne peut produire de jeton dans ses places ressource en sortie.

Par exemple, la *Figure* 3.2 représente un schéma d'allocation où rs est requise par p différentes unités, où la plus haute priorité est associée à O_1. Partant de ces hypothèses, pour modéliser la sémantique faible de la place $fr(rs)$ envers la transition $B(O_1)$, nous utilisons un arc fort pour connecter les deux éléments. Chaque place $bs_i(rs)\{\ i = 2..p\}$ est ensuite reliée à la transition $B(O_1)$ en utilisant un arc de violation indiquant que l'unité O_1 a le droit de violer (préempter) la ressource rs lorsqu'elle est détenue par l'une des unités conflictuelles $O_i\{\ i = 2..p\}$; ce qui déclenche la génération de l'événement de violation active ${}^*B(O_1)$ [5].

Remarque : La modélisation de la gestion des conflits nécessite d'associer à chaque unité O_i une priorité fixe notée $Pr(O_i)$, et de là, toutes les transitions impliquées dans la modélisation d'un

3. Puisque nous admettons que la disponibilité du jeton dans la place ressource n'est pas indispensable à la sensibilisation de la transition mais néanmoins sa présence (ou non), devrait être exploitée pour distinguer le type d'événement à générer lors du tir de la transition.

4. Toutes les places connectées à travers des arcs standards sont marquées.

5. La place $fr(rs)$ est non marquée et une des places ressource $bs_i(rs)\{\ i = 2..p\}$ l'est.

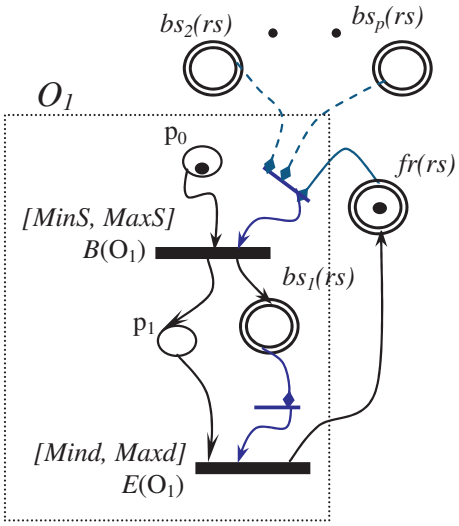

FIGURE 3.2 – Modélisation des contraintes ressource d'une présentation de base

même objet auront la même priorité. Autrement dit, $Pr(O_i)$ désigne aussi la priorité affectée aux transitions $B(O_i)$ et $E(O_i)$.

Modélisation des conflits de ressource

Pour expliciter la sémantique des hyperarcs préempteurs, nous montrons ci-après comment modéliser les contraintes de deux médias O_1 et O_2 en conflit pour la ressource rs. Nous généralisons ensuite cette modélisation pour plus de deux médias conflictuels, en se reférant aux fonctionnements des principaux lecteurs. Nous commençons par donner la spécification en considérant un fonctionnement épousant la sémantique du lecteur *Real one*[81], avant de considérer celui du lecteur *Ambulant*[27]. Pour cet effet, nous supposons que le média O_1 est moins prioritaire que O_2 (i.e, $Pr(O_1) \prec Pr(O_2)$) ; ce qui implique que O_1 peut se voir retirer la ressource rs par O_2 durant son exécution.

Par ailleurs, il est à rappeler que les fonctionnements différent selon les lecteurs. Par exemple, pour le lecteur *Real one* [81], si le média O_1 a été préempté en cours de son traitement ou qu'il se voit être dépourvu de sa ressource à son commencement, il ne peut la récupérer par la suite, même si elle est libérée par O_2 avant l'achèvement de l'exécution de O_1. Cependant, le fonctionnement du lecteur *Ambulant* [27] présente une toute autre sémantique ; à savoir qu'une ressource indisponible durant une partie du traitement de O_1, pourrait être récupérée et exploitée, si elle est libérée à temps (avant que l'exécution de O_1 ne se termine).

Modélisation de la gestion des conflits selon la sémantique du lecteur *Real One* [81]
La modélisation de deux médias O_1 et O_2 conflictuels pour une ressource rs, où nous assumons une exécution opérant sur un lecteur adoptant le fonctionnement du lecteur *Real one*, peut être spécifiée par le réseau de la *Figure* 3.3. Pour modéliser le schéma d'allocation, nous utilisons quatre hyperarcs préempteurs :

– Deux sont connectés aux transitions $B(O_1)$ et $B(O_2)$ i.e., Les médias O_1 et O_2 nécessitent la disponibilité de la ressource rs à leurs commencements [6]. Sinon, si la ressource rs est indisponible pour l'un des deux médias [7], alors seul O_2 qui est prioritaire (modélisée par l'*arc de violation* relié à $bs_1(rs)$), peut retirer [8] rs à O_1.
– Les deux autres hyperarcs préempteurs sont reliés aux transitions $E(O_1)$ et $E(O_2)$ dénotant que les exécutions de O_1 et O_2 nécessitent de détenir la ressource rs durant la totalité de leurs présentations. Donc, si la place ressource $bs_1(rs)$ est non marquée lors du tir de $E(O_1)$, un événement de *violation passive* est généré, indiquant que la ressource rs a été retirée de O_1 par O_2 durant son exécution. Notons que dans l'exemple de la *Figure* 3.3, seule $E(O_1)$ peut être préemptée puisque O_2 est prioritaire. Pour modéliser le cas où aucun média n'est prioritaire (i.e, $Pr(O_1) = Pr(O_2)$) nous avons besoin de rajouter un *arc de violation* partant de la place ressource $bs_2(rs)$ à la transition $B(O_1)$. En clair, les deux médias peuvent se préempter mutuellement.

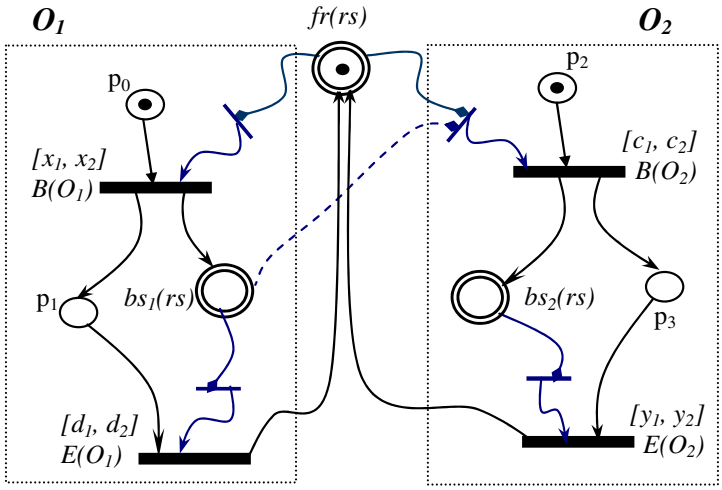

FIGURE 3.3 – Modélisation des conflits de ressource selon la sémantique du lecteur *Real one*

Afin d'expliciter la sémantique du mécanisme de préemption proposé, nous énumérons ci-après les séquences d'événements, possibles dérivées à partir de la modélisation donnée en *Figure* 3.3.

– $B(O_1) \rightarrow E(O_1) \rightarrow B(O_2) \rightarrow E(O_2)$ et $B(O_2) \rightarrow E(O_2) \rightarrow B(O_1) \rightarrow E(O_1)$: Les deux scénarios modélisent le cas où les exécutions des médias O_1 et O_2 ne se chevauchent pas (i.e., les présentations se réalisent en séquence). Dans ces deux configurations, les transitions sont tirées en générant des *événements forts*, puisque les places ressource fortes sont marquées lors du tir de chaque transition (voir *Figure* 3.4.*a*).
– $B(O_1) \rightarrow {}^*B(O_2) \rightarrow E(O_1)^* \rightarrow E(O_2)$: Dans ce cas, O_1 est le premier à débuter (i.e. l'*événement fort* $B(O_1)$ est donc généré puisque $fr(rs)$ est marquée) ensuite, O_2 recours à violer la ressource de O_1 pour initier sa propre exécution [9]. Cependant, pour exprimer que O_1 a eu sa ressource violée durant son exécution, l'*événement de violation passive* $E(O_1)^*$ est généré, puisque $bs_1(rs)$ n'est plus marquée. Finalement, le tir de $E(O_2)$ génère un *événement*

6. Un événement fort est généré si la place $fr(rs)$ est marquée.
7. La place ressource $fr(rs)$ est non marquée et l'une des places ressources $bs_1(rs)$ ou $bs_2(rs)$ est marquée.
8. En clair, *l'événement de violation active* ${}^*B(O_2)$ est généré puisque la place *ressource de violation* $bs_1(rs)$ est marquée.
9. L'événement de violation active ${}^*B(O_2)$ est généré car $fr(rs)$ est non marquée alors que $bs_1(rs)$ l'est.

fort car $bs_2(rs)$ devient marquée suite au tir de $B(O_2)$. Enfin, le marquage de $fr(rs)$ est restauré [10] seulement après le tir de $B(O_2)$; car le tir de $E(O_1)$ comme *événement de violation passive* ne produit pas de jeton dans la place $fr(rs)$ (voir *Figure* 3.4.*b*).

- $B(O_1) \rightarrow {}^*B(O_2) \rightarrow E(O_2) \rightarrow E(O_1)^*$: Ce scénario est le même que le précédent, mais en considérant ici que O_2 se termine en premier. Comme, $bs_1(rs)$ devient non marquée après le tir de $B(O_2)$ en mode violation active ; l'événement $E(O_1)^*$ est donc généré dénotant que le média O_1 a eu sa ressource violée par O_2 durant son exécution (voir *Figure* 3.4.*c*).
- $B(O_2) \rightarrow B(O_1)^* \rightarrow E(O_1)^* \rightarrow E(O_2)$ et $B(O_2) \rightarrow B(O_1)^* \rightarrow E(O_2) \rightarrow E(O_1)^*$: Les deux scénarios décrivent le cas où O_2 débute en premier. Par conséquent, le média O_1 est dans l'incapacité d'acquérir la ressource pour son exécution i.e., les *événements de violtion passive* $B(O_1)^*$ et $E(O_1)^*$ sont générés puisque $fr(rs)$ et $bs_1(rs)$ sont non marquées, modélisant le fait que O_1 ne peut obtenir la ressource rs à son début, et donc pour la totalité de son exécution, comme le stipule la sémantique du lecteur *Real One.* (voir *Figure* 3.4.*d*).

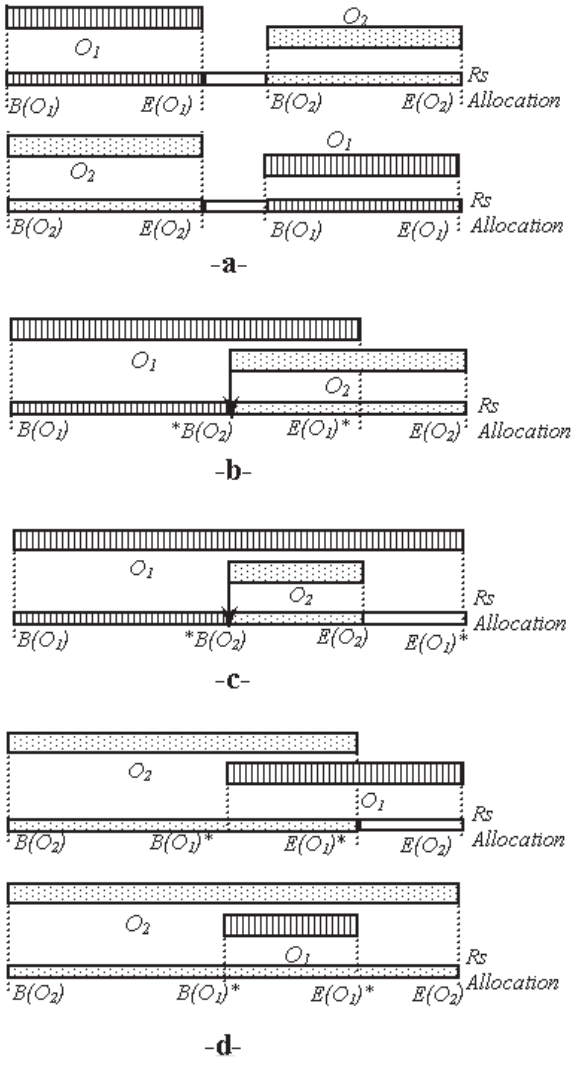

FIGURE 3.4 – Schémas d'allocation de ressource selon le lecteur *Real one* .

Notons que le mécanisme de préemption décrit précédemment peut être étendu aux cas où plus de deux objets sont en conflit pour la même ressource. Par exemple, soit O_1, O_2,..., O_p un ensemble

10. La ressource rs redevient disponible.

de d'unités modélisant un ensemble de médias en conflit pour la même ressource rs, respectant l'ordre de priorité suivant $Pr(O_1)... \prec Pr(O_i)... \prec Pr(O_p)$. Donc, pour modéliser le cas général, il suffit de considérer le schéma d'allocation de ressource donné par $(fr(rs),\ bs_i(rs)\ i = 1..p)$ où $fr(rs)$ est marquée initialement et connectée en entrée à toutes les transitions $B(O_i)$ $(i = 1..p)$ en utilisant des arcs forts [11]. Aussi, chaque place ressource $bs_i(rs)$ est connectée en sortie à la transition $B(O_i)$ grâce un arc standard, et en entrée à la transition $E(O_i)$ en utilisant un arc fort. Ainsi, si $bs_i(rs)$ est marquée, ceci dénote que la ressource rs est détenue par l'objet O_i. De plus, pour modéliser le mécanisme de préemption, nous avons besoin de relier chaque transition $B(O_j)$ $j = 1..p$ à toutes les places ressources $bs_i(rs)$ tel que $i \prec j$, en utilisant des arcs de violation. Chaque arc de violation donne le droit à la transition $B(O_j)$ de violer les transitions moins prioritaires lorsque la ressource requise rs est détenue par l'une d'elles, C.a.d lorsqu'il y a une place ressource marquée parmi l'ensemble $\{bs_i(rs)\ i = 1..j-1\}$. Sinon, si la ressource est détenue par un objet plus prioritaire $(i \succ j)$, $B(O_j)$ ne peut obtenir la resource puisqu'il n'y a pas d'arc de violation connectant la place ressource $bs_i(rs)$ à la transition $B(O_j)$.

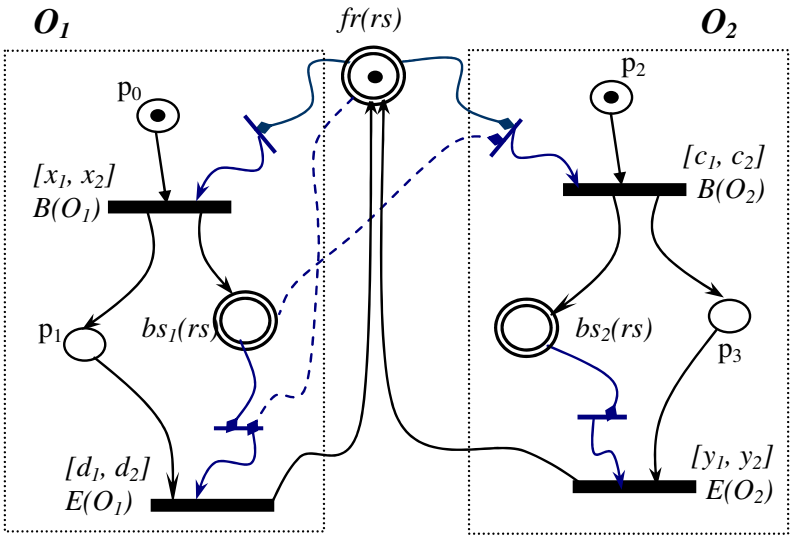

FIGURE 3.5 – Modélisation des contraintes ressource selon la sémantique du lecteur *Ambulant*

Modélisation de la gestion des conflits selon la sémantique du lecteur *Ambulant* [27] Si nous devons modéliser maintenant la gestion des conflits telle qu'elle est réalisée par le lecteur *Ambulant* [27], il suffirait de considérer le même fonctionnement que celui de *Real one*, mais en autorisant le média O_1 à recouvrer la ressource rs si cette dernière (détenue jusqu'à lors par O_2) est libérée dans les temps. Pour ce faire, il suffit de considérer la même modélisation que dans la *Figure* 3.3. mais en rajoutant un arc de violation reliant la place ressource $fr(rs)$ à la transition $E(O_1)$, comme décrit dans la *Figure* 3.5. Nous pouvons dériver à partir de cette spécification les séquences d'événements suivantes :

- $B(O_1) \rightarrow E(O_1) \rightarrow B(O_2) \rightarrow E(O_2)$ et $B(O_2) \rightarrow E(O_2) \rightarrow B(O_1) \rightarrow E(O_1)$: Ces deux exécutions sont similaires à celles décrites dans le cas d'une modélisation adoptant le fonctionnement du lecteur *Real one* (voir *Figure* 3.4.*a*).

11. Ce qui dénote que les objets O_i $(i = 1..p)$ nécessitent la disponibilité de rs pour être présentés.

- $B(O_1) \rightarrow {}^*B(O_2) \rightarrow E(O_1)^* \rightarrow E(O_2)$: Ce scénario décrit la même trace d'excution que dans le cas du lecteur *Real one* (voir *Figure* 3.4.*b*).
- $B(O_1) \rightarrow {}^*B(O_2) \rightarrow E(O_2) \rightarrow^* E(O_1)$: Cette séquence d'événements diffère de celle donnée dans le contexte du lecteur *Real one* pour le même scénario. Plus clairement, lors du tir de la transition $B(O_2)$ en violation active, la place $bs_1(rs)$ devient non marquée. Ensuite, le tir de $E(O_2)$ dénotant la fin de l'exécution de O_2 va restituer le jeton dans la place $fr(rs)$. La présence de ce dernier va valider l'arc de violation de la transition $E(O_1)$ générant l'événement ${}^*E(O_1)$ lors de son tir.
- $B(O_2) \rightarrow B(O_1)^* \rightarrow E(O_1)^* \rightarrow E(O_2)$: même fonctionnement que dans le cas du lecteur *Real one*.
- $B(O_2) \rightarrow B(O_1)^* \rightarrow E(O_2) \rightarrow^* E(O_1)$: La différence en terme de séquence d'événements avec la modélisation de la *Figure* 3.4.*d* est que l'événement de violation active ${}^*E(O_1)$ est généré à la place de $E(O_1)^*$ en raison du marquage de $fr(rs)$. Ainsi la combinaison des événements $B(O_1)^* \rightarrow {}^*E(O_1)$ dénote ici la non disponibilité de la ressource rs au lancement de O_1 et son obtention en cours d'exécution, après sa libération par O_2.

De même, cette modélisation pourrait être généralisée dans le cas où plusieurs médias sont conflictuels pour la même ressource rs. Il suffit de considérer le même cas général décrit pour le lecteur *Real one* en rajoutant pour toutes les unités O_i $(i = 1..p-1)$, un *arc de violation*[12] allant de la place $fr(rs)$ à la transition $E(O_i)$.

Cependant, à ce stade cette modélisation est incomplète ; car nous avons besoin d'un mécanisme supplémentaire permettant d'exprimer la priorité. Ce mécanisme doit avoir la capacité d'autoriser seulement la transition appartenant à l'unité prioritaire[13] (parmi celles en attente de la libération de rs), de consommer le jeton dans la place ressource $fr(rs)$. Pour ce faire, nous définissons un nouveau mécanisme que nous appelons "*arc de privation de ressource*" noté $(t_j \trianglerighteq t_i)$. Ce dernier permet de connecter deux transitions et est representé par un arc orienté en trait dicontinu partant de la transition t_j à la transition t_i. Par conséquent, la sémantique de cet arc opère comme suit : si t_i est franchissable, alors t_i est privée de consommer les jetons de ses places ressources (lors de son tir), aussi longtemps que t_j sera *franchissable*. Nous disons alors que t_i est *privée de ressource*, et cette privation cessera une fois qu'il n y aura plus aucune autre transition sensibilisée connectée à la transition t_i par un "*arc de privation*". De plus, il est à noter que le tir de t_i en étant "*privée de ressource*" génère un *événement de violation passive* t_i^*.

Pour revenir à la modélisation du cas général, nous aurons besoin donc de connecter chaque couple de transitions $(E(O_j) \trianglerighteq E(O_i))$ tel que $(i \prec j = 1..p-1)$, par un *arc de privation* $(E(O_j) \trianglerighteq E(O_i))$.

Par exemple, considérons la modélisation du cas décrit dans la *Figure* 3.6, où trois médias O_1, O_2 et O_3 sont en conflit pour la même ressource tel que $Pr(O_1) \prec Pr(O_2) \prec Pr(O_3)$. Nous assmons dans cela le scénario où O_1, O_2 et O_3 se chevauchent temporellement, tel que O_3 commence et se termine en premier. Ainsi, après la libértion de rs[14], $E(O_1)$ sera empêchée de consommer

12. Il modélise ainsi la possibilité donnée à O_i de récupérer la ressource rs durant son exécution.
13. Si deux médias sont en attente de libération d'une ressource. Seul le média qui détient la priorité pourra la récupérer pour le reste de son temps d'exécution.
14. Le tir de $E(O_3)$ va reproduire le jeton dans la place $fr(rs)$ sensibilisant ainsi les deux arcs de violation connectés aux transitions $E(O_2)$ et $E(O_1)$.

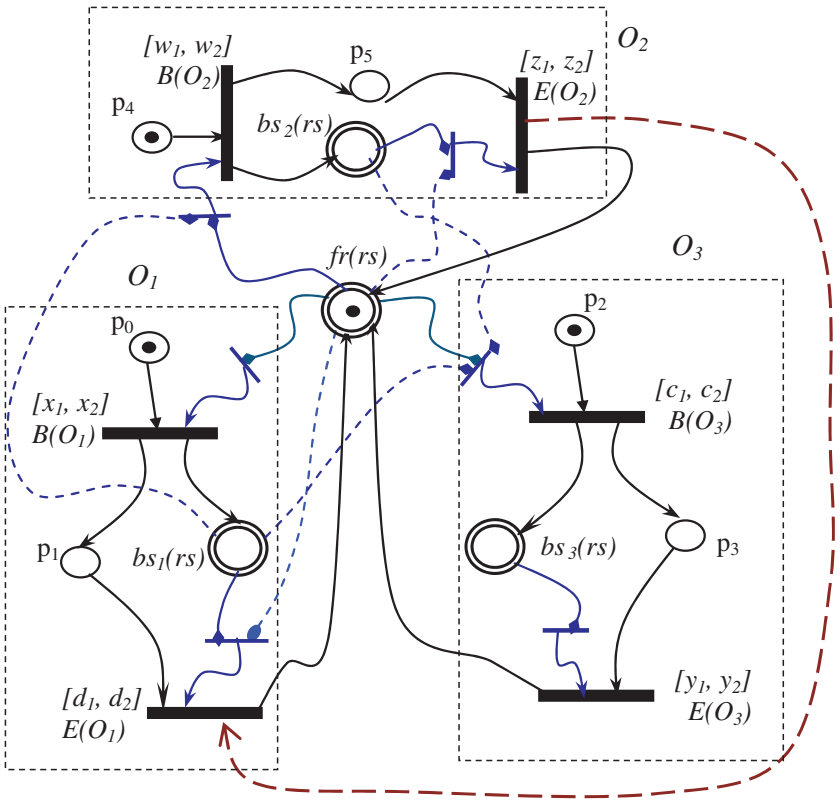

FIGURE 3.6 – Modélisation du cas général d'un conflit de ressource selon la sémantique du lecteur *Ambulant*

le jeton dans $fr(rs)$ aussi longtemps que la transition $E(O_2)$ reste sensibilisée [15]. Par conséquent, si O_1 se termine avant O_2, la génération de *l'événment de violation passive* $E(O_1)^*$ dénotera que O_1 n'a pas pu obtenir la ressource pour la fin de sa présentation, alors que la génération de l'*événement de violation active* $E(O_2)^*$ dénotera que O_2 ait pu recupérer la ressource pour la fin de son exécution. Au fait, le seul scénario permettant à O_1 de récupérer la ressource, est de terminer en dernier ; ce qui est illustré par la génération de l'*événement de violation active* [16] $^*E(O_1)$.

3.3 Modélisation d'un lien

Un **lien** est un ancre associé à un média de base et représente le seul mécanisme par lequel un utilisateur peut interagir avec une présentation multimédia. De plus, ces interactions permettent d'améliorer la qualité et la richesse d'expression des documents multimédias gràce à la flexibilité et l'interactivité qu'ils procurent lors de leurs présentations. Cependant, en raison de leur caractère non déterministe, ils induisent des comportements complexes qui mènent souvent à des disfonctionnements. Par conséquent, une modélisation correcte et exhaustive de leur comportements est très importante et devra être entreprise minutieusement, afin de pouvoir détecter ces erreurs lors de la phase d'analyse.

Par exemple, dans le contexte du langage *SMIL*, un lien est défini par une balise (<Anc \>) associée à la déclaration d'un media de base. De plus, un intervalle $[MinA, MaxA]$ délimitant

15. Puisqu'il y a un *arc de privation* partant de la transition $E(O_2)$ à $E(O_1)$.
16. La place ressource $fr(rs)$ est de nouveau marquée après le tir de $E(O_2)$ et la transition $E(O_1)$ n'est plus *privée de ressource.*

les contraintes de temps durant lesquelles le lien doit être affiché peut être spécifié. De la, si l'utilisateur *clique* sur le lien, la présentation ciblée par le lien est enclenchée et le comportement de l'ancienne présentation peut subir plusieurs sorts [90] :

- La présentation est arrêtée et remplacée par la nouvelle.
- La présentation continue de se produire en parallèle avec la nouvelle, mais dans un contexte diffèrent.
- La présentation est suspendue et sera reprise une fois la nouvelle présentation achevée.

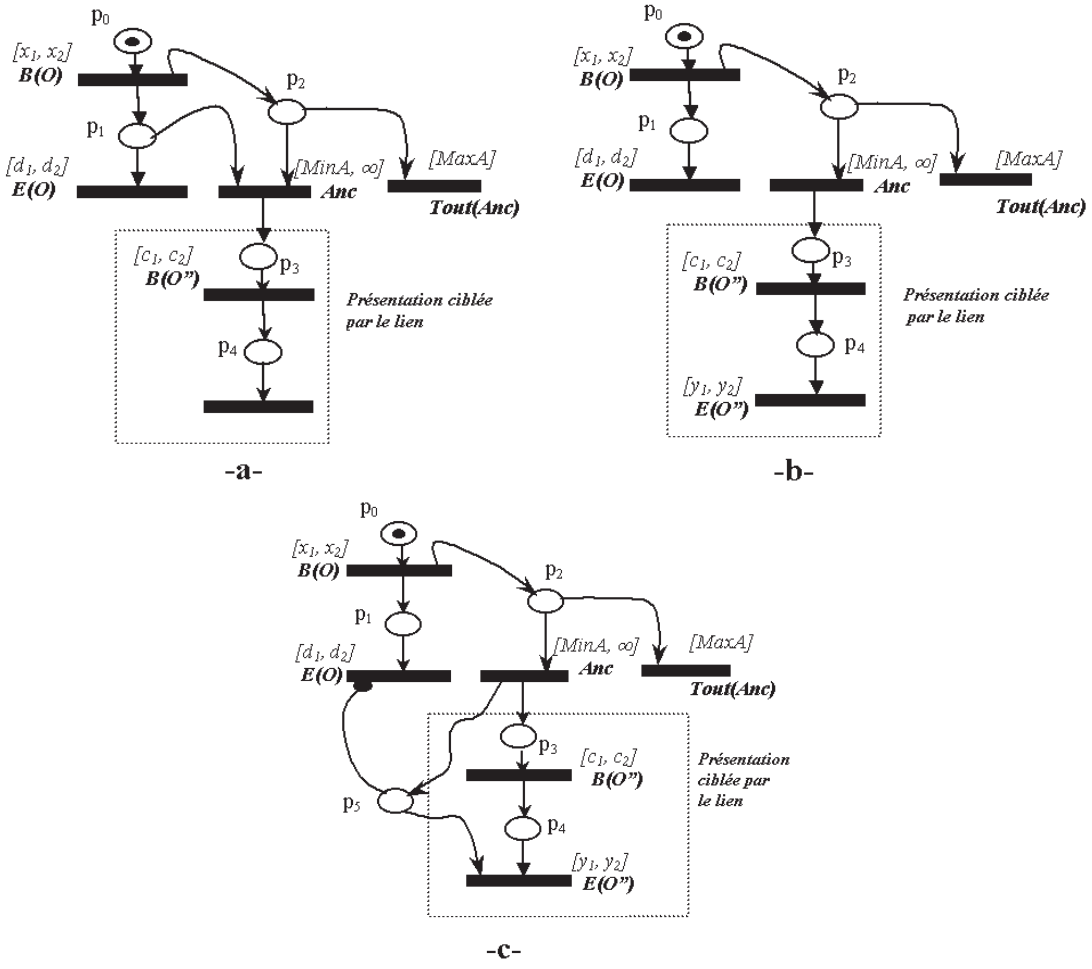

FIGURE 3.7 – Modélisation d'un lien

Le premier cas peut être modélisé par le réseau de la *Figure* 3.7.*a*, où l'unité associée au lien O est reliée à l'unité ciblée par le lien O", à travers la transition *Anc*. Cette dernière modélise l'interaction de l'utilisateur qui *pourait* se produire seulement après $MinA$. Donc, la modélisation de l'interaction utilisateur est réalisée en associant l'intervalle $[MinA, \infty]$ à la transition *Anc*. Ainsi, si la transition *Anc* est tirée, le jeton dans la place p_1 est consommé impliquant l'arrêt de O et l'activation de la nouvelle l'unité O". Par ailleurs, si l'interaction ne peut se produire avant $MaxA$, alors la transition $Tout(Anc)$ dénotant le timeout associé à la disponibilité du lien est tirée.

Remarque : Dans le respect de la sémantique forte d'un $RdPT$, associer un intervalle $[MinA, MaxA]$ à la transition *Anc* force cette dernière à tirer lorsque la progression du temps atteint la borne supérieure $MaxA$. Cette sémantique ne correspond pas avec le fait que l'interaction

utilisateur est un événement *faible ;* pouvant ne pas avoir lieu à l'instant $MaxA$. Ce qui a motivé l'utilisation de deux transitions Anc et $Tout(Anc)$ pour modéliser cet événement.

Le second cas peut être modélisé de la même manière que le premier, mais en supprimant l'arc reliant la place p_1 à la transition Anc (voir $Figure$ 3.7.*b*).
Finallement, le troisième cas est plus complexe puisqu'il implique l'utilisation d'un mécanisme ayant la capacité d'arrêter la progression du temps pour certaines transitions, et de la reprendre par la suite. Pour ce faire, nous considérons le mécanisme des chronomètres [46] qui associe pour chaque transition dans le modèle une horloge qui peut être enclenchée, suspendue, reprise voire arrêtée. Aussi, comme dans [83] des arcs standards supplées par des arcs inhibiteurs sont utilisés pour décider de l'action à appliquer sur chaque horloge.

Usant de ce mécanisme, la présente configuration peut être modélisée par le réseau de la $Figure$ 3.7.*c*, où la place p_5 est ajoutée pour déclencher l'inhibition de $E(O)$ une fois marquée. L'inhibition de $E(O)$ provoque la suspension du temps du chronomètre, et dont la reprise est activée lorsque la transition $E(O")$ sera tirée [17].

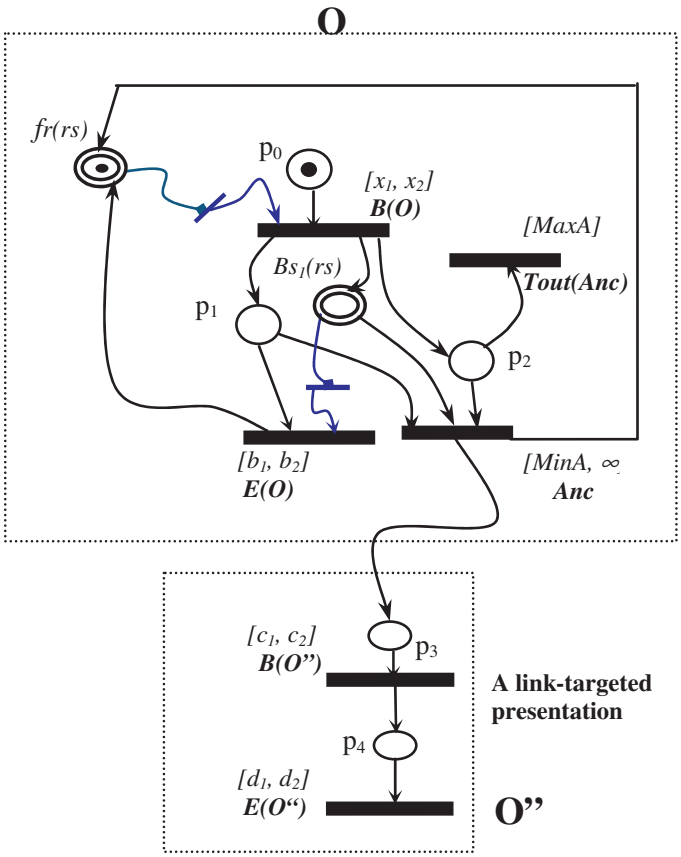

FIGURE 3.8 – Modélisation des contraintes ressource d'un lien dans le cas d'un fonctionnement épousant celui du lecteur *Real One.*

Par ailleurs, il est important de considérer la contrainte ressource lors de la modélisation des interactions utilisateur. Plus clairement, le lien nécessite la disponibilité de la ressource region [18] pour être accessible, et par conséquent permettre à l'utilisateur d'intervenir. Cependant, si la

17. Le tir de $E(O")$ consommera le jeton dans la place p_5, supprimant ainsi l'inhibition de $E(O)$).
18. La fenêtre dédiée à l'affichage du média auquel le lien est associé.

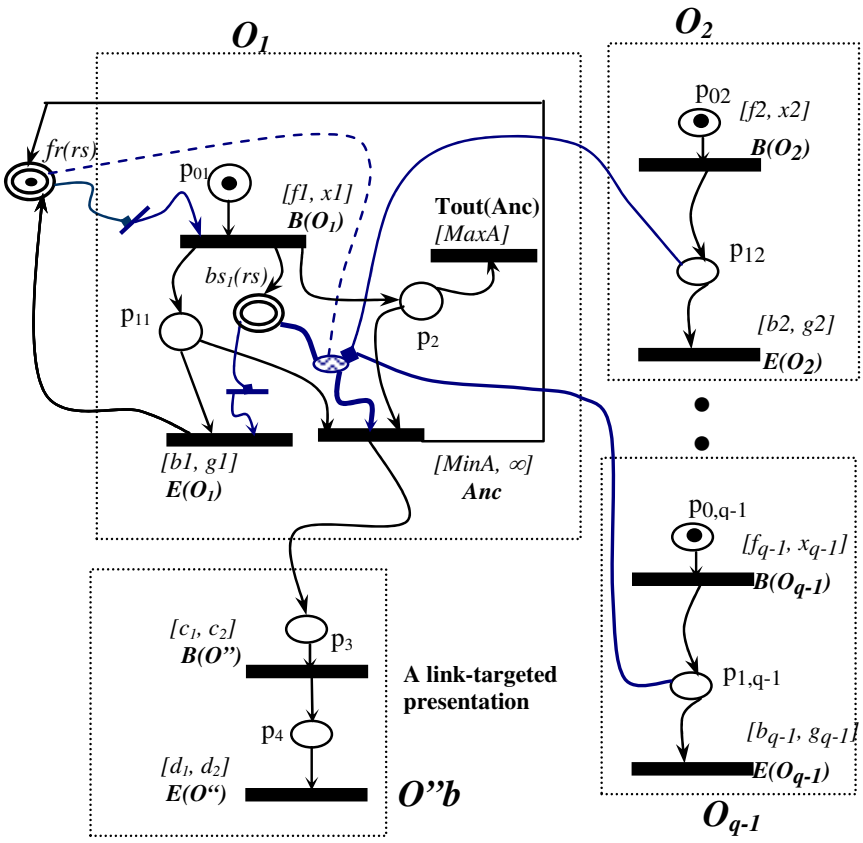

FIGURE 3.9 – Modélisation des contraintes ressource d'un lien dans le cas d'un fonctionnement semblable à celui du lecteur *Ambulant.*

region est utilisée par un média durant le temps de visulisation du lien [19], alors l'utilisateur ne pourrait interagir avec la présentation. Dans un contexte de fonctionnement épousant celui du lecteur *Real one*, ce comportement est traité en connectant la place $bs_1(rs)$ en entrée et $fr(rs)$ en sortie de la transition *Anc* en utilisant des arcs standards [20], comme décrit dans la $Figure$ 3.8. Par conséquent, la transition *Anc* peut tirer seulement si la place ressource $bs_1(rs)$ est marqué et son tir va consommer le jeton présent dans sa place ressource $bs_1(rs)$ et le reproduire dans $fr(rs)$. Autrement, si $bs_1(rs)$ est non marquée, celà dénote que la région rs est non détenue présentement par O et aussi bien le média que le lien ne sont pas affichés ; par conséquent l'utilisateur ne peut plus interagir puisque la région est non *recouvrable* selon le fonctionnement du lecteur *Real one.*

Par ailleurs, la modélisation de la contrainte ressource dans le contexte d'un fonctionnement épousant celui du lecteur *Ambulant*, nécessite d'introduire un mécanisme plus complexe pour conditionner la disponibilité ou non de la région pour l'affichage du lien. Au fait, nous avons besoin d'autoriser l'occurrence d'une interaction seulement lorsque la région n'est pas détenue par un média de plus haute priorité et en conflit pour la même région que le média associé au lien noté O_1. Soit $\{O_2..O_q\}$ l'ensemble des médias (et par extrapolation des unités), prioritaires et en conflict avec O_1. Pour que l'interaction puisse se produire, il faut soit que la région soit détenue

19. La place $bs_1(rs)$ est non marquée.
20. Lorsque la transition est reliée en entrée à une place resource via un arc standard, la disponibilité du jeton dans la place resource conditionne le tir de la transition mais il est non nécessaire pour sa sensibilisation.

par O_1 dès le début de son exécution [21] ou que la région rs ait été indisponible puis libérée [22] telle que aucun autre média $\{O_2..O_{q-1}\}$ ne soit en cours d'execution [23].

Pour cet effect, nous introduisons un nouveau mécanisme appelé "*hyperarc de franchissement*" connectant deux places ressources et un ensemble de places standards à une transition. Ce dernier utilise trois types d'arc :

- un arc de "*franchissement fort*" connecté à une place ressource et représenté par un trait continu ;
- un arc de "*franchissement faible*" connecté à une place ressource et représenté par un trait en pointillés ;
- et enfin des *arcs de conditionnement* tel que chacun est relié à une place standard appartenant à une unité différente.

L'hyperarc de franchissement autorise le tir de la transition, si la place ressource connectée à travers l'*arc de franchissement fort* est marquée, ou bien que la place ressource connectée à travers l'*arc de franchissement faible* est marquée et qu'aucune place standard connectée via un *arc de conditionnement* ne soit marquée.

Pour revenir à notre cas, il suffit comme décrit dans la *Figure* 3.9 de connecter la place $bs_1(rs)$ à la transition *Anc* via l'*arc fort* de l'*hyperarc de franchissement* [24], et de relier la place ressource $fr(rs)$ via l'*arc faible* [25] et les *places centrales* $p_{12}..p_{1q-1}$ respectivement des unités $O_2..O_{q-1}$ via des *arcs de conditionnement* [26] à la transition *Anc*. Par conséquent, si la transition *Anc* est sensibilisée et que son *hyperarc de franchissement* est validée, son tir provoquera la consommation du jeton présent soit dans la place ressource $bs_1(rs)$ ou dans de la place ressource $fr(rs)$ et produira un jeton dans la place ressource $fr(rs)$.

Remarque : Il est à noter que les approches traditionnelles ne traient pas les contraintes ressources. Il en résulte que le processus de modélisation admet que l'interaction utilisateur puisse se produire même si la région dédiée est indisponible durant le période d'affichage du lien. Ceci induit des scénarios supplémentaires qui n'ont pas d'existence réelle et pouvant par conséquent fausser les résultats obtenus lors de la phase de vérification. Cet aspect est explicité en *Chapitre VI.*

3.4 Modélisation des contraintes de synchronisation

En plus des précédentes contraintes, le modèle doit être capable de traiter les présentations requérant des contraintes de synchronisation. Ainsi, la synchronisation des événements (appelée ici $rendez-vous$) est réalisée par le tir simultané d'un ensemble de transitions selon un schéma prédéfini. Par exemple, *Senac et autres* [53][87] ont proposé différents schémas de synchronisation

21. La place ressource $bs_1(rs)$ est marquée.
22. La place ressource $fr(rs)$ est de nouveau marquée.
23. Aucune transition $E(O_i)$ $i=2..q-1$ ne doit être sensibilisée, ou plus précisément aucune place centrale p_{1i} relative à l'une des uniés O_i $i=2..q-1$ n'est marquée.
24. Si l'*arc fort* est validé ($bs_1(rs)$ est marquée), cela dénote de la disponibilité de la ressource région depuis le début de l'exécution de O_1 ; la région n'a pas été retirée de O_1 jusquà lors.
25. Si l'*arc faible* est activé ($fr(rs)$ est marquée), cela dénote que la région a été restituée après avoir été indisponible jusquà lors pour l'exécution de O_1 ; le lien n'était pas accessible.
26. Chaque *arc* conditionne le franchissement de *Anc* lorsque $fr(rs)$ est marquée. C'est à dire il ne faut pas qu'il y ait un autre média $O_i(i = 2..q-1)$ en cours d'exécution (les places centrales p_{1i} doivent être non marquées).

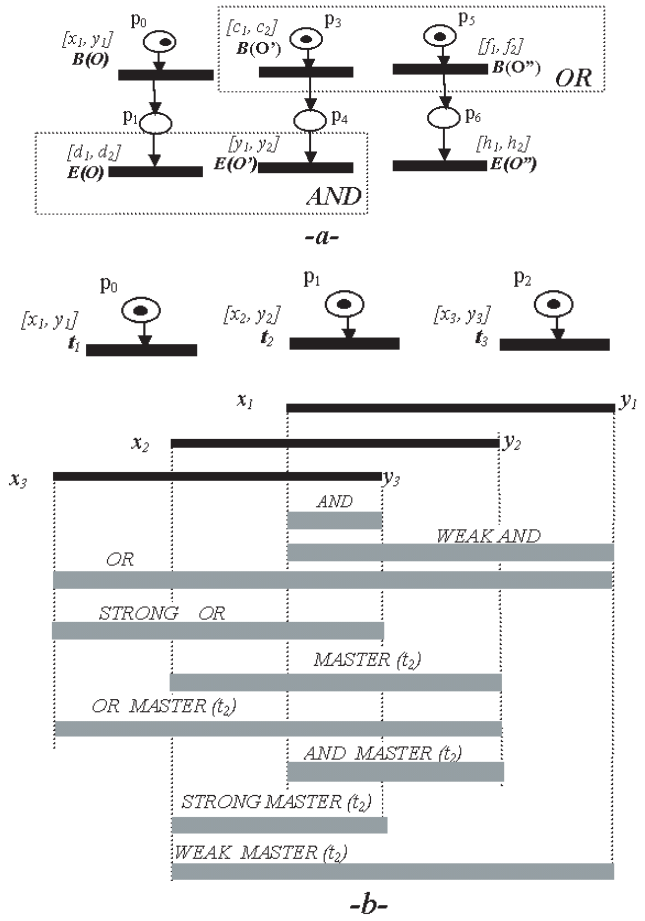

FIGURE 3.10 – Les règles de synchronisation dans un rendez-vous

entre plusieurs actions se déroulant en parallèle. Cette proposition déduite de l'algèbre d'*Allen* [23] offre trois stratégies de synchronisation de base.

- Stratégies de synchronisation dynamiques And, $Weak-And$, $And-Master$ dirigées par le dernier processus qui atteint sa borne supérieure.
- Stratégies de synchronisation dynamiques, Or, $Strong-Or$, $Or-Master$ dirigées par le premier processus qui atteint sa borne inférieure..
- Stratégies de synchronisation statiques $Master$, $Strong-Master$, $Weak-Master$ dirigées par le processus sélectionné.

Ces trois stratégies que nous adoptons aussi dans notre modèle, permettent la synchronisation parallèle d'un ensemble de transitions sensibilisées en offrant une plus grande puissance d'expression que dans un $TSPN$ [27][53]. (voir *Figure* 3.10.*b*) :

Par exemple, le réseau de la *Figure* 3.10.*a* modélise trois médias O, O' et O" devant se synchroniser comme suit : Les débuts de O" et O' doivent se synchroniser selon le schéma OR ; ce qui implique que la transition parmi $B(O')$ et $B(O")$ qui tire en premier provoque le tir de l'autre, même si les contraintes de temps de cette dernière sont non satisfaites. De plus, on impose que la fin des présentations de O et O' doit avoir lieu de manière synchrone ; ce qui implique que $E(O)$

27. Un processus modélisé par une transition peut participer à plusieurs synchronisations et peut détenir des ressources propres. Cet aspect ne peut être modélisé avec un TSPN où l'arc modélisant le processus ne peut participer qu'à une seule synchronisation et ne peut détenir de ressources.

peut tirer seulement si $E(O')$ peut être tirée aux mêmes instants et vice versa.

Remarque : Il est à remarquer que seules les transitions n'ayant pas été inhibées [28], ou *interdites de tir* [29] peuvent se synchroniser. Par conséquent, un rendez-vous dont l'une des transitions est *non tirable (inhibée* ou *interdite de tir),* ne peut être franchi. De plus, un rendez-vous peut se réaliser même si une ou plusieurs de ses transitions sont violées, car la disponibilité des ressources non critiques est non nécessaire pour la réalisation d'une synchronisation d'événement. Toutefois, la sensibilisation d'un rendez-vous requiert que toutes ses transitions soient sensibilisées et non conflit pour les places standards.

Par ailleurs, nous assumons que les transitions à *synchroniser* pour un même rendez-vous sont tirées simultanément et de manière indivisible. Par conséquent, nous devons adopter un ordre de priorité pour déterminer quelles transitions obtiennent les ressources conflictuelles lors de la réalisation du rendez-vous. Cette règle de priorité (qui suit le même ordre de priorité Pr défini précédemment), permet de résoudre le conflit entre les transitions conflictuelles participant à un même rendez-vous.

Par exemple, considérons le réseau de la *Figure* 3.3. Nous imposons en plus à cette spécification que les débuts des médias O_1 et O_2 soient synchronisés selon le schéma Or. En clair, les transitions $B(O_1)$ et $B(O_2)$ doivent être tirées en parallèle et simultanément dès que les contraintes temporelles de tir de l'une des deux transitions le permet. Donc le franchissement du rendez-vous $(Or, \{B(O_1), B(O_2)\})$ va octroyer la ressource à l'événement le plus prioritaire à savoir $B(O_2)$, générant les événements synchronisés $\{B(O_1)^*, B(O_2)\}$. En d'autres termes, le média O_1 s'éxécute initialement sans acquisition de la ressource rs retenue par la présentation de O_2.

3.5 Conclusion

Nous avons exploré durant le présent chapitre, comment capturer les sémantiques complexes inférées par les présentations des documents multimédia. Un intérêt particulier est réservé à la modélisation des schémas d'allocation et des gestions des conflits relatifs aux ressources non critiques et de leur impact sur les interactions utilisateurs. Pour ce faire, nous avons exploité un certain nombre de mécanismes existant tels, l'inihibition temporelle et la synchronisation, et avons défini des nouveaux pour les besoins de la modélisation. En finalité, les investigations entreprises ont permis de fournir un cadre général permettant la spécification de la majorité des exigences portées par les applications multimédia complexes.

Le cadre formel de cette proposition est décliné dans le chapitre suivant par la donnée de la syntaxe et de la sémantique formelles du modèle.

28. Présence d'un *arc inhibiteur* activé et connecté à l'une des transitions du rendez-vous.
29. Présence d'une place ressource non marquée connectée à une transition du rendez-vous via un arc standard, où bien d'un *hyperarc de franchissement* désactivé.

Chapitre 4

Synchronizing transitions Preemptive Time Petri Net

4.1 Introduction

Nous introduisons dans cette section une nouvelle extension de réseau de Petri dédiée à la spécification des systèmes temps réel complexes. Cette proposition synthétise l'étude présentée dans le chapitre précédent, et tente de proposer un modèle général permettant de traiter une large frange d'applications englobant entre autres les applications multimédia. Nous présentons d'abord la syntaxe du modèle, suivie de sa sémantique formelle.

4.2 Syntaxe formelle d'un $STPTPN$ [14][16][20]

Définition 7 *Un $STPTPN$ (*Synchronizing Transitions Pre-emptive Time Petri Net*) est défini par le tuple $(P_g, T_g, Res_g, U, FI, , B, F, M_0, RAS, IH, PH, Pr, \trianglerighteq, FH, RDV_s, RDV_m)$, où :*
- P_g et T_g sont deux ensembles non vides de places et de transitions globales respectivements ;
- Res_g est un ensemble de places ressource globales tel que : $Res_g \cap P_g = \varnothing$.
- U est un ensemble non vide d'unités tel que chaque unité notée $u_i (i = 1..p)$ est définie par le tuple (P_i, T_i, pc_i, rs_i) où :

1. *P_i et T_i sont deux ensembles disjoints de places et de transitions locales respectivement, propres à l'unité u_i tels que : $(P_i \cap P_g = \varnothing) \wedge (T_i \cap T_g = \varnothing)$, $\forall j \neq i$, $(P_i \cap P_j = \varnothing) \wedge (T_i \cap T_j = \varnothing)$. Par la suite, nous notons par P respectivement, T les ensembles $P_g \cup \left(\underset{j=1..p}{\cup} P_j \right)$ respectivement, $T_g \cup \left(\underset{j=1..p}{\cup} T_j \right)$ dénotant l'ensemble de toutes les places respectivement, transitions définies dans le modèle ;*
2. *pc_i est une place locale appartenant à P_i, appelée place centrale de l'unité u_i que l'on note $pc(u_i)$.*
3. *rs_i est une place ressource locale associée à l'unité u_i notée $resource(u_i)$, telle que, $\{rs_i\} \cap Res_g = \varnothing$ et $\forall j \neq i$, $resource(u_i) \neq resource(u_j)$. Nous notons par Res l'ensemble de toutes les places ressources définies dans le modèle, $Res = \{rs_1, ..rs_p\} \cup Res_g$.*

- FI est la fonction délai qui associe pour chaque transition t de T un intervalle défini sur $Q^+ \times (Q^+ \cup \{\infty\})$. Ainsi, FI donne l'intervalle de temps durant lequel la transition t peut être tirée : $FI(t) = [EFT(t), LFT(t)]$ avec $EFT(t) \preceq LFT(t)$. L'intervalle est délimité par un délai de tir au plus tôt $EFT(t)$, et par un délai de tir au plus tard $LFT(t)$, prenant leurs valeurs dans l'ensemble des nombres rationnels ;

- B et F sont deux fonctions déterminent les arcs standards, appelées fonctions d'incidence avant et arrière, telles que : $B : (P \cup Res) \times T \longrightarrow N$ et $F : (P \cup Res) \times T \longrightarrow N$. Cependant, pour la cohérence du modèle nous assumons que :

1. $\forall t \in T, \quad \sum_{\forall rs \in Res} F(rs,t) \preceq 1, \quad \sum_{\forall rs \in Res} B(rs,t) \preceq 1$; *une transition est connectée au plus à une place ressource via un arc standard 1-valué.*
2. $\forall t \in T, \exists p \in P, \quad B(p,t) \neq 0$; *chaque transition est reliée en entrée à au moins une place standard.*
3. *Si* $\exists t \in T, \exists rs \in Res, B(rs,t) = 1$, *alors* $LFT(t) = \infty$; *une transition reliée en entrée à une place ressource, via un arc standard doit avoir un délai de tir au plus tard non borné.*

- M_0 est une fonction appelée marquage initial qui associe pour chaque place du réseau un nombre de jetons[1]*, $M_0 : (P \cup Res) \longrightarrow N$. Nous assumons que toutes les places ressource sont au plus 1-marquées, $\forall r \in Res, \quad M_0(r) \preceq 1$;*

- RAS est un ensemble de "schémas d'allocation de ressources", tel que chaque élément l de RAS est donné par $(rs_g, \{rs_1, .., rs_q\})$ où rs_g est une place ressource globale de Res_g et $\{rs_1, .., rs_q\}$ est un ensemble de places ressources locales où $\forall i := 1..q, rs_i = resource(u_i)$. Nous notons par la suite par $units(l)$ les unités qui ont leur places ressources locales impliquées dans le schéma l. Nous assumons que deux schémas d'allocation de ressources impliquent deux ensembles disjoints de places ressources. $\forall (rs_g, \{rs_1, .., rs_q\}), (rs'_g, \{rs'_1, .., rs'_{q'}\}) \in RAS, \ \{rs_g, rs_1, .., rs_q\} \cap \{rs'_g, rs'_1, .., rs'_{q'}\} = \varnothing$. De plus, seule une place ressource parmi l'ensemble $(rs_g, \{rs_1, .., rs_q\})$ est marquée initialement, $M_0(rs_g) + \sum_{i=1..q} M_0(rs_i) = 1$;

- IH est une fonction d'inhibition définie par $P \times T \longrightarrow N$, telle que il y a un arc inhibiteur reliant la place p à la transition t, si $IH(p,t) \neq 0$;

- PH est un ensemble fini d'hyperarcs préempteurs : Chaque hyperarc est un tuple (srp, Vrp, t) où t est une transition propre à l'unité u_i, $\exists i \in \{1..p\}$, $t_i \in T_i$, srp est une place ressource d'entrée définie sur Res appelée "place ressource $forte$", et $Vrp = \{rs_1, ..rs_q\}$ est un ensemble pouvant être vide de places ressource défini sur Res appelées "place ressource de violation". Par ailleurs, nous assumons que :

1. $\exists (rs'_g, \{rs'_1, .., rs'_{q'}\}) \in RAS, \quad Vrp \cup \{srp\} \subseteq \{rs'_g, rs'_1, .., rs'_{q'}\}$; *l'ensemble des places ressource connectées à l'hyperarc préempteur appartient à l'un des schémas d'allocation de ressource définis.*
2. $Vrp \cap \{srp\} = \varnothing$; *la place forte ne peut être aussi une place de violation pour un même hyperarc préempteur ;*
3. *Soit $h := (srp, Vrp, t)$ un hyperarc préempteur. Si $t \in T_i$ alors pour chaque $rs \in Vrp$, $rs \neq resource(u_i)$; la place ressource associée à l'unité u_i auquelle appartient la transition t ne peut être une place de violation de h.*

1. L'ensemble N represente l'ensemble des entiers non négatifs.

4. $\forall t \in T_i, \quad |\{h = (srp, Vrp, t) \mid h \in PH\}| \preceq 1$ *; le nombre d'hyperarcs préempteurs reliés à chaque transition est au plus égal à un ;*

- Pr : est la fonction priorité affectant pour chaque unité une priorité, $Pr : U \longrightarrow N$*, et par extension aux transitions qui appariennent aux unités*[2]*.*

- $\trianglerighteq$ *: est une relation binaire non refelexive et non symétrique définie sur* $\bigcup_{i \neq j} \{T_i \times T_j \mid Pr(u_i) \succ Pr(u_j)\}$ *; si* $(t_i, t_j) \in \trianglerighteq$*, nous disons alors que* t_j *est privée par* t_i *de consommer les jetons de ses places ressource. Nou assumons ainsi qu'un arc de privation relie une transition propre à une unité, à une autre transition appartenant à une unité disctincte de faible priorité.*

- FH est un ensemble fini d'hyperarcs de franchissement. Chaque hyperarc de franchissement est un tuple (rs_i, rs_g, CP, t_i) *où* $t_i \in T_i$ *est une transition appartenant à l'unité* u_i *telle que son délai de tir au plus tard est non borné* $LFT(t_i) = \infty$*;* $rs_i = resource(u_i)$ *est la place ressource associée à* u_i *auquel* t_i *appartient ;* $rs_g \in Res_g$ *est une place ressource globale.* rs_i *respectivement,* rs_g *est appelée place de* $franchissement\ forte$ *respectivement, place de* $franchissement\ faible$*. Enfin,* $CP = \{p_1..p_q\}$ *est un ensemble de places centrales d'unités disctinctes appelées "places de conditionnement de franchissement" . Dans un but de garantir une certaine cohérence du modèle, nous assumons que :*

1. $\exists l = (rs'_g, \{rs'_1, .., rs'_{q'}\}) \in RAS, \quad \forall p_i \in CP, \quad \exists u_i, p_i = cp(u_i) \wedge (resource(u_i) \in units(l))$ *; l'ensemble des places de conditionnement de franchissement appartiennent à des unités connectées à un même schéma d'allocation.*
2. $\forall t_i \in T_i, |\{y = (rs_i, rs_g, CP, t_i) \mid y \in FH\}| + |\{h = (srp, Vrp, t_i) \mid h \in PH\}| + \sum_{\forall rs \in \mathrm{Re}\, s} B(rs, t_i) \preceq 1$; *une transition peut être reliée en entrée à un ensemble de places ressource soit par un seul hyperarc préempteur, un seul hyperarc de franchissement ou un seul arc standard ;*
3. $\forall t_i \in T_i, |\{y = (rs_i, rs_g, CP, t_i) \mid y \in FH\}| + |\{(t, t_i) \in \trianglerighteq\}| \preceq 1$; *une transition* t_i *reliée à un hyperarc de franchissement ne peut être reliée à un arc de privation ;*
4. $\forall t_i \in T_i, |\{y = (rs_i, rs_g, CP, t_i) \mid y \in FH\}| + |\{p \in P \mid IH(p, t_i) \neq 0\}| \preceq 1$; *une transition* t_i *reliée à un hyperarc de franchissement ne peut être reliée à un arc inhibiteur ;*
5. $\exists l = (rs'_g, \{rs'_1, .., rs'_{q'}\}) \in RAS, \quad \forall u_i = (P_i, T_i, rs_i) \in U, \forall t_i \in T_i,$
   $Si\ \exists (rs_i, rs_g, CP, t_i) \in FH,\ alors\ \{rs_i, rs_g\} \subseteq \{rs'_g, rs'_1, .., rs'_{q'}\}$
   $Si\ \exists (srp, Vrp, t_i) \in PH,\ alors\ \{srp\} \cup Vrp \subseteq \{rs'_g, rs'_1, .., rs'_{q'}\}$
   $Si\ \exists rs \in Res, \quad B(rs, t_i) \neq 0 \vee F(rs, t_i) \neq 0,\ alors\ rs \in \{rs'_g, rs'_1, .., rs'_{q'}\}$
   Les places ressource connectées à une unité appartiennent à un même schéma d'allocation de ressource.

- RDV_s *dénote l'ensemble fini de* $rendez-vous$ *standards. Un* $rendez-vous\ r$ *de* RDV_s *est donné par* $(typ, \{t_1, ..t_q\})$*, où les transitions* $t_1..t_q$ *sont différentes et prises respectivement dans* $T_1 \cup T_g, .., T_q \cup T_g$*.* typ *définit la règle de synchronisation, et prend une des valeurs suivantes :* $typ \in \{And,\ Weakand,\ Or,\ Strong - Or\}$ *;*

- RDV_m *est un ensemble fini de* $rendez-vous\ maître$ *. Un* $rendez-vous\ maître\ r$ *est donné par* $(typ,\ \{t_m\} \Rightarrow \{t_1, .., t_q\})$ *tel que* $\{t_m\} \cap \{t_1, .., t_q\} = \emptyset$*, et où* $t_1, ..., t_q$*, et* t_m *sont des transitions appartenant respectivement aux ensembles* $T_1 \cup T_g, .., T_q \cup T_g$*, et* $T_m \cup T_g$*, où* $m \notin \{1, .., q\}$*.*

2. Les transitions d'une même unité ont la même priorité.

Par ailleurs, typ définit la règle de synchronisation prenant une des valeurs suivantes : typ $\in \{Master,\ Or-Master,\ And-Master,\ Weak-Master,\ Strong-Master\}$. *La transition* t_m *est notée par* $MA(r_m)$ *et appelée la transition maître. Nous assumons de plus que la transition* $t_m \notin \{t_1, .., t_q\}$

Remarque : Une *place ressource* est reliée à une transition en entrée en utilisant soit un *hyperarc préempteur*, soit un *hyperarc de franchissement*, soit un *arc standard* (donné respectivent par PH ou FH ou B) et en sortie en utilisant un *arc standard* (donné par F).

4.3 Sémantique formelle d'un $STPTPN$

Un $STPTPN$ progresse à chaque pas par le franchissement d'un rendez-vous ; dénotant l'occurrence d'une synchronisation d'événements. Par conséquent, l'occurrence d'un rendez-vous est conditionnée par sa règle de synchronisation. Cette dernière implique le tir (de manière *indivisible*[3]) de toutes ses transitions, prenant en compte leurs contraintes de temps et la disponibilité des ressources nécessaires pour déterminer quels événements doivent être générés suite à la réalisation du rendez-vous.

	r_1		r_2		r_n	Rendez-vous franchis
0	<——>	1	<——> 2	. . . .	n-1 <——> n	points de franchissement
M_0		M_1	M_2		M_{n-1} M_n	marquages accessibles
e_0		e_1	e_2		e_{n-1} e_n	états accessibles

Soit $ST := (P_g, T_g, Res_g, U, FI, , B, F, M_0, RAS, IH, PH, Pr, \trianglerighteq, FH, RDV_s, RDV_m)$, un $STPTPN$.

On suppose qu'une séquence de rendez-vous $(r_1, .., r_n)$ a été franchie dans ST depuis l'état initial. Par convention, le rendez-vous r_j est franchi au $(j-1)^{ieme}$ point. Le marquage et l'état accessibles au $(j)^{ieme}$ point sont notés M_j et e_j, respectivement.

Définition 8 *Soit* M_n *un marquage accessible au point* (n) *:*

- Une transition t *est dite sensibilisée pour le marquage* M_n*, si et seulement si le nombre de jetons dans chaque place standard en entrée de* t *est supérieur ou égal à la valuation de l'arc entre cette place et* t *:* $\forall p \in P, \quad B(p,t) \preceq M_n(p)$. *On dénote alors par* Te_n *l'ensemble des transitions sensibilisées pour* M_n*.*

- Une transition est dite inhibée par le marquage M_n *si elle est sensibilisée et un de ses arcs inhibiteurs est activé :* $t \in Te_n \ \wedge \ \exists p \in P,\ (IH(p,t) \neq 0) \wedge (IH(p,t) \preceq M_n(p))$. *On dénote par* Ti_n *l'ensemble des transitions inhibées pour* M_n*.*

- Une transition t *est dite interdite de tir pour le marquage* M_n*, si et seulement si* t *est sensibilisée, non inhibée et que soit un hyperarc de franchissement est connecté à* t *tel que :*

$$\left(\begin{array}{l} (t \in Te_n) \wedge (t \notin Ti_n) \wedge \quad \forall (rs_i, rs_g, \{p_1..p_q\}, t) \in FH, \\ (M_n(rs_i) = 0) \wedge \left(\begin{array}{l} (M_n(rs_g) = 1) \wedge (\exists_{j \in \{1..q\}}, M_n(p_j) = 1) \quad \vee \\ (M_n(rs_g) = 0 \end{array} \right. \end{array} \right.$$

3. Toutes les transitions sont tirées ensemble et simultanément.

ou bien qu'il y ait une place ressource non marquée connectée à t via un arc standard : $\exists rs \in Res, (B(rs,t)=1) \wedge M(rs)=0$.

On dénote alors par Tf_n *l'ensemble des transitions interdites de tir pour* M_n.

- Une transition t est dite privée de ressources pour le marquage M_n*, si et seulement si elle est sensibilisée, non inhibée, et non interdite de tir, et qu'il existe une autre transition sensibilisée, non inhibée et non interdite de tir* t' *telle qu'il y a un arc de privation partant de* t' *vers t :* $(t \in Te_n) \wedge (t \notin Tf_n) \wedge (t \notin Ti_n) \wedge (\exists t' \in Te_n, (t' \notin Ti_n) \ \wedge (t' \notin Tf_n) \ \wedge \ (t',t) \in \trianglerighteq)$. *On dénote par* Td_n *l'ensemble des transitions qui sont privées de ressource pour* M_n.

- Une transition t est dite fortement sensibilisée pour le marquage M_n *si elle est sensibilisée, non inhibée, non interdite de tir, non privée de ressources et que s'il existait un hyperarc préempteur connecté à la transition t alors sa place ressource forte devrait être marquée :* $(t \in Te_n) \ \wedge (t \notin Tf_n) \wedge \ (t \notin Ti_n) \wedge \ (t \notin Td_n) \ \wedge \ (\forall \ (srp, Vrp, t) \in PH, \ M_n(srp)=1)$. *On note par* Ts_n *l'ensemble des transitions fortement sensibilisées pour* M_n.

- Une transition t est dite violée pour le marquage M_n*, si elle est sensibilisée, non inhibée, non privée de ressources et qu'il existe un hyperarc préempteur connecté à t tel que toutes ses places ressource connectées sont non marquées :* $(t \in Te_n) \ \wedge \ (t \notin Ti_n) \ \wedge (t \notin Tf_n) \wedge (\ t \notin Td_n) \wedge \ (\exists (srp, Vrp, t) \in PH, \ \ \forall rs \in \{srp\} \cup Vrp, \quad M_n(rs)=0)$. *On dénote par* Tv_n *l'ensemble des transitions violées pour* M_n.

- Une transition t est dite violeuse pour le marquage M_n *si, elle est sensibilisée, non inhibée, non privée de ressources, et qu'il existe un hyperarc préempteur connecté à t tel que sa place ressource forte est non marquée et au moins une de ses places ressource de violation est marquée :* $(t \in Te_n) \ \wedge \ (t \notin Ti_n) \ \wedge (t \notin Td_n) \ \wedge (\exists (srp, Vrp, t) \in PH, \quad (M_n(srp)=0) \wedge (\exists rs \in Vrp, \quad M_n(rs)=1)$. *On dénote alors par* Tp_n *l'ensemble des transitions violeuses pour* M_n.

Remarque : Une transition *interdite de tir* et une transition *inihibée* sont toutes deux non franchissables, seulement dans le premier cas l'horloge associée à la transition continue de progresser alos que dans le second cas elle est suspendue. Par ailleurs, on suppose ici que le modèle est un réseau $T-Sauf$: pour n'importe quel marquage, une transition t est au plus $1-$*sensibilisée* (Un marquage ne peut sensibiliser plusieurs fois la même transition).

Notation 1 *Nous notons par* RDV *l'ensemble des rendez − vous pouvant être franchis dans le modèle. Cet ensemble contient les rendez-vous de* RDV_s *et* RDV_m*, et aussi l'ensemble des rendez-vous asynchrones noté* RDV_a*. Un rendez-vous asynchrone* r_a *contient une seule transition t de T, telle que t n'est impliquée dans aucun rendez-vous de* $RDV_s \cup RDV_m$*. De plus, si r est un rendez-vous de RDV, on dénote par* $Trans(r)$ *l'ensemble des transitions se synchronisant dans r.*

Définition 9 *Soit* M_n *un marquage accessible au point* (n) *:*

- Un rendez-vous r est dit sensibilisé pour le marquage M_n*, si toutes ses transitions sont sensibilisées et non en conflit pour les places standards dans* M_n*, et on dénote par* $Enable_n$ *l'ensemble de tous les rendez-vous de RDV sensibilisés pour* M_n.

$$Enable_n = \left\{ \begin{array}{l} r \in RDV \mid \ Trans(r) \subseteq Te_n \quad \wedge \quad \forall t,t' \in Trans(r), \\ \qquad\qquad\qquad \forall p \in P, \quad B(p,t)+B(p,t') \preceq M_n(p) \end{array} \right\}$$

- Un rendez-vous r et dit inhibé pour le marquage M_n, si il est sensibilisé pour M_n, et au moins une de ses transitions est inhibée pour le marquage M_n. On dénote par $Inhibit_n$ l'ensemble des rendez-vous de RDV inhibés pour M_n.
$Inihibit_n = \{r \in RDV \mid (r \in Enable_n) \wedge (\exists t \in Trans(r),\ t \in Ti_n)\}$.

- Un rendez-vous r et dit interdit de franchissement pour le marquage M_n, si il est sensibilisé non inhibé pour M_n, et au moins une de ses transitions est interdite de tir pour le marquage M_n. On dénote par $Interdit_n$ l'ensemble des rendez-vous de RDV interdits de franchissement pour M_n. $Interdit_n = \{r \in RDV \mid (r \in Enable_n) \wedge (r \notin Inhibit_n) \wedge (\exists t \in Trans(r),\ t \in Tf_n)\}$.

- Un rendez-vous r est dit fortement sensibilisé pour le marquage M_n, si il est sensibilisé, non inhibé et non interdit de franchissement pour M_n. On note par $Senable_n$ l'ensemble des rendez-vous qui sont fortement sensibilisés pour M_n.
$Senable_n = \{r \in RDV \mid (r \in Enable_n) \wedge (r \notin Interdit_n) \wedge (r \notin Inhibit_n)\}$

Définition 10 (Etat d'un $STPTPN$) *Un état d'un $STPTPN$ accessible après le franchissement de la séquence de rendez-vous $S_n = (r_1, ..r_n)$ est le couple $e_n = (M_n, V_n)$, où M_n est le marquage courant, et V_n est une fonction associant pour chaque transition sensibilisée, ses contraintes de tir : $V_n(t) = x_n(t) \preceq \underline{t} \preceq y_n(t)$ avec $\underline{t}$ est une horloge écoulant le temps résiduel de t.*
L'état initial du modèle est donné par la paire (M_0, V_0), où M_0 est le marquage initial et V_0 associe pour chaque transition sensibilisée son intervalle de tir statique :

$$\forall t \in Te_0 \qquad V_0(t) = EFT(t) \preceq \underline{t} \preceq LFT(t)$$

Définition 11 (Intervalle de validité temporel d'un rendez-vous) *. Soit e_n un état accessible et r un rendez-vous fortement sensibilisé pour e_n. On associe à r son intervalle de validité temporel donné par la contrainte $TVI(r) = Low_n(r) \preceq \underline{r} \preceq Up_n(r)$, où $Low_n(r)$ et $Up_n(r)$ représentent respectivement, la borne inférieure et la borne supérieure de r, et $\underline{r}$ étant l'horloge égrénant le temps résiduel associé au rendez-vous r. L'intervalle $[Low_n(r), Up_n(r)]$ détermine l'intervalle relatif des dates potentielles pour lesquelles r peut être franchi à partir de e_n. Ainsi, r peut se réaliser à partir de e_n à l'instant relatif $\underline{r}$ si $0 \preceq Low_n(r) \preceq \underline{r} \preceq Up_n(r)$, où $Low_n(r)$ et $Up_n(r)$ sont calculées en fonction de la règle de synchronisation fournie par $typ(r)$:*

$Low_n(r) :=$

$$\begin{cases} Si\ typ(r) \in \{And,\ Wand,\ Amas,\ Async\}\ alors \\ \quad \underset{\forall t \in Trans(r)}{MAX} \{x_n(t)\} \\ Si\ \ typ(r) \in \{Or,\ Sor,\ Omas\}\ alors \\ \quad \underset{\forall t \in Trans(r)}{MIN} \{x_n(t)\} \\ Si\ typ(r) \in \{Mas,\ Smas,\ Wmas\} \wedge (t_m = MA(r))\ alors \\ \quad x_n(t_m) \end{cases}$$

$Up_n(r) :=$

$$\begin{cases} Si\ typ(r) \in \{And,\ Sor,\ Smas, Async\}\ alors \\ \quad \underset{\forall t \in Trans(r)}{MIN} \{\ y_n(t)\} \\ Si\ typ(r) \in \{Wand,\ Or,\ Wmas\}\ alors \\ \quad \underset{\forall t \in Trans(r)}{MAX} \{\ y_n(t)\} \\ Si\ typ(r) \in \{Mas,\ Omas,\ Amas\}\ \wedge\ (t_m = MA(r))\ alors \\ \quad y_n(t_m) \end{cases}$$

Il est à remarquer que les formules précédentes peuvent calculer trois types d'intervalles selon l'agencement des contraintes de temps des transitions se synchronisant pour le rendez-vous :

- L'intervalle de validité du rendez-vous est *cohérent*, si $0 \preceq Low_n(r) \preceq Up_n(r)$ offrant ainsi l'intervalle des dates pour lesquelles r **pourrait** être franchi.
- L'intervalle calculé pourrait être *vide* (incohérent) si, $0 \preceq Up_n(r) \preceq Low_n(r)$ dénotant l'état du système où le rendez-vous r ne peut être franchi car le schéma de synchronisation imposé ne peut être satisfait, étant donné les contraintes de temps de ses transitions constatées à l'état e_n.
- Par ailleurs, l'intervalle pourrait être *temporellement inconsistant* si $Up_n(r) \prec Low_n(r) = 0$, dénotant le cas où le rendez-vous a son intervalle surpassé par la progression du temps et ce avant sa sensibilisation. Plus précisément, cette situation modélise un état de blocage du à une violation temporelle. Par conséquent, les états modélisant ces inconsistances sont appelés états *temporellement inconsistants*, et sont formellement identifiés dans l'espace des états accessibles en utilisant la définition suivante.

Définition 12 (état temporellement inconsistant) *Un état $e_n = (M_n, V_n)$ est dit temporellement inconsistant, si la borne supérieure de l'intervalle de validité d'un rendez-vous fortement sensibilisé a été surpassée :* $\exists r \in Senable_n \qquad Up_n(r) \prec 0$.

En essayant de distinguer les principales différences entre un $RdPT$ et un $STPTPN$, nous remarquons que dans un $RdPT$ la notion du temps adoptée est *forte* ; les contraintes de temps des transitions sensibilisées ne peuvent être violées car le temps progresse dans le respect de ces contraintes. Au contraire d'un $STPTPN$ où nous observons deux types de sémantiques :

- Une sémantique *faible* qui se réfère aux contraintes liées aux transitions, i.e., le passage du temps peut surpasser la borne supérieure d'une transition sensibilisée ;
- Une sémantique *forte* associée au franchissement des rendez-vous, i.e., un rendez-vous r ne peut se réaliser que dans le respect de ses contraintes temporelles. Par conséquent, une transition n'est jamais forcée de tirer durant son propre intervalle. Au fait, elle est forcée de tirer dans l'intervalle du rendez-vous auquel elle est assujettie et dont le calcul dépend de la règle de synchronisation de ce dernier et des intervalles associés aux transitions à synchroniser.

De là, la violation des contraintes d'une transition sensibilisée (i.e, $y_n(t) \prec 0$) peut se produire si la règle associée à son rendez-vous l'autorise. De plus, comme les contraintes d'un rendez-vous sont calculées à partir de celles de ses transition synchronisantes, l' *intervalle de validité* d'un rendez-vous pourrait être violé lors de sa sensibilisation, produisant des états de blocages appelés *états temporellement inconsistants*. Donc, un état *temporellement inconsistant* indique un état de blocage à partir duquel aucun rendez-vous ne peut être réalisé.

Par exemple, dans le $STPTPN$ de la $Figure$ 4.1, les transitions t_2 et t_3 doivent tirer de manière synchrone alors que t_1 évolue de manière asynchrone. Initialement, seul le rendez-vous $r_1 = \{t_1\}$ est fortement sensibilisé et peut se réaliser durant $[3, 4]$. Donc, tous les états accessibles de là sont temporellement inconsistants car la borne supérieure du rendez-vous $r_2 = (And, \{t_3, t_2\})$ est surpassée par la progression du temps. $TVI(r_2) = [MAX(1-\delta, 0),\ MIN(1-\delta, 3] = [0, 1-\delta]$ tel que $\delta \in [3, 4]$ ce qui implique : $1 - \delta \prec 0$.

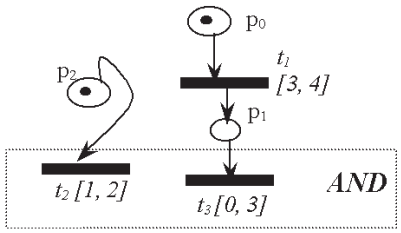

FIGURE 4.1 – Un $STPTPN$ génerant des états temporellement inconsistants.

Les contraintes temporelles associées aux transitions sensibilisées et aux rendez-vous fortement sensibilisés au point $(n-1)$, peuvent être représentées par un système linéaire où les variables sont les horloges associées à ces derniers :

$$F_{n-1} = \begin{cases} FT_{n-1} = \underset{\forall t \in Te_{n-1}}{\wedge} V_{n-1}(t) \\ FR_{n-1} \underset{\forall r \in Senable_{n-1}}{\wedge} TVI_{n-1}(r) \end{cases}$$

Ainsi, le système F_{n-1} inclut deux sous-systèmes FT_{n-1} et FR_{n-1}. Le premier FT_{n-1} modélise les contraintes des transitions, et FR_{n-1} celles des rendez-vous. Cependant, bien que ce dernier est redondant (puisqu'il est calculé à partir du premier), son calcul est nécessaire pour la réalisation du test de franchissement. D'ailleurs, la définition suivante exploite cette notation et propose une condition nécessaire et suffisante pour le franchissement d'un rendez-vous à partir d'un état *temporellement consistant*[4].

Définition 13 (franchissement d'un rendez-vous) *Soit $e_{n-1} = (M_{n-1}, V_{n-1})$ un état temporellement consistant accessible après le franchissement de la séquence S_{n-1}. Un rendez-vous r_n est dit franchissable à partir de e_{n-1} à la date relative $\underline{r_n}$, si : (i) r_n est fortement sensibilisé pour le marquage M_{n-1} ; (ii) La progression de temps permet de satisfaire les contraintes du système FR_{n-1} sans violer les contraintes d'aucun autre rendez-vous fortement sensibilisé (iii) et qu'aucun rendez-vous fortement sensibilisé incluant strictement r_n n'est franchissable pour la même date :*

(i) $r_n \in Senable_{n-1}$

(ii) *Le système* $FR_{n-1} \wedge (\forall r \in Senable_{n-1},\ \ \underline{r_n} \preceq \underline{r})$ *est consistant.*

(iii) *Le système* $FR_{n-1} \wedge\ (\forall r' \in \Phi_{n-1}(r_n)\ ,\ \ \underline{r_n} \prec Low_{n-1}(r'))$ *est consistant.*

avec : $\Phi_{n-1}(r_n) = \{r' \quad | \ r' \in Senable_{n-1} \wedge Trans(r_n) \subset Trans(r')\}$

Remarque : Les clauses (ii) et (iii) reviennent à satisfaire la condition suivante :

$$\left(Low_{n-1}(r_n) \prec \underset{\forall r' \in \Phi_{n-1}(r_n)}{MIN} \{Low_{n-1}(r')\}\right) \wedge \left(Low_{n-1}(r_n) \preceq \underset{\forall r \in Senable_{n-1}}{MIN} \{Up_{n-1}(r)\}\right)$$

4. Un état e_n est dit *temporellement consistant* s'il ne satisfait pas la $Définition$.12.

L' hypothèse (*iii*) applique la sémantique de priorité en temps réel introduite dans [10][11] pour traiter le non déterminisme lorsque plusieurs rendez-vous sont offerts pour les même instants. Pour cet effet, ce mécanisme opère lorsque des rendez-vous sont strictement inclusifs. Clairement, le modèle peut franchir un rendez-vous r_n à l'instant $\underline{r_n}$ seulement si il n'existe pas un autre rendez-vous fortement sensibilisé r' incluant strictement r_n et franchissable pour la même date. Donc, à un instant donné la priorité est donnée au rendez-vous impliquant le plus grand nombre d'événements. Cependant, si les deux rendez-vous sont non strictement inclusifs alors le modèle franchit l'un des deux de manière non déterministe. Par exemple, si on considère l'exemple donné dans la *Figure* 4.2, initialement t_1, t_2, t_3 et t_4 sont sensibilisées et peuvent être tirées selon trois schémas : $r_1 = (Or, \{t_1, t_2\})$, $r_2 = (AND, \{t_1, t_2, t_3\})$ et $r_3 = (Wand, \{t_3, t_4\})$. Nous obtenons les TVI suivants : $TVI_0(r_1) = [0, 5]$, $TVI_0(r_2) = [2, 2]$ and $TVI_0(r_3) = [2, 4]$. De la, r_1 peut se réaliser seulement lorsque r_2 est non franchissable, C.à.d durant $[0, 2[$. Par contre, à l'instant 2 et plus, la priorité est donnée à r_2, puisqu'il inclut strictement r_1. Par ailleurs, les rendez-vous non inclusifs r_2 et r_3 peuvent se réaliser de manière non déterministe à l'instant 2.

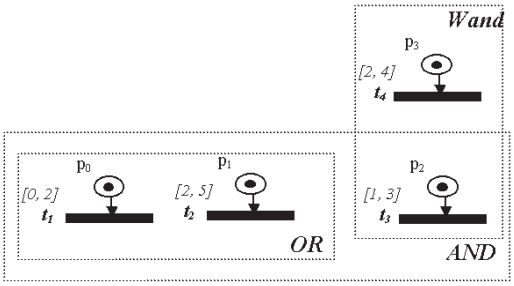

FIGURE 4.2 – Rendez-vous conflictuels et inclusifs.

De plus, rappellons que le franchissement d'un rendez-vous r_n implique le tir simultané de toutes ses transitions (i.e., $t \in Trans(r_n)$) de manière indivisible. Ceci produit un nouvel état accessible e_n où le marquage correspondant M_n est obtenu à partir de M_{n-1} après le tir des transitions de $Trans(r_n)$.

Comme le marquage des places ressource dénote la disponibilité de ces dernières pour l'occurrence d'un événement, les transitions impliquées dans le rendez-vous à franchir pourraient être en conflit pour le marquage courrant pour les places ressource. Donc, le franchissement du rendez-vous implique la résolution du conflit en déterminant quelle transition consommerait le jeton présent dans la place ressource sujette au conflit. Pour ce faire, le problème est résolu lors du calcul du marquage accessible. Ce dernier obtenu en tirant les transitions *synchronisantes* séquentiellement partant de la plus à la moins prioritaire [5], selon l'ordre Pr. Chaque sous-marquage obtenu dans l'ordre établi permet de déterminer le statut de la prochaine transition à tirer (*fortement sensibilisée, violeuse, ou violée*) et de là, l'événement à générer. Ce passage est formalisé par la relation de transition entre états accessibles, définie ci-après.

Définition 14 *Soit $e_{n-1} = (M_{n-1}, V_{n-1})$ un état temporellement consistant accessible après le franchissement de la séquence S_{n-1}. On dénote par $e_{n-1} \xrightarrow{L(r_n)} e_n$ la réalisation du rendez-vous r_n à partir de l'état e_{n-1} à la date relative $\underline{r_n}$ telle que $e_n = (M_n, V_n)$ est l'état accessible, et $L(r_n)$*

5. Dans le cas où des transitions ont la même priorité ou qu'elles aient des priorités indéfinies. Le choix de la transition à tirer en premier se fait de manière non déterministe.

est le label des événements générés :

- M_n est calculé à partir de M_{n-1} en tirant les transitions de $Trans(r_n)$ selon l'ordre de priorité donné par la fonction Pr ; commençant par la transition la plus prioritaire jusqu'à celle ayant la plus basse priorité.

Soit $Trans(r) = \{t_1, ..t_q\}$ où, $Pr(t_1) \succeq Pr(t_2)$ ... $\succeq Pr(t_q)$. On note par : $M^0 \overset{event(t_1)}{\longrightarrow} M^1 ... M^{q-1} \overset{event(t_q)}{\longrightarrow} M^q$ la séquence des marquages accessibles après le tir séquentiel simultané et indivisible des transitions $t_1, .., t_q$, où $M_{n-1} = M^0$ et $M_n = M^q$. Donc, $event(t_i)$ représente l'événement induit lors du tir de la transition t_i, et peut être : un événement standard noté t_i si la transition est $fortement\ sensibilisée$ pour M^{i-1} ; un événement de violation passive noté t_i^* si la transition est violée ou privée de ressources pour M^{i-1} ; Un événement de violation active noté $^*t_i$ si la transition est violeuse pour M^{i-1} :

$$\begin{cases} event(t_i) := "t_i" \quad si\ t_i \in Ts^{i-1} \\ event(t_i) := "t_i^*" \quad si\ t_i \in Tv^{i-1} \cup Td^{i-1} \\ event(t_i) := "^*t_i" \quad si\ t_i \in Tp^{i-1} \end{cases}$$

Chaque marquage M^i est calculé à partir de M^{i-1} de manière usuelle pour les places standards. Pour l'évolution du marquage relatif aux places ressource[6]*, il s'agira de considérer l'état de la transition tirée t_i. Plus concrètement, si la transition t_i est connectée à un hyperarc de franchissement*[7]*, son tir provoquera la concommation du jeton présent dans toute place ressource reliée à l'hyperarc de franchissement et la production de jeton dans toute place ressource en sortie. Par ailleurs, si la transition est connectée à la place ressource via un hyperarc préempteur tel qu'il n'existe pas d'arc de privation connecté à t_i, alors son statut (fortement sensibilisée, violée, violeuse), permettra de déterminer l'évolution du marquage. Autrement, si la transition est "privée de ressources"*[8]*, alors le marquage de toute place ressource reliée à l'hyperarc préempteur reste inchangé après le tir de t_i. Il est à noter que seules les transitions franchies avec un statut de violeuses ou fortement sensibilisées peuvent produire un jeton dans les places ressource en sortie. Plus formellement, nous écrivons :*

$\forall p \in P, \qquad M^i(p) := M^{i-1}(p) - B(p, t_i) + F(p, t_i)$

$\forall rs \in Res,$

Si $rs \in \{rs_1, rs_2\}$ *tel que* $\exists (rs_1, rs_2, CP, t_i) \in FH,$

Alors $M^i(rs) := F(rs, t_i)$

Sinon Si $\exists (srp, Vrp, t_i) \in PH,\ rs \in \{srp\} \cup Vrp$ *Alors*

$$\begin{cases} \text{Si } t_i \in Ts^{i-1}\ alors\ M^i(rs) := F(rs, t_i) \\ \text{Si } t_i \in Tv^{i-1}\ alors\ M^i(rs) := 0 \\ \text{Si } t_i \in Td^{i-1}\ alors\ M^i(rs) := M^{i-1}(rs) \\ \text{Si } t_i \in Tp^{i-1}\ alors\ M^i(rs) := F(rs, t_i) \end{cases}$$

Sinon si $(F(rs, t_i) = 1\ ) \vee (B(rs, t_i) = 1)$, *alors*

$$\begin{cases} \text{Si } t_i \in Ts^{i-1}\ alors\ M^i(rs) := F(rs, t_i) \\ \text{Si } t_i \in Td^{i-1}\ alors\ M^i(rs) := M^{i-1}(rs) \end{cases}$$

6. Nous assumons que le nombre de jetons dans chaque *place resource* est borné à un.

7. t_i est non *interdite de tir*. De plus, dans le respect de la syntaxe formelle du modèle, il ne peut y avoir un *hyperarc préempteur* ou un *arc de privation* connecté à t_i.

8. Il y a en plus un *arc de privation* relié à la transition t_i.

$$Sinon \quad M^i(rs) := M^{i-1}(rs)$$

- La fonction V_n est calculée en trois étapes :

a) éliminer les transitions qui ont été désensibilisées après le tir de r_n ; les transitions sont celles sensibilisées pour M_{n-1} et non pas pour M_n ou celles sensibilisées pour M_{n-1} et M_n et en conflit avec l'une des transitions franchies dans r_n.

b) pour chaque transition persistante t (t sensibilisée pour M_{n-1} et le reste pour M_n), deux cas sont observés :

* *Si t est non inhibée pour M_{n-1}, alors décaler $V_{n-1}(t)$ de la valeur $\underline{r_n}$ vers la nouvelle origine temporelle, et tronquer si nécessaire $V_{n-1}(t)$ au niveau de sa borne inférieure à une valeur non négative : $V_n(t) = x_n \preceq \underline{t} \preceq y_n$ avec, $[x_n, y_n] = [MAX(0, x_{n-1}(t) - \underline{r_n}),\ y_{n-1}(t) - \underline{r_n}]$*
* *Si t est inhibée pour le marquage M_{n-1}, alors ses contraintes de tir restent inchangées : $V_n(t) = V_{n-1}(t)$*

c) Introduire les transitions nouvellement sensibilisées pour M_n en leur assignant leur intervalles statiques : $V'(t) = EFT(t) \preceq \underline{t} \preceq LFT(t)$.

- Le label $L(r_n)$ est donné par l'ensemble des événements générés lors du tir de chaque transition se synchronisant dans r_n : $L(r_n) = \{event(t) \ \mid \ t \in Trans(r_n)\}$

Remarque : Notons que la borne supérieure de $V_{n-1}(t)$ est non tronquée à une valeur non négative ; permettant ainsi de détécter les états inconsistants. Par ailleurs, comme nous avons assumé qu'une transition *interdite de tir*[9] doit avoir un intervalle non borné, nous garantissons que ses contraintes de temps ne seront jamais violées suite à la progression du temps.

Jusqu'à lors nous nous sommes investi dans la définition d'un nouveau modèle permettant d'étendre les réseaux de Petri temporels aux sémantiques de préemption de ressource, de préemption de temps et enfin permettant de capturer les mécanismes de synchronisation. Nous explorons maintenant, comment exploiter ce modèle pour la spécification des présentations multimédia complexes en exploitant les mécanismes nouvellement introduits.

4.4 Application

Le langage $SMIL$ [90] est devenu ces dernières années la référence dominante pour la spécification et la caractérisation des présentations multimédia. D'ailleurs, il a suggéré différents travaux particulièrement ceux relatifs à l'étude de la consistance de ses documents [7][6][47]. Des algorithmes basés sur des méthodes formelles ont ainsi été exploités dans ses outils d'édition [64] et de lecture [27][81] dans le but vérifier la consistance de ses présentations. A partir de là, il devint possible de programmer des scénarios consistants et appropriés et de définir des protocoles de délivrance garantissant une qualité de service continue [52][88].

Nous montrons dans cette section comment modéliser les contraintes d'un document $SMIL$ par un $STPTPN$. Nous focalisons sur les contraintes ressources, mettant en exergue les inconsistances qui en résultent. Cet aspect particulier ignoré dans les travaux existants [47][84], fera l'objet d'une plus grande attention dans l'étude menée ci-après.

9. Soit par un arc standard ou un hyperarc de franchissement.

4.4.1 Le langage SMIL

SMIL (Synchronized Multimedia Integration Language) [90] est un langage déclaratif basé sur XML décrivant les aspects spatiaux et temporels des différentes présentations des objets. Un objet de base peut être une *image*, un *texte*, une *vidéo* ou un *clip audio*. Chaque object est déclaré gràce à des balises, et est principalement caractérisé par des paramètres tels que : son identification Id, son URL, sa durée, ses dates de début et de fin, et sa zone d'affichage appelée région...etc

D'autre part, des opérateurs de construction sont fournis permettant de composer différentes présentations, selon différents schémas (e.g. Composition parallèle gràce à la balise $< par >$ ou la composition séquentielle gràce à la balise $< seq >$). Le non-déterminisme est introduit aussi par le biais de balises "**liens**" : La balise "a" spécifiant un lien URL associé à une présentation, et la balise *Anchor* similaire à a mais qui permet en plus de spécifier les coordonnées de la zone d'affichage du lien, ainsi que l'intervalle de temps durant lequel le lien est offert.

Nous associons comme décrit dans le $Chapitre\ III$ à chaque élément d'une présentation $SMIL$ noté O deux événements, son événement de début $B(O)$ et son événement de fin $E(O)$. Notons que les paramètres temporelles caractérisant le début et la fin d'un object O peuvent être exprimés comme des *valeurs d'horloges* ou des *valeurs d'événements*. Une *valeur d'horloge* exprime une date relative dénotant l'heure à laquelle l'événement doit se réaliser alors qu'une *valeur d'événement* exprime une relation de synchronisation entre deux événements. Une valeur d'événement prend la forme $Id(C)\ (e)$, où e spécifie soit l'occurrence d'un événement ou une valeur d'horloge : e.g. $End :=$ "$id(C)\ (3)$" exprime que O termine "3" unités de temps après que C ait commencé, tandis que $End :=$ "$id(C)\ (end)$" exprime que O termine dès que C achève sa présentation. La relation de synchronisation précédente est qualifiée de relation *maître*, c.-à-d., l'occurrence du premier événement (dit *esclave*) est décidée par l'occurrence du second événement (dit *maître*). Cependant, l'événement *maître* peut avoir lieu même si l'événement *esclave* est non réalisable. Par conséquent, pour modéliser ce mécanisme de synchronisation dans un $STPTPN$, nous devons considérer deux rendez-vous, le premier utilise la règle "MAS" modélisant la synchronisation *maître* entre les deux événements, et le second modélise l'évolution asynchrone de l'événement *maître* (i.e., dans le cas où l'événement esclave ne peut se produire).

4.4.2 Exemple

Considérons le document $SMIL$ décrit dans la $Figure$ 4.3 présenté dans [8], et qui consiste en une séquence seq_1 d'une image Img_1 suivie par la présentation parallèle par_1 d'un clip audio Aud_1 et d'un texte Txt. Cette séquence doit être présentée simultanément avec deux autres séquences : La première des deux séquences seq_2 montre un clip vidéo Vid_1 suivi de la présentation parallèle par_2 d'une image Img_2 et la double répétition d'un clip audio Aud_2. L' autre séquence présente un clip vidéo Vid_2 qui débute $4s$ après le début de Vid_1. Nous assumons que la durée de Img_1, Aud_1, Vid_1, et Aud_2 sont respectivement, 5, 10, 8 et 5 secondes et que les médias Img_2, Txt, et Vid_1 n'ont pas de durées explicites.

De plus, notons que le début du bloc par_2 conditionne la fin du média Aud_1 et que Aud_2 doit commencer lorsque le média Aud_1 termine. Aussi, la présentation de Vid_1 peut être interrompue par l'interaction d'un utilisateur qui *pourrait* se produire entre 4 et 6 secondes après le lancement

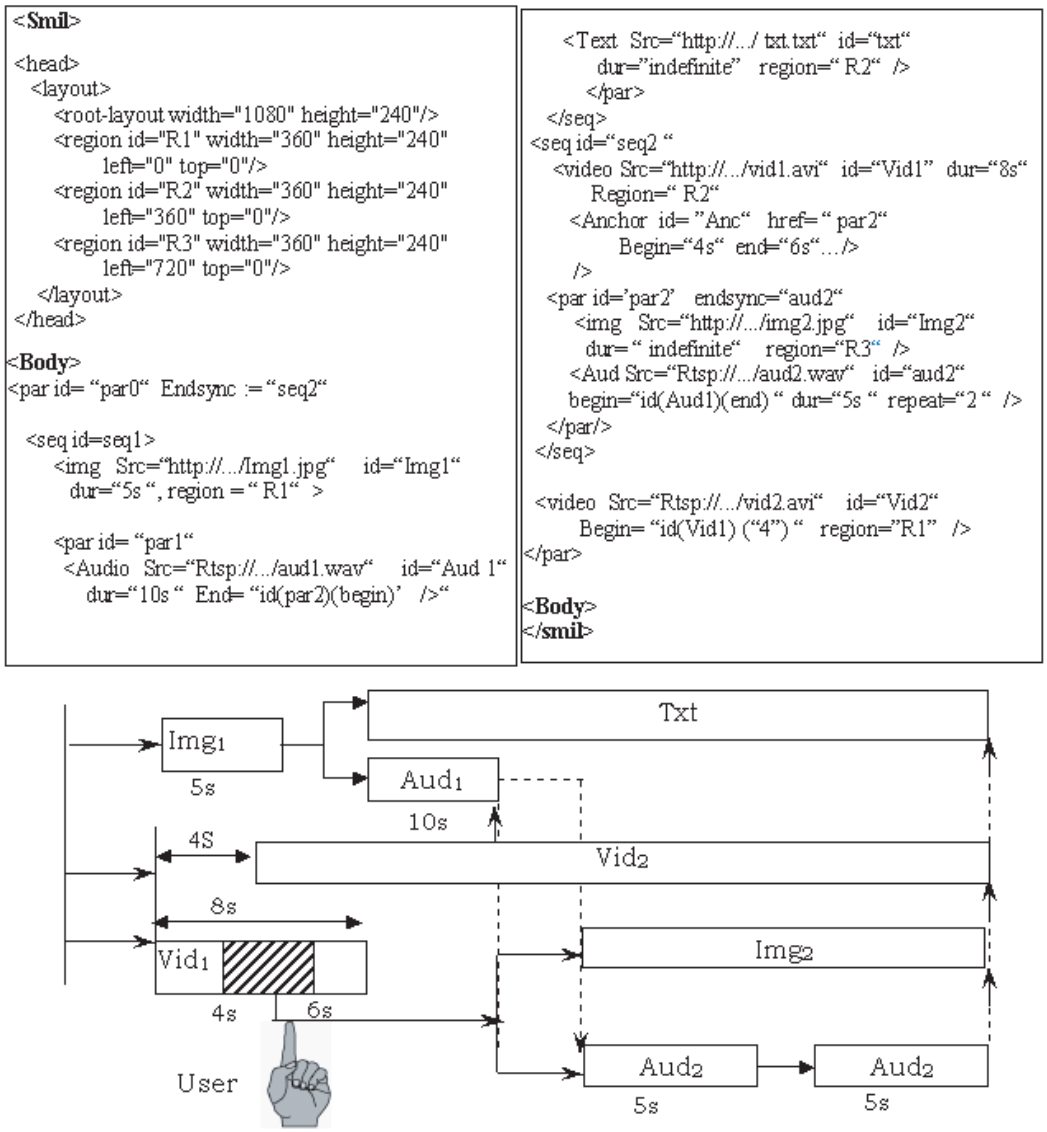

FIGURE 4.3 – Une présentation multimédia décrite au moyen du langage SMIL.

de Vid_1. Par ailleurs, nous assumons que la fin de la séquence seq_2 est déterminée par la fin de Aud_2 et que la fin de la présentation globale est déterminée par la fin de la séquence seq_2. Par conséquent, la présentation $SMIL$ termine quand Aud_2 termine. Pous les contraintes spaciales, il est à noter que la présentation définit trois régions R_1, R_2 et R_3 dans lesquelles les différentes présentations (des médias à visualiser) doivent être affichées. La région R_1 est dédiée à Img_1 et Vid_2 alors que la région R_2 est allouée aux médias Vid_1 et Txt. Finalement, la région R_3 est utilisée par le média Img_2.

4.4.3 Modélisation des contraintes d'un document $SMIL$

Les contraintes du document $SMIL$ de la $Figure$ 4.3 peuvent être modélisées de deux manières, selon le fonctionnement du lecteur adopté. Par exemple le $STPTPN$ de la $Figure$ 4.4, dépeint une modélisation basée sur un fonctionnement épousant celui du lecteur $Real-one$ [81]. Dans cela, les contraintes relatives à chaque média sont spécifiées par une "*unité*". Ensuite, les différentes *unités* sont organisées selon la structuration du document $SMIL$. Initialement, les places standards P_{14},

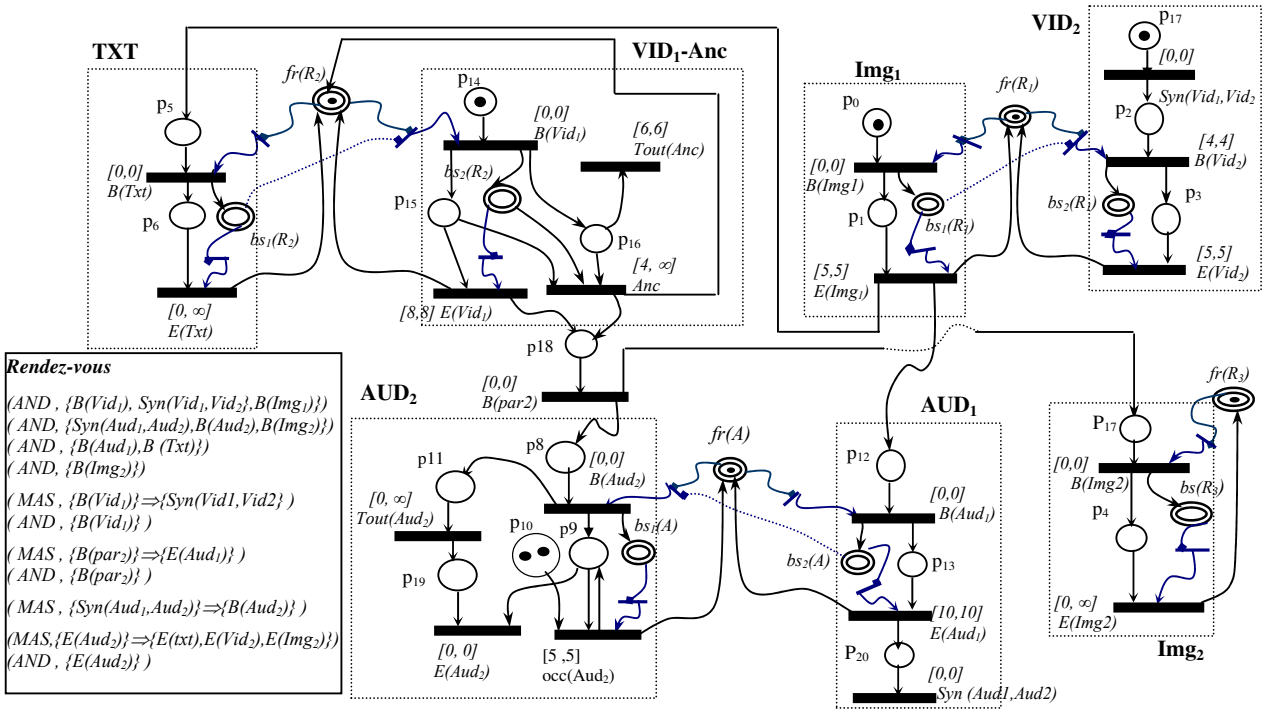

FIGURE 4.4 – Le *STPTPN* modélisant les contraintes de la présentation *SMIL*, selon le fonctionnement du lecteur *Real One*.

P_0 et P_{17} sont marquées dénotant l'activation des unités Vid_1, Img_1 et Vid_2, respectivement. Concernant les contraintes sur les ressources non critiques, elles sont modélisées en utilisant des places ressource et des hyperarcs préempteurs. De plus, l'ordre de priorité associé aux unités correspond à l'ordre ascendant des introductions des déclarations des médias dans le code du document *SMIL*. Par conséquent, les médias Vid_1, Vid_2 et Aud_2 ont la priorité sur respectivement les médias Txt, Img_2 et Aud_1 pour les ressources en conflit. Rappelons que les ressources sujettes aux conflits dans une présentation $SMIL$ sont de deux types :

- Les *fenêtres de visualisation* matérialisées par la définition des trois régions R_1 R_2 et R_3, dont les schémas d'allocation sont modélisés par des places ressource. Par exemple, la région R_2 est modélisée par l'ensemble des places ressource $(fr(R_2), bs_1(R_2), bs_2(R_2))$. Notons que la place $fr(R_2)$ dénote une fois marquée que la région R_2 est disponible alors que la place $bs_1(R_2)$ respectivement, $bs_2(R_2)$ dénote une fois marquée que R_2 est utilisée pour afficher le média Txt respectivement, le média Vid_1. Cependant, Vid_1 est prioritaire par rapport au média Txt. Cette priorité est établie grâce à l'arc de violation partant de la place ressource $bs_1(R_2)$ à la transition $B(Vid_1)$.
- Le *canal audio* noté "A", est spécifié par le schéma d'allocation $(fr(A), bs_1(A), bs_2(A))$ et est sujet de conflit entre les médias Aud_1 et Aud_2 ; La priorité est donnée ici au média Aud_1.

Les contraintes de synchronisation $SMIL$ sont spécifiées par l'ensemble des rendez-vous. Chaque rendez-vous peut prendre comme type AND ou MAS. Pour ce faire, des transitions assurant les synchronisations entre unités sont rajoutées ; la transition $Syn(Vid_1, Vid_2)$ est utilisée dans l'unité Vid_2 pour modéliser la synchronisation des débuts des médias Vid_1 et Vid_2, puisque Vid_2 doit commencer 4 secondes après Vid_1. De même, la transition $Syn(Aud_1, Aud_2)$

est utilisée dans l'unité Aud_1 pour modéliser la synchronisation entre la fin de Aud_1 et le début de Aud_2.

De plus, pour modéliser la synchronisation entre les débuts de Img_2 et Aud_2 (i.e, bloc Par_2), et la fin de Aud_1, on ajoute la transition $B(par_2)$ dénotant l'événement de début du bloc parallèle Par_2 devant se synchroniser avec la transition $E(Aud_1)$. Il est à remarquer que la transition $B(par_2)$ est connectée en entrée à l'unité Vid_1 et en sortie aux unités Img_2 et Aud_2 selon la structuration donnée en *Figure* 4.3.

Par ailleurs, d'autres rendez-vous sont rajoutées pour exprimer l'évolution synchrone de certains événements comme par exemple celles dénotant l'exécution synchrone des débuts de Aud_1 et Txt d'une part, et des évenements $B(Img_1)$, $B(Vid_1)$ et $Syn(Vid_1, Vid_2)$ d'autre part [10]. Plus concrètement, la réalisation du rendez-vous $(AND, \{B(Img_1), B(Vid_1), Syn(Vid_1, Vid_2)\})$ permet de décrire le cas où les média Img_1 et Vid_1 synchronisent leur débuts. Dans le cas contraire les rendez-vous $(AND, \{(Img_1)\})$ et $(MAS, \{B(Vid_1)\} \implies \{Syn(Vid_1, Vid_2)\})$ se réalisent [11]. Aussi, le même ojectif est assigné au rendez-vous $(AND, \{B(Img_2), B(Aud_2), Syn(Aud_1, Aud_2)\})$.

Aussi, il est à rappeler qu'une contrainte de synchronisation $SMIL$ de type $A :=$ "B" dénote que l'événement A peut se réaliser **seulement si** l'événement B se réalise aussi. Cependant, B ne requiert pas l'occurrence de A pour se produire. Cette sémantique ne cadre pas exactement avec celle d'un $STPTPN$. Par conséquent, chaque synchronisation $SMIL$ de type *Maître* est modélisée par la donnée de deux rendez-vous ; un pour modéliser l'occurrence de la synchronisation *Maître,* lorsque tous les événements sont disponibles ; et le second rendez-vous pour modéliser le cas où l'événement maître se produit de manière asynchrone, en raison de la non disponibilité de l'événement esclave.

Par exemple, le début du média Aud_1 est conditionné par la fin du bloc Par_2. Cette synchronisation est spécifiée par le rendez-vous *Maître* suivant $(MAS, \{B(Par_2)\} \Rightarrow \{E(Aud_1)\})$. Cependant, si la synchronisation ne peut avoir lieu, alors $B(par_2)$ peut se réaliser de manière asynchrone selon le rendez-vous $(AND, B(par_2))$ [12].

A d'autres égards, la modélisation de la contrainte ressource de l'ancre associé à Vid_1 est modélisée en connectant à la transition Anc, la place $bs_2(R_2)$ en entrée et la place $fr(R_2)$ en sortie, via des arcs standards; dénotant la nécessité de disposer de la ressource région R_2 pour l'occurrence de l'interaction de l'utilisateur.

Par ailleurs, la modélisation des mêmes contraintes en considérant un fonctionnement adoptant la sémantique du lecteur *Ambulant* [27] pourrait être dépeinte par le $STPTPN$ de la *Figure* 4.5. Dans cela, nous reprenons la même structuration des unités que dans le cas précédent, mais en modifiant l'ordre des priorités et les mécanismes d'allocations des différentes ressources. Notons que les priorités sont inversées et que par conséquent les médias Txt, Img_2 et Aud_1 sont prioritaires sur respectivement les médias Vid_1, Vid_2 et Aud_2. De plus, la modélisation de l'aptitude de ces derniers à recouvrer les ressources après leur libération est spécifiée par l'ajout de l'arc de violation reliant respectivement, les places $fr(R_2), fr(R_1)$, et $fr(A)$ respectivement, en entrée des

10. Ces synchronisations permettent d'éliminer les comportements d'entrelacement dans le graphe d'accessibilité.

11. Le rendez-vous $(AND, \{B(Img_1), B(Vid_1), Syn(Vid_1, Vid_2)\})$ est plus prioritaire que les rendez-vous $(AND, \{B(Img_1)\})$ et $(MAS, \{B(Vid_1)\} \implies \{Syn(Vid_1, Vid_2)\})$ lorsqu'ils sont offerts pour les mêmes dates, puisque le premier l'inclut strictement les derniers.

12. Le rendez-vous $(MAS, \{B(Par_2)\} \Rightarrow \{E(Aud_1)\})$ a une priorité supérieure que $(AND, B(Par_2))$ du fait qu'il inclut strictement. Il en résulte que le second rendez-vous peut se réaliser que pour les instants où le premier n'est pas offert (priorité en temps réel)[11].

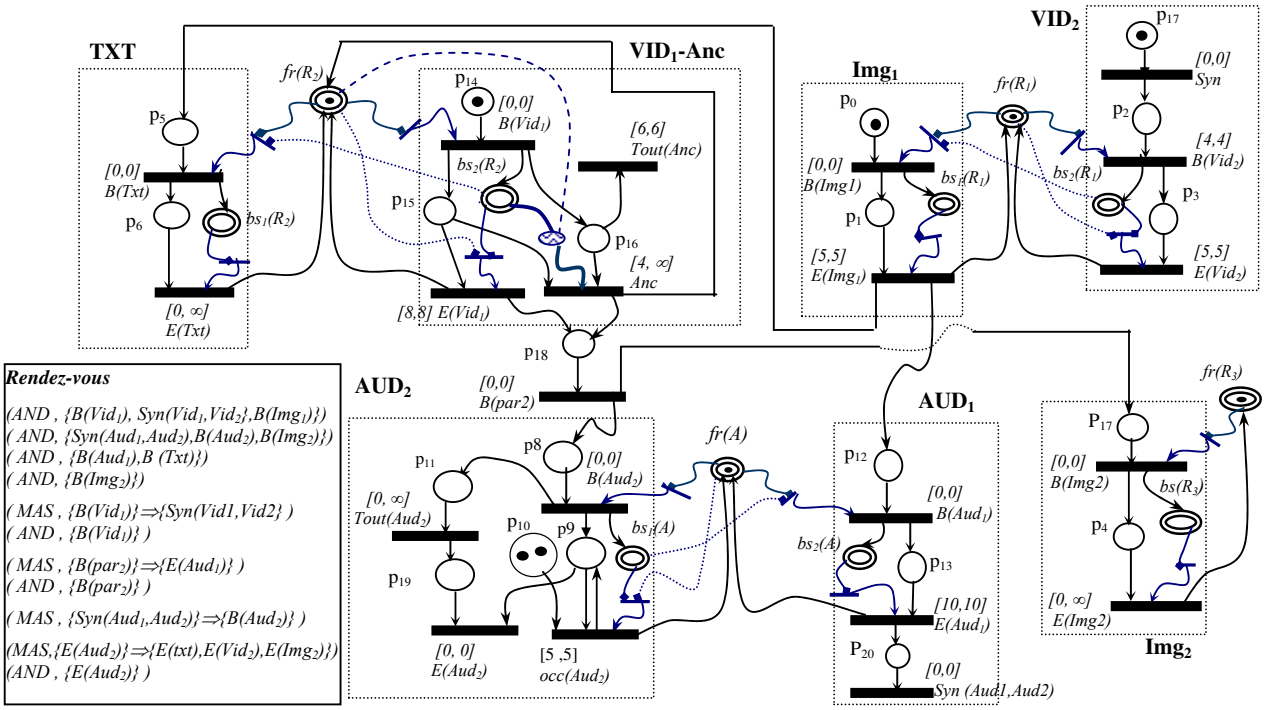

FIGURE 4.5 – Le *STPTPN* modélisant les contraintes de la présentation *SMIL*, selon le fonctionnement du lecteur *Ambulant*.

transitions $E(Vid_1)$, $E(Vid_2)$, et $E(Aud_2)$.

De plus, pour modéliser les contraintes ressource associées au lien, nous connectons en entrée de la transition *Anc*, les places ressource $bs_2(R_2)$ en utilisant l'arc fort et $fr(R_2)$ via l'arc faible de l'*hyperarc de franchissement* [13].

En conclusion, nous avons montré grâce au processus de modélisation présenté dans le chapitre II, comment exploiter les modèle défini pour capturer les différentes contraintes (aussi complexes soit elles), caractérisant les documents multimédia *SMIL*. Ensuite, la disponibilité d'un algorithme énumérant les comportements du système permettra de détecter les éventuelles erreurs contenues dans le document, dont l'origine est une mauvaise caractérisation au niveau de l'édition des contraintes de temps, de synchronisation, voire des allocations des ressources. Ainsi, l'identification de la cause de ces inconsistances permettra de les corriger en procédant au formattage du code source du document *SMIL*. De plus, le graphe dérivé de la spécification *STPTPN* pourrait être utilisé comme un outil de décision pour la programmation et la lecture appropriées des scénarios consistants (s'ils existent), en considérant les durées d'exécutions de chacun [6]. D'autre part, il permettrait de déduire un Pattern d'accès [12] pouvant être utilisé dans la gestion d'un schéma de délivrance ou de préchargement applicable à différents niveaux (client [8], serveur [9][13][22]). Cet aspect de l'analyse des documents multimédia est décrypté dans le *Chapitre.VI*.

13. Les *arcs de conditionnement de franchissement* ne sont pas utilisés car nous avons à faire à deux médias conflictuels.

4.5 Comparaison avec d'autres modèles

Dans cette section, nous comparons le modèle $STPTPN$ avec d'autres modèles existants ; particulièrement ceux dédiés à la modélisation des applications multimédia :

- **Les automates temporisés [24] :** Les automates temporisés [24] particulièrement la classe des automates hybrides [25], définit un modèle puissant permettant l'expression de plusieurs sémantiques introduites par les $STPTPN$ comme par exemple celle des *chronomêtres* [46]. Cependant, les automates temporisés sont des modèles purement orientés événements avec une abstraction totale des notions de ressources ou de synchronisation. Au fait, ces modèles ne sont pas dédiés à la base pour la spécification, car ne possédant pas l'expressivité naturelle des modèles algébriques ou des réseaux de Petri. Ils sont utilisés comme des modèles intermédiares associés à des logiques temporelles sur lesquels elles s'appliquent dans le but de vérifier et d'analyser les propriétés des spécifications de base [98].
- Le langage **RT-LOTOS [49] :** Les spécifications algébriques ont été particulièrement utilisées pour modéliser les contraintes logiques et temporelles des systèmes multimédia. Par exemple, le langage $RT-LOTOS$ qui est l'un des langages algébriques les plus expressifs a été largement utilisé pour la modélisation de ces systèmes [44][47][51]. Cependant, $RT-LOTOS$ n'est pas pourvu de mécanismes tels ceux permettant la suspension du temps, ou ceux permettant la gestion de la préemption de ressource. D'ailleurs, le langage $RT-LOTOS$ n'a été utilisé que pour l'analyse de la cohérence temporelle des systèmes multimédia sans prendre en compte les problématiques soulevées dans cette thèse.
- Le *Modèle* $TSPN$ [53][87] : Le modèle $TSPN$ défini pour la modélisation des systèmes multimédia faiblement synchrones semble être le modèle le plus apparenté au $STPTPN$. Par conséquent, la comparaison entre les deux modèles nous permet de faire valoir quelques différences de fond : Dans un $TSPN$, les intervalles de temps sont associés aux arcs entrants. De plus, une place dans un $TSPN$ modélise un processus du système appelé flux ou *stream* (voir *Figure*.2.12). Chaque schéma de synchronisation entre les différents processus se présente par une régle associée à la transition correspondante. De là, un $TSPN$ ne peut être considèré comme une extension directe d'un $RdPT$. Aussi, il n'est pas adapté à la modélisation des spécifications orientées événements puisque chaque action y est modélisée comme une place. Plus encore, certains comportements ne peuvent être modélisés intuitivement avec un $TSPN$, menant à des spécifications ambiguës [41]. Comme cas particulier, la modélisation des ressources d'un processus dans un $TSPN$ est assez floue, car une place y est utilisée pour la représentation d'un processus ou d'une action. Par conséquent, comme un $STPTPN$ permet de prendre en compte les schémas de synchronisation définis dans un $TSPN$, et ce en plus d'autres mécanismes tels que la suspension du temps et les mécanismes de préemption des ressources (non supportés par un $TSPN$), il offre ainsi un outil plus puissant et plus adapté pour modéliser un large panel dépassant le cadre des systèmes dits *faiblement synchrones*, et cela tout en préservant l'expressivité intuitive qui a permi aux $RdPT$ d'être ce modèle pratique si largement utilisé.

Toutefois, les langages précédents sont dotés pour certains d'entre eux d'outils et de techniques permettant leurs éditions ainsi que leurs analyses, alors que le $STPTPN$ est toujours en cours de validation. Etant conscient de ces lacunes, et dans le but de rendre ce modèle pratique, nous

devons doter ce dernier de techniques permettant son analyse et un environnement réalisant les différents stages de la vérification :

- La proposition d'un algorithme permettant de dériver le graphe d'accessibilité du $STPTPN$. Dans ce sens, deux problématiques doivent êre considérées : La première est liée à l'utilisation des chronomètres dont la sémantique est connue pour être une entrave à la construction du graphe d'accessibilité. Pour cet effet, différentes approches [42][72] ont été proposées dans la littérature permettant de faciliter le calcul de l'espace d'état par sa *sur-approximation*. Ces dernières sont envisagées pour être adaptés au $STPTPN$. La deuxième problématique est liée à la sémantique de synchronisation qui implique d'étendre les algorithmes existants [32][94] pour le traitement des sémantiques de franchissements complexes induites par l'introduction de ces différents mécanismes ;
- Un module d'édition doté d'une interface graphique ;
- Un module de traduction des documents multimédia (*SMIL* par exemple) vers leurs équivalents $STPTPN$;
- Un interpréteur permettant de traduire un $STPTPN$ vers son graphe d'accessibilité correspondant et d'y vérifier les propriétés de la spécification ;
- Un simulateur dont le fonctionnement est basé sur la sémantique formelle du modèle, réalisant des tests de simulation en générant les traces d'un $STPTPN$.

4.6 Conclusion

Nous avons présenté dans ce chapitre, à travers sa syntaxe et sa sémantique formelles un modèle permettant d'étendre les *RdPT* à différents mécanismes, et ce, dans le but de modéliser un large panel de systèmes temps réel et en particulier des applications multimédia. Nous avons explicité la puissance d'expression de cette proposition à travers son utilisation pour la spécification des contraintes et des exigences des documents multimédia décrits au moyen du langage $SMIL$ [90]. Néanmoins, l'utilisation opérationnelle de ce modèle ne pourrait être productive que par la dotation de ce dernier de techniques et d'algorithmes permettant la dérivation d'un graphe d'accessibilité capturant les comportements de la spécification de base. D'ailleurs, ce besoin fera l'objet d'une proposition abordée dans le prochain chapitre, permettant l'énumération exhaustive de l'espace d'état d'un $STPTPN$ ainsi que son analyse.

Chapitre 5

Construction du graphe d'accessibilité d'un $STPTPN$

Nous proposons dans le présent chapitre une approche d'énumération de l'espace d'état d'un $STPTPN$. Nous discutons des problématiques et des solutions envisagées, liées à l'introduction des sémantiques des chronomètres et de synchronisation. Une première approche basée sur la représentation en polyèdre des systèmes de contraintes est déclinée, permettant de construire le graphe des classes en se basant sur deux niveaux de contraction. D'abord, en regroupant des états accessibles après franchissement de la même séquence de rendez-vous ; Ce regroupement est appelée *classe d'état*. Ensuite, en rassemblant les classes d'états proposant les mêmes séquences de franchissements dans le futur. Ces deux niveaux de contractions permettent dans la majorité des cas de construire un graphe fini. Toutefois, la finitude du graphe reste une propriété indécidable pour un $STPTPN$. Une seconde approche moins couteuse implémentant la *sur-approximation DBM* est proposée, permettant de préserver un sous ensemble de propriétés linéaires du modèle. Cette dernière approche construit un graphe *sur-approximé,* si le modèle de base contient des arcs inhibiteurs et un graphe *exact* dans le cas contraire.

5.1 Problématique et méthodologie

Le graphe des états est obtenu en utilisant la règle de franchissement de la $Définition$ 13. Les noeuds du graphe représentent les états accessibles du $STPTPN$, et les arcs dénotent la relation d'accessibilité entre états. Deux noeuds e_n et e_{n-1} sont reliés par un arc si $e_{n-1} \xrightarrow{L(r_n)} e_n$. En considérant un temps dense, le graphe est infini et à branchement infini. Par conséquent, nous avons besoin de contracter le graphe de telle sorte que cette contraction préserve les propriétés les plus importantes comme l'accessibilité, le blocage, la vivacité,...etc.

Dans le cas d'un $RdPT$, le graphe des états est aussi infini mais peut être contracté en regroupant dans une même classe tous les états ayant été obtenus après le franchissement de la même séquence. Cette approche de construction dite méthode du *graphe des classes* [73], représente chaque classe comme une paire (M, D), où M est le marquage courant et D est un ensemble d'inéquations codées sous forme DBM *(Difference Bound Matrix)* [54]. Cette représentation permet d'appliquer un algorithme efficace pour le calcul des classes accessibles et pour le test d'équivalence. Cette forme exprime chaque inéquation soit par $\underline{t_i} - \underline{t_j} \preceq c_{ij}$ ou bien par $c_{i\bullet} \preceq$

$\underline{t_i} \preceq c_{\bullet i}$ avec $c_{ij} \in Q \cup \{\infty\}$, $c_{\bullet i} \in Q^+ \cup \{\infty\}$ et $c_{i\bullet} \in Q^+$. Le calcul d'une classe accessible dans sa forme minimale DBM (dite *forme canonique*[1]) est toujours possible et obtenue en appliquant un algorithme du plus court chemin dont le plus célèbre est celui de $Floyd - Warshall$ dont la complexité est $o(l^3)$ où (l) étant le nombre de variables dans D [32]. Des alternatives à cet algorithme ont été proposées dans [37][94] permettant de réduire la complexité de calcul d'une classe à $o(l^2)$. Ces algorithmes offrent aussi la possibilité de calculer les délais minimaux et maximaux de n'importe quelle séquence de tir en un temps de complexité estimé à $o(l \times k)$ où k est la longueur de la séquence.

Par ailleurs, nous avons montré dans la $Section.II.4.2$ que l'introduction de la sémantique des chronomètres faisait que l'algorithme d'énumération ne permettait plus d'exprimer le système D sous forme DBM. En d'autres termes, pour certaines classes accessibles l'espace des solutions D ne peut être encodé sous cette forme. Au fait, il ne peut être capturé que par un système de contraintes général de *forme polyédrique convexe*, dont la résolution est exponentielle au nombre de variables présentes (transitions sensibilisées). Pour illustrer ce fait, considérons l'exemple de la *Figure 2.11,* et soit le système D obtenu après franchissement des transitions t_1, t_4 et t_5.

$$D : \begin{cases} 0 \preceq \underline{t_2} \preceq 4, & -4 \preceq \underline{t_2} - \underline{t_3} \preceq 4, \\ 0 \preceq \underline{t_3} \preceq 4, & -4 \preceq \underline{t_2} - \underline{t_6} \preceq 0, \\ 4 \preceq \underline{t_6} \preceq 4, & -4 \preceq \underline{t_3} - \underline{t_6} \preceq 0, \\ & 1 \preceq \underline{t_2} + \underline{t_3} \preceq 6 \end{cases}$$

Ce dernier contient l'inéquation $1 \preceq \underline{t_2} + \underline{t_3} \preceq 6$ qui a une forme polyédrique. De ce fait, il est impossible de représenter l'ensemble des solutions de cette classe par un système de forme DBM. De plus, comme pour les systèmes DBM, il a été prouvé qu'un système de forme polyédrique admet une forme canonique unique [28] distinguant clairement deux sous systèmes : le premier de forme DBM et le second capturant toutes les autres contraintes ne pouvant être encodées que par des *polyèdres*. Plus encore, il a été démontré que les variables participant au second sous ensemble de contraintes sont celles relatives exclusivement aux transitions persistantes ; confrontant les transitions *anciennement inhibées* avec les autres transitions[42][72]. Du fait de sa complexité, un tel système consomme un temps exponentiel pour son calcul et sa canonisation. Par ailleurs, ce système nécessite des structures de données lourdes pour être représenté en mémoire. Par conséquent, les approches exactes basées sur la représentation en polyèdres[68] ont montré des insuffisances lorsque les systèmes prennent des formes plus élaborées ; D'ailleurs, les crashages mémoires en plus des temps prohébitifs engendrés par ce type d'algorithmes, en sont une conséquence directe de cette complexité.

Cependant, il a été constaté que dans certains cas les contraintes exclusivement polyédriques sont redondantes et n'influent pas sur les tests de franchissements. A partir de la, des solutions de sur-approximation de l'espace d'état (en éliminant les contraintes ne respectant pas la forme DBM), ont été proposées[42][83]. Ces approches permettent de construire efficacement en un temps réduit, un graphe néanmoins grossier pouvant contenir des *états non accessibles* i.e., induisant des séquences non franchissables dans le modèle de base. Il en demeure que ce graphe permet de préserver un certain nombre de propriétés linéaires notamment celles liées à la sûreté. Pour illustrer cette technique, cela revient à éliminer la contrainte $1 \preceq \underline{t_2} + \underline{t_3} \preceq 6$ dans le système précédent, ce qui autoriserait le tir de la transition t_6 qui est pourtant non franchissable dans le

1. Il est prouvé que tous les systèmes équivalents ont la même forme canonique.

premier système d'où la *sur-approximation.*

Par ailleurs, afin de trouver un compromis entre les deux techniques, une approche hybride a été proposée par *Roux et Autres* [72] ; Cette dernière exploite une condition suffisante détectant les cas où le sous système polyédrique devient redondant par rapport au système *DBM.* Ainsi, la combinaison des deux représentations permet de construire le graphe exact en réduisant les temps de calcul et les crashages mémoire de manière sensible par rapport à la méthode utilisant exclusivement une représentation par des polyèdres. Plus récemment, une nouvelle approche exploitant la technique d'approximation *des grilles* a été développée[33]. Cette dernière permet la résolution des systèmes polyédriques par sur-approximation des valeurs des coefficients. Cette approche est paramétrable et calcule une sur-approximation du polyèdre produisant un graphe exact dans la majorité des cas, en un temps de complexité plus réduit que les autres approches exactes. Néanmoins cette technique reste couteuse par rapport à la sur-approximation *DBM*, même si elle produit des graphes moins grossiers.

Dans le cas d'un $STPTPN$, à la problématique des chronomètres s'ajoute celle liée à l'introduction des mécanismes de synchronisations. Au fait, l'association des sémantiques temporelles faible et forte dans un même modèle, génère des classes inconsistantes dont l'identification est un prérequis au processus d'énumération. De plus, l'addition du concept de rendez-vous dans le modèle, en plus du concept standard de transition rend complexe la formalisation des contraintes de franchissement. L'expression de ce système devrait inférer aussi bien les contraintes des transitions que celles des rendez-vous. En clair, nous avons besoin de déterminer le plus large système capturant toutes les solutions des systèmes $FT_n \wedge FR_n$ associés aux états accessibles dans la même classe. Pour ce besoin, nous proposons d'abord une méthode d'énumération exacte basée sur la formulation de chaque classe accessible E par un couple (M, D), où D est la conjonction de deux sous systèmes l'un de forme *polyédrique* noté $\widehat{D}$, et l'autre de forme DBM noté $\widetilde{D}$; déterminant les contraintes des transitions sensibilisées. De là, les tests de consistance, de franchissement ainsi que le calcul des classes accessibles nécessitent l'extension du système D à un système additif capturant les contraintes des rendez-vous. Cependant, comme la complexité de calcul d'une classe exacte d'un $STPTPN$ est exponentielle aux nombres de variables, l'approche en question ne peut être efficace pour l'analyse des modèles complexes car s'avérant très couteuse en temps de calcul et en espace mémoire.

Pour palier à cet inconvénient, la sur-approximation DBM peut s'avérer une solution alternative pour l'énumération et l'analyse de l'espace d'état des $STPTPN$ complexes. Pour ce faire, de la même manière que pour les $RdPT$ à chronomètres, nous envisageons une approche d'énumération basée sur le relâchement de toutes contraintes sous forme polyédriques pour ne garder que celles sous forme *DBM.* Cela permet de déterminer efficacement des conditions suffisantes pour les tests de consistance, de franchissement et enfin d'équivalence. Cette construction permet de calculer un graphe sur-approximé moins riche [2] mais cependant moins couteux.

Pour cet effet, le calcul d'une classe sur-approximée notée $\widetilde{E}$ pourrait être obtenue en appliquant l'algorithme de $Floyd - Warshall$, dont la complexité est estimée dans le meilleur des cas à $o((m+l)^3)$ où l et m sont respectivement le nombre de transitions sensibilisées et le nombre de rendez-vous fortement sensibilisés pour la classe. Toutefois, l'application de cet algorithme

2. Le graphe préserve un sous ensemble de propriétés linéaires et temps réel du modèle, pouvant être amplement suffisant pour l'analyse prospectée.

reste néanmoins coûteuse. Pour y remédier, nous proposons un algorithme alternatif permettant de réduire la complexité de calcul d'une classe accessible en un temps polynômial d'ordre deux, estimé à $o(l^2+l\times m+m)$ [15][17]. Au fait, l'algorithme que nous proposons permet d'éliminer les calculs redondants dans le même esprit de ce qui a été entrepris auparavant pour les $RdPT$[37][94].

En conlusion, l'approche de sur-approximation proposée permet de calculer le *graphe des classe exact* d'un $STPTPN$ lorsque ce dernier ne contient pas d'arc inhibiteur, et un espace *sur-approximé* dans le cas contraire. Chaque noeud du graphe est une paire $(M,\widetilde{D})$ où $\widetilde{D}$ correspond au système de contraintes DBM relatif aux transitions sensibilisées, à partir du quel nous pouvons déduire celui relatif aux rendez-vous fortement sensibilisés, noté DR. Donc, le modèle évolue ici, en franchissant à chaque pas un rendez-vous impliquant le tir simultané de ses transitions synchronisantes et générant un ensemble d'événements étiquetant l'arc du graphe.

Par aillreurs, le processus d'énumération munie d'une relation d'équivalence basée sur l'égalité des classes permet de construire le graphe contractant l'espace d'état [3] du $STPTPN$. Cette contraction permet de préserver les propriétés linéaires du modèle. Toutefois, en raison de l'introduction des sémantiques des chronomètres et de synchronisation, les graphes exact et sur-approximé obtenus peuvent être infinis même si le réseau est borné, et ce au contraire d'un $RdPT$ dont le graphe des classes est prouvé fini si le modèle de base est borné.

Les approches en question sont développées ci-après.

5.2 Graphe des classes exact d'un *STPTPN*

Le graphe des classes exact d'un $STPPTN$ noté GR est défini comme suit :

Définition 15 *Soit ST un STPTPN. Le graphe des classes exact de ST noté GR est un tuple* $(CE, E_0, L, \longmapsto)$ *où :*

- CE *est l'ensemble des classes accessibles dans GR.*
- E_0 *est une classe de* CE *dite classe initiale.*
- L *est l'ensemble des labels des rendez-vous RDV définis dans ST.*
- $\longmapsto$: *est une relation entre classes définie sur* $CE\times L\times CE$. *On note* $(E,l,E')\in\longmapsto$ *par* $E\overset{l}{\longmapsto}E'$.

Une classe d'état notée E_n est l'ensemble de tous les états accessibles après le franchissement de la même séquence de rendez-vous S_n, mais à des dates différentes. Par conséquent, tous les états d'une même classe ont le même marquage M_n. Formellement, on dit qu'une classe E_n est accessible à partir de la classe E_{n-1} en franchissant r_n et on note $E_{n-1}\overset{L(r_n)}{\longmapsto}E_n$, si chaque état appartenant à la classe E_n est accessible à partir d'un état de E_{n-1}, mais cette relation n'implique pas que chaque état de E_n est accessible à partir de chaque état de E_{n-1}. Au fait, elle garantit que pour un état e_n de E_n il existe au moins un état e_{n-1}de E_{n-1} tel que le rendez-vous r_n peut être franchi à la date relative $\underline{r_n}$, satisfaisant le système suivant :

$$A_{n-1}\equiv\begin{cases}(\forall r'\in\ \Phi_{n-1}(r_n),\quad(\underline{r_n}\prec Low_{n-1}(r'))\wedge FR_{n-1}\\(\forall\ r\in Senable_{n-1},(\ \underline{r_n}\preceq\underline{r})\wedge FR_{n-1}\end{cases}$$

Donc, l'*espace de tir* d'une classe accessible noté D_n détermine l'ensemble dense des valeurs admissibles des horloges associées aux transitions sensibilisées ; ces valeurs sont des solutions pour

3. Dans le cas d'une sur-approximation *DBM,* le graphe obtenu inclut l'espace d'état du *STPTPN.*

les systèmes A_{n-1}. Autrement dit, D_n décrit pour chaque transition sensibilisée l'union de ses domaines de tir lorsqu'on considére tous les états accessibles dans E_n.

Remarque : Notons que la classe initiale E_0 contient seulement l'état initial e_0.

Définition 16 *Soit E_n une classe accessible dans GR après le franchissement de la séquence S_n à partir de l'état initial. La classe E_n est donnée par la paire (M_n, D_n) où M_n est le marquage accessible et D_n est l'espace de tir donné par un système d'inéquations mis sous forme canonique se présentant comme suit :* $D_n = \widetilde{D}_n \wedge \widehat{D}_n$

$$\widetilde{D}_n = \begin{cases} \bigwedge\limits_{\forall t_i,t_j \in Te_n} & (\underline{t_j} - \underline{t_i} \preccurlyeq c_{ij}) \\ \bigwedge\limits_{\forall t_i \in Te_n} & (c_{i\bullet} \preccurlyeq \underline{t_i} \preccurlyeq c_{\bullet i}) \end{cases} \quad c_{ij}, c_{\bullet i} \in Q \cup \{\infty\}\,,\ c_{i\bullet} \in Q^+$$

$$\widehat{D}_n = \wedge_{k=1..p} \quad (\alpha_{1k}\underline{t_{1k}} + .. + \alpha_{sk}\underline{t_{sk}} \preccurlyeq d_{\bullet k}) \ \ où\ \ t_{ik} \in Te_n,\ d_{\bullet k} \in Q \cup \{\infty\}\ et^4\ (\alpha_{1k}, .., \alpha_{sk}) \in Z^s$$

Le *forme canonique* du système D_n est représentée par la conjonction de deux sous systèmes d'inéquations : $\widetilde{D}_n$ représentant les contraintes encodables sous forme DBM et le système $\widehat{D}_n$ regroupant les contraintes ne pouvant être formalisées que sous forme polyédrique.

La classe initiale E_0 est donnée par (M_0, D_0) où $D_0 = \widetilde{D}_0$ et est définie comme suit : $\forall t_i \in Te_0$, $EFT(t_i) \preceq \underline{t_i} \preceq LFT(t_i)$,

La forme du système D_n étant minimale et de surcroît capturant seulement les contraintes des transitions sensibilisées, elle ne permet pas de déduire une condition suffisante efficace pour le test de franchissement. D'ailleurs comme le modéle franchit un rendez-vous, nous avons besoin d'étendre ce premier système aux contraintes des rendez-vous fortement sensibilisés. Pour ce faire, la notation suivante permet d'introduire ces nouvelles contraintes en fonction de la régle de synchronisation associée à chaque rendez-vous.

Notation 2 *Soit $E_n = (M_n, D_n)$ une classe accessible dans GR et r un rendez-vous fortement sensibilisé pour la classe E_n. Nous notons respectivement par $Inf_n(r)$ et $Sup_n(r)$ les contraintes du délai résiduel minimal, et celles du délai résiduel maximal relatives au rendez-vous r.*

$Inf_n(r) \equiv$

$$\begin{cases} Si\ typ(r) \in \{And,\ Wand,\ Amas,\ Async\}\ alors \\ \quad \bigwedge\limits_{\forall t \in Trans(r)} \{\underline{t} \preceq \underline{r}\} \\ Si\ \ typ(r) \in \{Or,\ Sor,\ Omas\}\ alors \\ \quad \bigvee\limits_{\forall t \in Trans(r)} \{\underline{t} \preceq \underline{r}\} \\ Si\ typ(r) \in \{Mas,\ Smas,\ Wmas\}\ \ alors \\ \quad \underline{t_m} \preceq \underline{r} \end{cases}$$

$Sup_n(r) \equiv$

$$\begin{cases} Si\ typ(r) \in \{And,\ Sor,\ Smas, Async\}\ alors \\ \quad \bigwedge\limits_{\forall t \in Trans(r)} \{\underline{r} \preceq \underline{t}\} \\ Si\ typ(r) \in \{Wand,\ Or,\ Wmas\}\ alors \\ \quad \bigvee\limits_{\forall t \in Trans(r)} \{\underline{r} \preceq \underline{t}\} \\ Si\ typ(r) \in \{Mas,\ Omas,\ Amas\}\ alors \\ \quad \underline{r} \preceq \underline{t_m} \end{cases}$$

La variable $\underline{r}$ désigne l'instant pour lequel le rendez-vous r peut être franchi. Donc, en assumant la *Définition 10*, *l'intervalle de validité temporel* d'un rendez-vous permet de regrouper tous ces instants, et les bornes de cet intervalle peuvent être formulées sous forme de conjonction ou de disjonction de contraintes. Par exemple, si on suppose que $typ(r) = Wand$ alors la contrainte de r peut s'ecrire en fonction des variables transitions $\underline{t}$ comme suit : $\underset{\forall t \in Trans(r)}{MAX} \{\underline{t}\} \preceq \underline{r} \preceq \underset{\forall t \in Trans(r)}{MAX} \{\underline{t}\}$. Ensuite cette dernière inéquation peut être formulée comme une conjonction de deux contraintes,

4. Z dénote l'ensemble des entiers relatifs.

$Inf_n(r) \wedge Sup_n(r)$ telles que, $Sup_n(r) \equiv \left\{ \underline{r} \preceq \underset{\forall t \in Trans(r)}{MAX} \{\underline{t}\} \right.$ et $Inf_n(r) \equiv \left\{ \underset{\forall t \in Trans(r)}{MAX} \{\underline{t}\} \preceq \underline{r} \right.$. Enfin, il suffit d'écrire le MAX du $Sup_n(r)$ respectivement, le MAX de $Inf_n(r)$ par une disjonction, respectivement une conjonction de contraintes. Les formules des autres types de rendez-vous se déduisent de la même manière.

Nous explorons maintenant comme exploiter la notation précédente pour la définition d'une condition suffisante et nécessaire pour le test de franchissement. Mais avant de discuter cette condition, nous devons d'abord vérifier la consistance temporelle de chaque classe accessible E_n, i.e., nous devons vérifier que la classe accessible E_n contient au moins un état qui soit *temporellement consistant*. Dans le cas contraire, la classe sera dite "*temporellement inconsistante*", notée par $E_n \not\models^t$. Pour ce faire, le théorème suivant introduit une condition pour le test de consistance d'une classe.

Notation 3 *Soit $E_n = (M_n, D_n)$ une classe accessible. Nous notons par $\overline{R_n}$ le vecteur des variables associées aux rendez-vous fortement sensibilisés pour M_n.*

Théorème 2 *Soit $E_n = (M_n, D_n)$ une classe accessible dans GR après le franchissement de la séquence S_n, : $E_n \not\models^t$, Ssi il n'existe pas de valuation sur $Q^{|Senable_n|}$ pour le vecteur $\overline{R_n}$ satisfaisant le système $D_n \wedge \left\{ \underset{\forall r \in Senable_n}{\wedge} (\underline{r} \succeq 0) \wedge Sup_n(r) \right\}$.*

Preuve. La preuve se déduit à partir de la *Définition 12* ; une classe est *temporellement inconsistante* ssi tous ses états sont *temporellement inconsistants i.e,* $\nexists e \in E_n$, tel que il existe une valuation sur $Q^{|Senable_n|}$ pour le vecteur $\overline{R_n}$ satisfaisant le système $\underset{\forall r \in Senable_n}{\wedge} \{(\underline{r} \succeq 0) \wedge Sup_n(r)\}$.

Par ailleurs, le système $D_n \wedge \underset{\forall r \in Senable_n}{\wedge} \{Sup_n(r)\}$ détermine l'ensemble des vecteurs $\overline{R_n}$ relatif à tout état accessible dans E_n, satisfaisant les contraintes de la borne supérieue. Par conséquent, $E_n \not\models^t$ ssi $D_n \wedge \left\{ \underset{\forall r \in Senable_n}{\wedge} \{\underline{r} \succeq 0\} \wedge Sup_n(r) \right\}$ n'admette pas de solution sur $Q^{|Senable_n|}$ pour le vecteur $\overline{R_n}$. ■

Le modèle ne peut évoluer à partir d'une classe *temporellement inconsistante* ; la progression du temps est bloquée et aucun rendez-vous ne peut être réalisé car les conditions de franchissements ne sont pas satisfaites pour aucun rendez-vous fortement sensibilisé. Par conséquent, toutes les classes *temporellement inconsistantes* accessibles dans un $STPTPN$ sont équivalentes puisqu'elles n'autorisent aucun franchissement. Donc, lors de la construction du graphe des classes d'un $STPTPN$, l'algorithme d'énumération devra opérer seulement sur les classes *temporellment consistantes*[5] en utilisant le test de franchissement énoncé dans le théorème suivant.

Théorème 3 *Soit $E_{n-1} = (M_{n-1}, D_{n-1})$ une classe temporellement consistante accessible dans GR après le franchissement de la séquence S_{n-1}. Un rendez-vous r_n peut pêtre franchi à partir de E_{n-1},* Ssi *:*

(i) $r_n \in Senable_{n-1}$;

(ii) Il n'existe une valuation sur $Q^{|Senable_{n-1}|}$ pour le vecteur $\overline{R_{n-1}}$ satisfaisant le système $D_{n-1} \wedge CF_{n-1}(r_n)$, tel que :

$$CF_{n-1}(r_n) \equiv \bigwedge \left\{ \begin{array}{l} \underset{\forall r \in Senable_{n-1}}{\wedge} Sup_{n-1}(r) \wedge Inf_{n-1}(r) \wedge (r_n \preceq r) \wedge (0 \preceq r) \\ \underset{\forall r \in \Phi_{n-1}(r_n)}{\wedge} CBI_{n-1}(r_n, r) \end{array} \right.$$

5. Une classe *temporellement consistante* est celle qui ne vérifie pas le *Théorème 2*.

avec

$$CBI_{n-1}(r_n,r) \equiv \left\{ \begin{array}{l} Si\ typ(r) \in \{And,\ Wand,\ Amas,\ Async\}\ alors \\ \quad \bigvee\limits_{\forall t \in Trans(r)} \left\{ \begin{array}{ll} \underline{r_n} \prec \underline{t} + \alpha(t)) & \vee \\ \underline{r_n} \prec 0 & \end{array} \right. \\ Si\ \ typ(r) \in \{Or,\ Sor,\ Omas\}\ alors \\ \quad \bigwedge\limits_{\forall t \in Trans(r)} \left\{ \begin{array}{ll} \underline{r_n} \prec \underline{t} + \alpha(t)) & \vee \\ \underline{r_n} \prec 0 & \end{array} \right. \\ Si\ typ(r) \in \{Mas,\ Smas,\ Wmas\}\ \ alors \\ \quad \left\{ \begin{array}{ll} \underline{r_n} \prec \underline{t_m} + \alpha(t_m)) & \vee \\ \underline{r_n} \prec 0 & \end{array} \right. \end{array} \right.$$

où : $\alpha(t) = \quad EFT(t) - LFT(t)$.

Preuve. Le rendez-vous r_n est franchissable ssi il existe au moins un état $e_{n-1} \in E_{n-1}$ tel que e_{n-1} franchit r_n i.e, vérifie les contraintes de la *Définition 13* : $\exists \overline{R_{n-1}} \in Q^{|Senable_{n-1}|}$, satisfaisant le système :

$FR_{n-1} \wedge \left(\bigwedge\limits_{\forall r \notin \Phi_{n-1}(r_n)} \underline{r_n} \prec Low_{n-1}(r) \right) \wedge \left(\bigwedge\limits_{\forall r \in Senable_{n-1}} \underline{r_n} \preceq \underline{r} \right)$.

Rappelons que FR_{n-1} est le système associé à l'état e_{n-1} représentant les contraintes des rendez-vous fortement sensibilisés.

Comme le système étendu $D_{n-1} \wedge \bigwedge\limits_{\forall r \in Senable_{n-1}} \{Sup_{n-1}(r) \wedge Inf_{n-1}(r) \wedge (0 \preceq r)\}$ détermine l'espace des vecteurs $\overline{R_{n-1}}$ satisfaisant tout système FR_{n-1} relatif à tout état *temporellement consistant* accessible dans E_{n-1}. Par conséquent, r_n est franchissable à partir de E_{n-1} ssi il existe une valuation de $\overline{R_{n-1}}$ satisfaisant le système :

$$\left(D_{n-1} \wedge \left(\bigwedge_{\forall r \in Senable_{n-1}} \{Sup_{n-1}(r) \wedge Inf_{n-1}(r)\} \wedge (0 \preceq r) \right) \right) \wedge \left(\bigwedge_{\forall r \in Senable_{n-1}} \underline{r_n} \preceq \underline{r} \right) \wedge \left(\bigwedge_{\forall r \notin \Phi_{n-1}(r_n)} \underline{r_n} \prec \underset{\forall e_{n-1} \in E_{n-1},}{MAX} Low_{n-1}(r) \right).$$

Posons $CBI_{n-1}(r_n,r) \equiv \bigwedge\limits_{\forall r \notin \Phi_{n-1}(r_n)} \underline{r_n} \prec \underset{\forall e_{n-1} \in E_{n-1},}{MAX} Low_{n-1}(r)$

Ensuite, nous remplaçons $Low_{n-1}(r)$ par son expression selon la règle de synchronisation associée à r. Par exemple, si $typ(r) = Or$, nous obtenons $\underline{r_n} \prec \underset{\forall t \in Trans(r)}{MIN} \{x_{n-1}(t)\}$. Par ailleurs, en utilisant la *Définition 14*, nous prouvons que : $x_{n-1}(t) = MAX(y_{n-1}(t) + \alpha(t), \quad 0)$, tel que $\alpha(t) := EFT(t) - LFT(t)$. A partir de là, nous pouvons écrire

$$CBI_{n-1}(r_n,r) \equiv \bigwedge_{\forall r \notin \Phi_{n-1}(r_n)} \left\{ \underline{r_n} \prec \underset{\forall e_{n-1} \in E_{n-1}, \forall t \in Trans(r)}{MAX} \underset{}{MIN} \underset{}{MAX} \left(\begin{array}{l} y_{n-1}(t) + \alpha(t) \\ 0 \end{array} \right) \right\}$$

$$CBI_{n-1}(r_n,r) \equiv \bigwedge_{\forall r \notin \Phi_{n-1}(r_n)} \underset{\forall e_{n-1} \in E_{n-1},}{MAX} \left\{ \bigwedge_{\forall t \in Trans(r)} \left\{ \begin{array}{ll} \underline{r_n} \prec y_{n-1}(t) + \alpha(t) & \vee \\ \underline{r_n} \prec 0 & \end{array} \right\} \right.$$

Comme $y_{n-1}(t)$ représente la valeur optimale de la variable $\underline{t}$ pour l'état e_{n-1}, et comme toutes les valeurs de $\underline{t}$ sont données par l'espace des solutions du système D_{n-1}, par conséquent il suffit de remplacer $y_{n-1}(t)$ par la variable $\underline{t}$ prenant ses valeurs dans l'espace D_{n-1}.

De là, nous pouvons écrire

$$CBI_{n-1}(r_n,r) \equiv \bigwedge_{\forall r \notin \Phi_{n-1}(r_n)} \bigwedge_{\forall t \in Trans(r)} \left\{ \begin{array}{ll} \underline{r_n} \prec \underline{t} + \alpha(t) & \vee \\ \underline{r_n} \prec 0 & \end{array} \right. \quad \blacksquare$$

Le franchissement du rendez-vous r_n à partir d'une classe temporellement consistante, implique le tir simultané de toutes les transitions se synchronisant dans r_n, telle que la classe accessible est calculée en utilisant le théorème suivant.

Théorème 4 *Soit $E_{n-1} = (M_{n-1}, D_{n-1})$ une classe temporellement consistante accessible dans GR après franchissement de la séquence S_{n-1}. La classe $E_n = (M_n, D_n)$ accessible à partir de E_{n-1} en franchissant le rendez-vous r_n peut être calculée comme suit :*

- *Le marquage M_n est calculé comme établi dans la Définition 14.*
- *La système D_n est calculé à partir du système D_{n-1} comme suit :*

  1. *Augmenter le système D_{n-1} par les containtes des rendez-vous et de franchissement : $D_{n-1} \wedge CF_{n-1}(r_n)$.*
  2. *Dans ce nouveau système, pour chaque variable non inhibée $\underline{t_{jk}}$, telle que $\underline{t_{jk}} \notin Trans(r_n)$ remplacer alors $\underline{t_{jk}}$ comme la somme d'une nouvelle variable $\underline{t'_{jk}}$ et $\underline{r_n}$.*
  3. *éliminer par substitution dans le système obtenu toutes les variables relatives aux transitions désensibilisées après le tir de r_n.*
  4. *éliminer par substitution dans le système obtenu toutes les variables relatives aux rendez-vous.*
  5. *Dans ce nouveau système, rajouter une variable pour chaque transition nouvellement sensibilisée en lui affectant son intervalle de tir statique. Nous obtenons enfin le système D_n.*

Preuve. La preuve se déduit à partir du *Théorème 3* et de la *Définition 14*. Par définition, la classe E_n regroupe tous les états déja accessibles dans E_{n-1} qui satisfont en plus les contraintes du *Théorème 3 ;* ceci est dénoté par l'extension des contraintes D_{n-1} aux contraintes de franchissement $CF_{n-1}(r_n)$. Ensuite, la progression du temps nécessite la mise à jour des contraintes des transitions non inhibées comme stipulé dans la *Définition 14*. Cela revient à décaler l'origine du temps de la classe par le délai de franchissement variable formalisé par $\underline{r_n}$. Ce qui équivaut à procéder au renommage des variables préscrit dans l'*étape 2*. Ensuite, dans le respect du processus donné en *Définition 14,* on élimine par substitution les variables des transitions désensibilisées et des rendez-vous fortement sensibilisés, pour ne garder que celles relatives aux transitions persistantes. Enfin l'étape cinq compléte le système avec les contraintes des transitions nouvellement sensibilisées. On obtient enfin le système D_n résumant les contraintes des transitions persistantes et nouvellement sensibilisées aprés franchissement du rendez-vous $\underline{r_n}$. ■

Pour un $STPTPN$ dérivant des séquences finies, les résultats précédents permettent d'obtenir un graphe fini représentant un nombre infini d'états. Cependant, de nombreux systèmes temps-réel génèrent des comportements cycliques, et par conséquent offrent des séquences infinies. Nous explorons maintenant comment regrouper ces classes afin de contracter le graphe des classes lorsque le nombre de marquages accessibles est fini. Pour ce faire, nous avons besoin de définir un critère permettant de fusionner deux classes lorsqu'elles offrent les mêmes séquences finies ou infinies. Deux notions d'équivalences sont utilisées dans la littérature l'équivalence basée sur l'égalité des domaines et celle basée sur l'inclusion des domaines. La première garantit la préservation des propriétés linéaires et temps réel du modèle ; la seconde l'atteignabilité des marquages.

Définition 17 ***(Condition d'equivalence).*** *Deux classes temporellement consistantes* $E_{n-1} = (M_{n-1}, D_{n-1})$ *et* $E_{m-1} = (M_{m-1},\ D_{m-1})$ *obtenues après le franchissements respectivement, des séquences* S_{n-1} *et* S_{m-1} *sont :*

- *fortement équivalentes, et on note* $E_{m-1} \approx E_{n-1}$*, si :*
  1. $M_{n-1} = M_{m-1}$
  2. $D_{n-1} = D_{m-1}$
- *faiblement équivalentes, et on note* $E_{m-1} \approxeq E_{n-1}$*, si :*
  1. $M_{n-1} = M_{m-1}$
  2. $D_{n-1} \subseteq D_{m-1}$

Le processus d'énumération présenté précédemment est assez complexe. Déja l'algorithme introduit en *Théorème 4* consomme un temps de complexité exponentiel au nombre de variables présentes dans le système $D_{n-1} \wedge CF_{n-1}(r_n)$. Cette complexité se voit encore plus exacerbée lorsque le processus d'énumératon se prolonge. Malheureusement, les conditions d'équivalences énoncées en *Définition 17* ne permettent pas de garantir un graphe fini dans tous les cas. D'ailleurs la finitude du graphe est discutée ci-après.

5.2.1 Finitude du graphe des classes exact d'un $STPTPN$

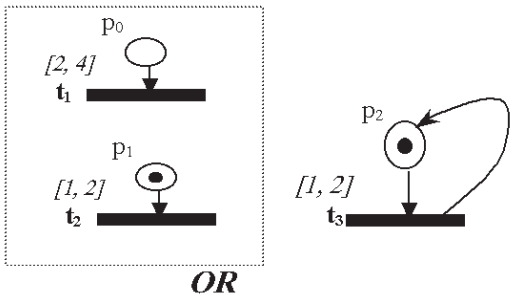

FIGURE 5.1 – Un $STPTPN$ borné générant un graphe infini.

La construction du graphe obtenu en utilisant l'algorithme précédent ne garantit pas nécessairement un graphe fini, et particulièrement lorsque le modèle a un nombre infini de marquages. De plus, le graphe calculé par notre approche pourrait être infini, même si le modèle est borné. D'ailleurs, la finitude du graphe exact associé à un $RdPT$ étendu aux chronomètres (i.e, aux arcs inhibiteurs) est une propriété indécidable pour les réseaux bornés [72]. Nous discutons maintenant de la finitude du graphe lorsque le modèle ne contient pas d'arcs inhibiteurs. Pour ce faire, considérons le $STPTPN$ 1-borné de la *Figure*.5.1.

La classe E_0 est donnée par. $\left(M_0 : p_1, p_2 \to 1, D_0 : \left\{ \begin{array}{l} 1 \preceq \underline{t_2} \preceq 2 \\ 1 \preceq \underline{t_3} \preceq 2 \end{array} \right.\right)$.

De là, le franchissement de $r_2 = (async, \{t_3\})$ mène vers la classe

$$E_1 = \left(M_1 : p_1, p_2 \to 1,\ D_1 : \left\{ \begin{array}{l} 0 \preceq \underline{t_2} \preceq 1 \\ 1 \preceq \underline{t_3} \preceq 2 \end{array} \right.\right)$$

Nous remarquons que $M_0 = M_1$ mais $D_0 \neq D_1$, en clair les classes E_0 et E_1 sont non *fortement équivalentes* selon le $Définition.17$, alors qu'elles offrent les mêmes séquences de franchissement dans le future, puisqu'il n y a aucune possibilité de tirer la transition t_1 selon le rendez-vous r_1. Le processus d'énumération produit malheureusement un graphe infini, alors que le modèle est borné. Ce constat ne s'applique pas pour les $RdPT$ dont le graphe des classes est prouvé fini si le modèle est borné [32]. Par conséquent, la définition d'une condition suffisante garantissant un graphe fini est à proposer.

5.2.2 Exemple

Considérons le $STPTPN$ décrit dans la $Figure.5.2$. Ce réseau permet de décrire un comportement complexe induit par la sémantique des mécanismes des chronomètres et de synchronisation. A cet effet, nous définissons un certain nombre de rendez-vous ; la transition t_6 qui n'apparait dans aucun de ces derniers évoluera quant à elle de manière asynchrone. De plus, un arc inhibiteur connecté à la transition t_1 est spécifié permettant de suspendre le chronomètre de t_1 lorsque p_5 est marquée. A partir du maquage initial M_0 : $\{p_1, p_2, p_3, p_4, p_5\} \rightarrow 1$; $\{p_6, p_7\} \rightarrow 0$, nous déduisons les ensembles des transitions sensibilisées, inhibées et fortement sensibilisées au point (0) : $Te_0 = \{t_1, t_2, t_4, t_5\}$, $Ti_0 = \{t_1\}$ et $Ts_0 = \{t_2, t_4, t_5\}$. Par ailleurs, cela nous permet de calculer l'ensemble des rendez-vous fortement sensibilisés : $Senable_0 = \{r_1, r_4, r_5\}$.

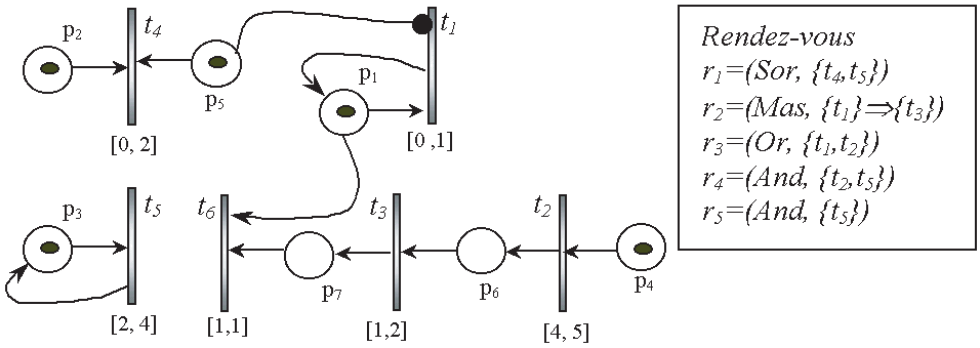

FIGURE 5.2 – Exemple d'un $STPTPN$

La classe initiale est formulée par $E_0 = (M_0, D_0)$ où :

$$D_0 : \begin{cases} 0 \preceq \underline{t_1} \preceq 1, \\ 4 \preceq \underline{t_2} \preceq 5, \\ 0 \preceq \underline{t_4} \preceq 2, \\ 2 \preceq \underline{t_5} \preceq 4, \end{cases}$$

A partir de la classe E_0; seul le rendez-vous r_1 peut être franchi, puisque le système $D_0 \wedge CF_0(r_1)$admet la solution $\underline{r_1}$= 2, $\underline{r_4}$= 4, $\underline{r_5}$= 4. Le calcul de la classe accessible $E_1 = (M_1, D_1)$ accessible après le tir de r_1 se fait en appliquant les étapes du *Théorème 4* comme suit : M_1 : $\{p_1, p_3, p_4\} \rightarrow 1$; Ensuite, nous étendons D_0 aux contraintes de franchissement de r_1 :

$$D_0 \wedge CF_0(r_1) : \left\{ \begin{array}{llll} 0 \preceq \underline{t_1} \preceq 1, & \left\{ \begin{array}{ll} \underline{t_4} \preceq \underline{r_1} & \vee \\ \underline{t_5} \preceq \underline{r_1} & \end{array} \right. , & \left\{ \begin{array}{ll} \underline{r_1} \preceq \underline{t_4} & \wedge \\ \underline{r_1} \preceq \underline{t_5} & \end{array} \right. , & \underline{r_1} \preceq \underline{r_4} \\ 4 \preceq \underline{t_2} \preceq 5, & \left\{ \begin{array}{ll} \underline{t_5} \preceq \underline{r_4} & \wedge \\ \underline{t_2} \preceq \underline{r_4} & \end{array} \right. , & \left\{ \begin{array}{ll} \underline{r_4} \preceq \underline{t_5} & \wedge \\ \underline{r_4} \preceq \underline{t_2} & \end{array} \right. , & \underline{r_1} \preceq \underline{r_5} \\ 0 \preceq \underline{t_4} \preceq 2, & \underline{t_5} \preceq \underline{r_5}, & \underline{r_5} \preceq \underline{t_5}, & \\ 2 \preceq \underline{t_5} \preceq 4, & 0 \preceq \underline{r_4} & 0 \preceq \underline{r_1} & 0 \preceq \underline{r_5} \end{array} \right.$$

Nous procédons à la normalisation du système,

$$\left\{ \begin{array}{ll} 0 \preceq \underline{t_1} \preceq 1, & \\ 4 \preceq \underline{t_2} \preceq 5, & \underline{r_1} = \underline{t_4} \\ 0 \preceq \underline{t_4} \preceq 2, & \underline{r_1} \preceq \underline{t_5} \\ 2 \preceq \underline{t_5} \preceq 4, & \underline{t_5} = \underline{t_2} = \underline{r_4} = \underline{r_5} \end{array} \right.$$

Nous remplaçons ensuite dans ce dernier système chaque occurrence de $\underline{t_2}$ par $\underline{t'_2} + \underline{r_1}$:

$$\left\{ \begin{array}{ll} 0 \preceq \underline{t_1} \preceq 1, & \\ 4 \preceq \underline{t'_2} + \underline{r_1} \preceq 5, & \underline{r_1} = \underline{t_4} \\ 0 \preceq \underline{t_4} \preceq 2, & \underline{r_1} \preceq \underline{t_5} \\ 2 \preceq \underline{t_5} \preceq 4, & \underline{t_5} = \underline{t'_2} + \underline{r_1} = \underline{r_4} = \underline{r_5} \end{array} \right.$$

Nous éliminons par substitution les variables relatives aux transitions désensibilisées t_4 et t_5, et des variables rendez-vous.

$$\left\{ \begin{array}{l} 0 \preceq \underline{t_1} \preceq 1 \\ 2 \preceq \underline{t'_2} \preceq 5 \end{array} \right.$$

Nous rajoutons enfin les contraintes des transitions nouvellement sensibilisées, à savoir t_5, et nous obtenons enfin le système D_1 :

$$D_1 : \left\{ \begin{array}{l} 0 \preceq \underline{t_1} \preceq 1 \\ 2 \preceq \underline{t'_2} \preceq 5 \\ 2 \preceq \underline{t_5} \preceq 4 \end{array} \right.$$

Par contre, r_4 respectivement, r_5 est non franchissable car le système $D_0 \wedge CF_0(r_4)$ respectivement, $D_0 \wedge CF_0(r_5)$ n'admet pas de solutions pour le vecteur $\overline{R_0}$. Pour illustrer cela, considérons le système $D_0 \wedge CF_0(r_5)$

$$\left\{ \begin{array}{llll} 0 \preceq \underline{t_1} \preceq 1, & \left\{ \begin{array}{ll} \underline{t_4} \preceq \underline{r_1} & \vee \\ \underline{t_5} \preceq \underline{r_1} & \end{array} \right. , & \left\{ \begin{array}{ll} \underline{r_1} \preceq \underline{t_4} & \wedge \\ \underline{r_1} \preceq \underline{t_5} & \end{array} \right. , & \underline{r_5} \preceq \underline{r_4}, \quad CBI_0(r_5, r_1) \\ 4 \preceq \underline{t_2} \preceq 5, & \left\{ \begin{array}{ll} \underline{t_5} \preceq \underline{r_4} & \wedge \\ \underline{t_2} \preceq \underline{r_4} & \end{array} \right. , & \left\{ \begin{array}{ll} \underline{r_4} \preceq \underline{t_5} & \wedge \\ \underline{r_4} \preceq \underline{t_2} & \end{array} \right. , & \underline{r_5} \preceq \underline{r_1}, \quad CBI_0(r_5, r_4) \\ 0 \preceq \underline{t_4} \preceq 2, & \underline{t_5} \preceq \underline{r_5}, & \underline{r_5} \preceq \underline{t_5}, & \\ 2 \preceq \underline{t_5} \preceq 4, & 0 \preceq \underline{r_4}, & 0 \preceq \underline{r_1}, & 0 \preceq \underline{r_5} \end{array} \right.$$

avec

$$CBI_0(r_5, r_1) : \left\{ \begin{array}{ll} \left\{ \begin{array}{ll} \underline{r_5} \prec \underline{t_4} - 2 & \vee \\ \underline{r_5} \prec 0 & \end{array} \right. & \\ \wedge & \\ \left\{ \begin{array}{ll} \underline{r_5} \prec \underline{t_5} - 2 & \vee \\ \underline{r_5} \prec 0 & \end{array} \right. & \end{array} \right. \qquad CBI_0(r_5, r_4) : \left\{ \begin{array}{ll} \left\{ \begin{array}{ll} \underline{r_5} \prec \underline{t_2} - 1 & \vee \\ \underline{r_5} \prec 0 & \end{array} \right. & \\ \vee & \\ \left\{ \begin{array}{ll} \underline{r_5} \prec \underline{t_5} - 2 & \vee \\ \underline{r_5} \prec 0 & \end{array} \right. & \end{array} \right.$$

Après normalisation du système, nous aboutissons à une contradiction entre les contraintes, $\underline{r_5} \prec \underline{t_2} - 1$ et $\underline{r_5} = \underline{t_2}$

De la même manière, nous calculons la classe $E_{21} = (M_{21}, D_{21})$ obtenue après franchissement du rendez-vous r_4 à partir de la classe E_1 telle que : $M_{21} : \{p_1, p_3, p_6\} \rightarrow 1$; et

$$D_{21} : \begin{cases} 0 \preceq \underline{t_1} \preceq -1 \\ 0 \preceq \underline{t_3} \preceq 2 \\ 2 \preceq \underline{t_5} \preceq 4 \end{cases}$$

Cette dernière classe est *temporellement inconsistante* puisque le rendez-vous r_2 vérifie la *Théorème 2 :* $D_{21} \wedge Sup_{21}(r_2) \wedge (\underline{r_2} \succeq 0)$ n'admet pas de solutions pour la variable $\underline{r_2}$. Par ailleurs, l'exploration du graphe par le franchissement du rendez-vous r_3 à partir de la classe E_1 permet de construire le reste des branches pour aboutir au graphe de la *Figure 5.3*.

$M_1 : \{p_1, p_3, p_4\} \rightarrow 1$;
$M_{22} = M_{32} = M_{21} : \{p_1, p_3, p_6\} \rightarrow 1$;
$M_{31} = M_{42} = M_{43} = M_{53} : \{p_1, p_3, p_7\} \rightarrow 1$;
$M_{41} = M_{54} = M_{51} = M_{52} : \{p_5\} \rightarrow 1$.

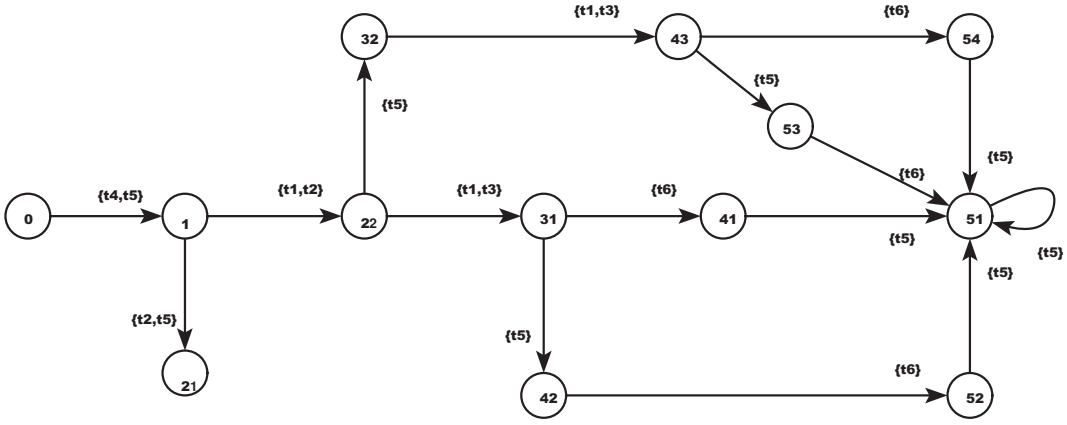

FIGURE 5.3 – Le graphe des classes exact du *STPTPN* de la figure.5.2.

$$D_{22} : \begin{cases} 0 \preceq \underline{t_5} \preceq 4 \\ 0 \preceq \underline{t_1} \preceq 1 \\ 1 \preceq \underline{t_3} \preceq 2 \end{cases} \quad D_{31} : \begin{cases} 0 \preceq \underline{t_5} \preceq 4 \\ 0 \preceq \underline{t_1} \preceq 1 \\ 1 \preceq \underline{t_6} \preceq 1 \end{cases} \quad D_{32} : \begin{cases} 2 \preceq \underline{t_5} \preceq 4, \quad 0 \preceq \underline{t_3} - \underline{t_1} \preceq 2 \\ 0 \preceq \underline{t_1} \preceq 1 \\ 0 \preceq \underline{t_3} \preceq 2 \end{cases}$$

$$D_{53} : \begin{cases} 2 \preceq \underline{t_5} \preceq 4 \\ \underline{t_1} = 0 \\ \underline{t_6} = 0 \end{cases} \quad D_{42} : \begin{cases} 2 \preceq \underline{t_5} \preceq 4 \\ 0 \preceq \underline{t_1} \preceq 1 \\ 0 \preceq \underline{t_6} \preceq 1 \end{cases} \quad D_{43} : \begin{cases} 1 \preceq \underline{t_5} \preceq 4 \\ 0 \preceq \underline{t_1} \preceq 1 \\ 1 \preceq \underline{t_6} \preceq 1 \end{cases}$$

$$D_{41} : \begin{cases} 0 \preceq \underline{t_5} \preceq 3 \end{cases} \quad D_{51} : \begin{cases} 2 \preceq \underline{t_5} \preceq 4 \end{cases} \quad D_{52} : \begin{cases} 1 \preceq \underline{t_5} \preceq 4 \end{cases} \quad D_{54} : \begin{cases} 1 \preceq \underline{t_5} \preceq 3 \end{cases}$$

5.2.3 Discussion

Dans le graphe de la *Figure 5.3*, nous remarquons que toutes les classes énumérées sont caractérisées par des domaines de tirs D réduisibles à leurs sous systèmes *DBM* $\widetilde{D}$; le système de

forme polyédrique $\widehat{D}$ étant redondant pour ces classes. Par conséquent, la sur-approximation du système D par sa partie DBM aurait été appropriée pour cet exemple puisque l'approche permet de générer le même graphe avec un coût présumé polynômial pour le calcul d'une classe, alors que l'approche basée sur les polyèdres est de complexité exponentielle, nécessitant des structures lourdes pour son implémentation. De plus, le calcul du graphe exact peut être contre productif si la propriété à analyser peut être vérifiée sur des abstractions de l'espace d'état certes moins riches mais néanmoins moins couteuses.

L'approche de sur-approximation classique consiste donc à éliminer systématiquement les contraintes du sous-système $\widehat{D}$ dès qu'elles apparaîssent et de ne laisser que celles formant le sous-système $\widetilde{D}$. Il s'en suit que les tests de consistance et de franchissement ainsi que le processus de normalisation se déterminent plus efficacement par l'utilisation d'algorithmes adaptées à cette forme. Toutefois, la complexité de calcul et de canonisation d'une classe réduite à sa forme DBM reste importante en utilisant l'algorithme du plus court chemin de *Floyd/Warshall.* Au fait, elle dépend du nombre de variables présentes dans le système, estimé approximativement à $o((m+l)^3)$ où l et m sont respectivement le nombre de transitions sensibilisées et le nombre de rendez-vous fortement sensibilisés pour la classe. Pour réduire cette complexité, nous proposons dans la section suivante une technique alternative implémentant la sur-approximation DBM du graphe des classes d'un $STPTPN$ en utilisant des structures matricielles plus adaptées, et un algorithme plus efficace pour le calcul d'une classes dont la complexité est d'ordre deux estimé à $o(l^2+l\times m+m)$.

5.3 Construction du graphe des classes sur-approximé d'un *STPTPN*

Nous montrons dans cette section comment construire le graphe sur-approximé d'un $STPTPN$. La construction de ce graphe est basée sur le principe de relâchement des contraintes non DBM de toute classe $E=(M,D)$ accessible dans le graphe exact GR. Ceci produit une classe $\widetilde{E}=(M,\widetilde{D})$ en forme DBM l'enveloppant [6], pouvant néanmoins générer de nouvelles classes (ensemble d'étas), non accessibles dans GR. La mise en oeuvre pratique de cette technique exploitant des *IHTPN [83]* a été réalisée et les résultats des simultations et comparaisons sont présentés en *Annexe B.*

Ci-après, nous introduisons formellement le graphe des classes sur-approximée.

Définition 18 *Le graphe des classes sur-approximé d'un STPTPN noté $\widetilde{GR}$ est défini par le tuple $(\widetilde{CE},\widetilde{E}_0,L,\rightsquigarrow)$ où :*

- $\widetilde{CE}$ *: est l'ensemble des classes DBM accessibles dans $\widetilde{GR}$;*
- $\widetilde{E}_0$ *: est une classe de $\widetilde{CE}$ dite classe initiale ;*
- L *: l'ensemble des labels ;*
- $\rightsquigarrow$*: est une relation entre classes définie sur $\widetilde{CE}\times L\times\widetilde{CE}$. On note $(\widetilde{E},l,\widetilde{E}')\in\rightsquigarrow$ par $\widetilde{E}\overset{l}{\rightsquigarrow}\widetilde{E}'$.*

La construction du graphe $\widetilde{GR}$ nécessite l'encodage de toute classe accessible $\widetilde{E}_n$ sous sa forme DBM. Une classe dans $\widetilde{E}$ est donnée par la représentation suivante :

Définition 19 ***(Matrices des distances des bornes des transitions).*** *Soit $\widetilde{E}_n=(M_n,\widetilde{D}_n)$ une classe accessible dans $\widetilde{GR}$ après le franchissement de la séquence S_n. Nous représentons le*

6. Le domaine représenté par $\widetilde{D}$ inclut celui de D.

système $\widetilde{D}_n$ par une matrice sous forme DBM, et définie comme suit :

$$\widetilde{D}_n : \{Te_n \cup \{\bullet\}\}^2 \longrightarrow Q \cup \{\infty\} \ : \forall (t_i, t_j) \in Te_n^2$$
$$\widetilde{D}_n[\bullet, t_i] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \quad \{y_n(t_i)\}$$
$$\widetilde{D}_n[t_i, \bullet] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \quad \{-x_n(t_i)\}$$
$$\widetilde{D}_n[t_i, t_j] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \quad \{y_n(t_j) - x_n(t_i)\}$$
$$\widetilde{D}_n[\bullet, \bullet] := 0$$

Les éléments $\widetilde{D}_n[\bullet, t_i], -\widetilde{D}_n[t_i, \bullet]$ et $\widetilde{D}_n[t_i, t_j]$ représentent respectivement, les coefficients $c_{\bullet i}, c_{i\bullet}$, et c_{ij} du sous-système linéaire $\widetilde{D}_n$ relative à la classe $\widetilde{E}_n$. L'élément $\{\bullet\}$ dénote l'origine temporelle de la classe. La matrice $\widetilde{D}_0$ relative à la classe initiale $\widetilde{E}_0$ est définie comme suit :

$$\forall t_i \in Te_0, \quad \widetilde{D}_0[\bullet, t_i] := LFT(t_i), \quad \widetilde{D}_0[t_i, \bullet] := -EFT(t_i)$$
$$\forall (t_i, t_j) \in (Te_0)^2 \qquad \widetilde{D}_0[t_i, t_j] := LFT(t_j) - EFT(t_i)$$

$\widetilde{D}_n[\bullet, t_i], \widetilde{D}_n[t_i, \bullet]$ et $\widetilde{D}_n[t_i, t_j]$ sont respectivement, le délai résiduel maximum de la transition t_i, la valeur opposée du délai résiduel minimum de la transition t_i, et la distance maximale entre les délais résiduels de t_i vers t_j. Il est à noter que la valeur de $\widetilde{D}_n[t_i, \bullet]$ est toujours négative ou nulle. Concernant la valeur de $\widetilde{D}_n[\bullet, t_i]$, elle pourrait être négative dénotant ainsi la violation des contraintes de temps de la transition t_i relativement à l'origine du temps de la classe, i.e., Il n y a pas de solutions satisfaisant n'importe quel système FT_n (associé à un état accessible dans $\widetilde{E}_n$), pour que la séquence S_n puisse ête franchie sans violer les contraintes de temps de la transition t_i. D'autre part, $\widetilde{D}_n[t_i, t_j]$ peut être négative, dénotant qu'il n'existe pas de solution satisfaisant tout système FT_n (associé à un état accessible après le franchissement de S_n), tel que le délai résiduel de t_j soit supérieur à celui de t_i. Au fait, la matrice $\widetilde{D}_n$ représente ici le plus large domaine de valeurs contenant toutes les solutions des systèmes FT_n relatifs aux états accessibles après franchissement de la séquence S_n mais aussi un sous ensemble de solutions [7] qui ne satisfait pas FT_n. Autrement, si E_n est une classe accessible dans le graphe exact, alors sa sur-approximation $\widetilde{E}_n$ est une classe qui englobe aussi bien les états de E_n que ceux qui ne vérifie que le système $\widehat{D}_n$. Il en résulte que la classe $\widetilde{E}_n$ peut inférer des séquences de franchissement non accessibles à partir de E_n. Ces séquences peuvent induire des nouvelles classes atteignables dans $\widetilde{GR}$ mais pas dans GR. Cependant, la matrice $\widetilde{D}_n$ ne permet pas de formuler une *condition approximée* efficace pour le franchissement d'un rendez-vous. Nous avons besoin de calculer de cela un autre système équivalent capturant l'espace des contraintes relatif aux rendez-vous fortement sensibilisés, noté DR_n. Comme pour $\widetilde{D}_n$, le système DR_n doit être encodé en forme DBM afin qu'il puisse être exploité efficacement dans la construction du graphe $\widetilde{GR}$. La définition suivante introduit formellement le système DR_n dans sa forme DBM. Nous montrons par la suite comment le calculer directement à partir de la représentation matricielle de $\widetilde{D}_n$.

Définition 20 ***(Matrice des distances des bornes des rendez-vous).*** *Soit $\widetilde{E}_n = (M_n, \widetilde{D}_n)$ une classe accessible dans $\widetilde{GR}$ après le franchissement de la séquence S_n. Les écarts de franchissements entre rendez-vous fortement sensibilisés pour M_n peuvent être représentées par le système DR_n présenté sous forme d'une matrice des distances des bornes : $DR_n : \{Senable_n \cup \{\bullet\}\}^2 \longrightarrow$*

7. Ces solutions sont celles qui satisfont $\widetilde{D}_n$ mais pas $\widehat{D}_n$.

$Q \cup \{\infty\}$: $\forall (r, r') \in (Senable_n)^2 \wedge (r \neq r')$

$$DR_n[\bullet, r] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{Up_n(r)\} \qquad DR_n[r, \bullet] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{-Low_n(r)\}$$

$$DR_n[r, r'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \begin{cases} MIN \begin{cases} Low_n(r') - Low_n(r) \\ Up_n(r') - Low_n(r) \end{cases} & si\ r' \in \Phi_n(r) \\ Up_n(r') - Low_n(r) & sinon \end{cases}$$

$$DR_n[r, r] := 0; \qquad DR_n[\bullet, \bullet] = 0$$

La matrice DR_0 est définie comme suit :

$$\forall r \in Senable_0 \quad DR_0[r, r] := 0, \qquad DR_0[\bullet, r] := Up_0(r), \quad DR_0[r, \bullet] := -Low_0(r)$$

$$\forall (r, r') \in (Senable_0)^2 \qquad DR_0[r, r'] := \begin{cases} MIN \begin{cases} Low_0(r') - Low_0(r) \\ Up_0(r') - Low_0(r) \end{cases} & \text{si } r' \in \Phi_0(r) \\ Up_0(r') - Low_0(r) & \text{sinon} \end{cases}$$

$DR_n[\bullet, r]$ et $DR_n[r, \bullet]$ sont respectivement le délai résiduel maximum du rendez-vous r, la valeur opposée du délai résiduel minimum du rendez-vous r. $DR_n[r, r']$ dénote si r est non strictement inclusif dans r', l'écart maximal entre la borne inférieure de r et la borne supérieure de r', sinon il dénote l'écart maximal entre les bornes inférieures de r et r'. Toujours est-il que la valeur de $DR_n[r, \bullet]$ est toujours négative ou nulle. Par contre, la valeur de $DR_n[\bullet, r]$ pourrait être négative, dénotant ainsi la violation des contraintes de temps du rendez-vous r. D'autre part, $DR_n[r, r']$ peut être négative, dénotant que les contraintes de franchissement de r sont non satisfaites par r'. Le calcul de la matrice DR_n est obtenu à partir de celle de $\widetilde{D}_n$ en utilisant la proposition suivante.

Proposition 1 *Soit $\widetilde{E}_n = (M_n, \widetilde{D}_n)$ une classe accessible dans $\widetilde{GR}$ après le franchissement de la séquence S_n. La matrice DR_n associée à la classe $\widetilde{E}_n$ peut être calculée comme suit :*
$\forall r \in Senable_n$

$$DR_n[\bullet, r] := \begin{cases} \underset{\forall t \in Trans(r)}{MAX} \left\{\widetilde{D}_n[\bullet, t]\right\} & Si\ typ(r) \in \{Or, Wand, Wmas\} \\ \underset{\forall t \in Trans(r)}{MIN} \left\{\widetilde{D}_n[\bullet, t]\right\} & Si\ typ(r) \in \{Sor, And, Smas, Async\} \\ \widetilde{D}_n[\bullet, t_m] & Si\ typ(r) \in \{Mas, Omas, Amas\} \wedge (t_m = MA(r)) \end{cases}$$

$$DR_n[r, \bullet] := \begin{cases} \underset{\forall t \in Trans(r)}{MIN} \left\{\widetilde{D}_n[t, \bullet]\right\} & Si\ typ(r) \in \{And, Wand, Amas, Async\} \\ \underset{\forall t \in Trans(r)}{MAX} \left\{\widetilde{D}_n[t, \bullet]\right\} & Si\ typ(r) \in \{Or, Sor, Omas\} \\ \widetilde{D}_n[t_m, \bullet] & Si\ typ(r) \in \{Mas, Smas, Wmas\} \wedge (t_m = MA(r)) \end{cases}$$

$\forall (r, r') \in (Senable_n)^2 \wedge (r \neq r')$

$$DR_n[r, r'] := \begin{cases} \underset{\forall t \in Trans(r)}{MIN} \{\beta_n[t, r'](r)\} & If\ typ(r) \in \{And, Wand, Amas, Async\} \\ \underset{\forall t \in Trans(r)}{MAX} \{\beta_n[t, r'](r)\} & If\ typ(r) \in \{Or, Sor, Omas\} \\ \beta_n[t, r'](r) & If\ typ(r) \in \{Mas, Smas, Wmas\} \wedge\ (t = MA(r)) \end{cases}$$

$\forall (r, r') \in Senable_n^2 \wedge \forall t \in Te_n - Ti_n$

$\beta_n[t, r'](r) :=$

$\begin{cases} \underset{\forall t' \in Trans(r')}{MIN} \left\{ \widetilde{D}_n[t, t'] \right\} & Si\ typ(r') \in \{Sor, Smas, Async, And\} \\ \underset{\forall t' \in Trans(r')}{MAX} \left\{ \widetilde{D}_n[t, t'] \right\} & Si\ typ(r') \in \{Wand, Or, Wmas\} \\ \widetilde{D}_n[t, t_m] & Si\ typ(r') \in \{Amas, Omas, Mas\} \wedge \\ & t_m = MA(r') \end{cases}$	$si\ r' \notin \Phi_n(r)$
$\begin{cases} MIN \begin{cases} MAX \begin{cases} \underset{\forall t' \in Trans(r')}{MAX} \left\{ \widetilde{D}_n[t, t'] + \alpha(t') \right\} \\ \widetilde{D}_n[t, \bullet] \end{cases} & Si\ typ(r') \in \begin{cases} Async \\ And \end{cases} \\ \underset{\forall t' \in Trans(r')}{MIN} \left\{ \widetilde{D}_n[t, t'] \right\} \end{cases} \\ MIN \begin{cases} MAX \begin{cases} \underset{\forall t' \in Trans(r')}{MAX} \left\{ \widetilde{D}_n[t, t'] + \alpha(t') \right\} \\ \widetilde{D}_n[t, \bullet] \end{cases} & Si\ typ(r')=Wand \\ \underset{\forall t' \in Trans(r')}{MAX} \left\{ \widetilde{D}_n[t, t'] \right\} \end{cases} \\ MIN \begin{cases} MAX \begin{cases} \underset{\forall t' \in Trans(r')}{MAX} \left\{ \widetilde{D}_n[t, t'] + \alpha(t') \right\} \\ \widetilde{D}_n[t, \bullet] \end{cases} & Si\ typ(r')=Amas \\ \widetilde{D}_n[t, t_m] \end{cases} \\ MIN \begin{cases} MAX \begin{cases} \underset{\forall t' \in Trans(r')}{MIN} \left\{ \widetilde{D}_n[t, t'] + \alpha(t') \right\} \\ \widetilde{D}_n[t, \bullet] \end{cases} & Si\ typ(r')=Or \\ \underset{\forall t' \in Trans(r')}{MAX} \left\{ \widetilde{D}_n[t, t'] \right\} \end{cases} \\ MIN \begin{cases} MAX \begin{cases} \underset{\forall t' \in Trans(r')}{MIN} \left\{ \widetilde{D}_n[t, t'] + \alpha(t') \right\} \\ \widetilde{D}_n[t, \bullet] \end{cases} & Si\ typ(r')=Sor \\ \underset{\forall t' \in Trans(r')}{MIN} \left\{ \widetilde{D}_n[t, t'] \right\} \end{cases} \\ MIN \begin{cases} MAX \begin{cases} \underset{\forall t' \in Trans(r')}{MIN} \left\{ \widetilde{D}_n[t, t'] + \alpha(t') \right\} \\ \widetilde{D}_n[t, \bullet] \end{cases} & Si\ typ(r')=Omas \\ \widetilde{D}_n[t, t_m] \end{cases} \\ MIN \begin{cases} MAX \begin{cases} \widetilde{D}_n[t, t_m] + \alpha(t_m) \\ \widetilde{D}_n[t, \bullet] \end{cases} & Si\ typ(r')=Mas \\ \widetilde{D}_n[t, t_m] \end{cases} \\ MIN \begin{cases} MAX \begin{cases} \widetilde{D}_n[t, t_m] + \alpha(t_m) \\ \widetilde{D}_n[t, \bullet] \end{cases} & Si\ typ(r')=Smas \\ \underset{\forall t' \in Trans(r')}{MIN} \left\{ \widetilde{D}_n[t, t'] \right\} \end{cases} \\ MIN \begin{cases} MAX \begin{cases} \widetilde{D}_n[t, t_m] + \alpha(t_m) \\ \widetilde{D}_n[t, \bullet] \end{cases} & Si\ typ(r')=Wmas \\ \underset{\forall t' \in Trans(r')}{MAX} \left\{ \widetilde{D}_n[t, t'] \right\} \end{cases} \end{cases}$	$si\ r' \in \Phi_n(r)$ $avec$ $t_m = MA(r')$

Preuve. La preuve est donnée en *Annexe A* ■

La valeur de $\beta_n[t, r'](r)$ peut exprimer dans le contexte où t est une transition à tirer en franchissant r, l'écart maximal de la borne inférieure de t à la borne supérieure de r' si r est strictement non inclusif dans r'; Autrement, elle dénote l'écart maximal de la borne inférieure de t à la borne inférieure de r'. Pour chaque rendez-vous r, les valeurs $\beta_n[t, r'](r)$ obtenues en variant [8] t et r' peuvent être représentées par une matrice où les lignes dénotent les transitions sensibilisées et non inhibées et les colonnes les rendez-vous fortement sensibilisées. Toutefois, nous pouvons remarquer que pour tous les rendez-vous r non inclus strictement dans r', on obtient la même matrice, et de même pour tous les rendez-vous r inclus strictement dans r'. Par conséquent, nous aurons seulement besoin de calculer deux matrices, la première matrice notée $\overrightarrow{\beta_n}$, calculant les valeurs de $\beta_n[t, r'](r)$ dans le contexte où r est non strictement inclusif dans r' et la seconde matrice $\overleftarrow{\beta_n}$ dans le cas contraire. De là, l'algorithme doit procéder initialement au calcul des deux matrices $\overrightarrow{\beta_n}$ et $\overleftarrow{\beta_n}$, avant de procéder à celui de la matrice DR_n.

Remarque : Il est à noter que les deux matrices $\overrightarrow{\beta_n}$ et $\overleftarrow{\beta_n}$ seront exploitées (comme il est montré dans l'énoncé de la *Proposition*.2), dans le calcul des classes successeurs de $\widetilde{E}_n$. Par conséquent, nous devons sauvegarder ces matrices en mémoire jusqu'à la fin du calcul de tous les successeurs de E_n ; Autrement, nous serons obligés de les recalculer à chaque énumération d'une classe successeur. Donc, ce facteur devrait être pris en compte lors de l'élaboration de la stratégie d'exploration lors du calcul du graphe des classes sur-approximé.

$\widetilde{D}_0$	$\bullet$	t_1	t_2	t_4	t_5
$\bullet$	0	1	5	2	4
t_1	0	1	5	2	4
t_2	−4	−3	1	−2	0
t_4	0	1	5	2	4
t_5	−2	−1	3	0	2

$\overrightarrow{\beta_0}$	r_1	r_4	r_5
t_2	−2	0	0
t_4	2	4	4
t_5	0	2	2

$\overleftarrow{\beta_0}$	r_1	r_4	r_5
t_2	0	−2	−2
t_4	0	−2	2
t_5	−2	0	0

DR_0	$\bullet$	r_1	r_4	r_5
$\bullet$	0	2	4	4
r_1	0	0	4	4
r_4	−4	−2	0	0
r_5	−2	−2	0	0

TABLE 5.1 – Les matrices $\widetilde{D}$, $\overrightarrow{\beta}$, $\overleftarrow{\beta}$, DR au point (0).

Les matrices $\widetilde{D}_0, \overrightarrow{\beta_0}, \overleftarrow{\beta_0}$ et DR_0 associées à la classe initiale du $STPTPN$ de la *Figure 5.2* sont présentées en *Table*.5.1. Pour obtenir ces matrices nous calculons d'abord la matrice $\widetilde{D}_0$; ensuite à partir de cette dernière nous déterminons les matrices $\overrightarrow{\beta_0}$ et $\overleftarrow{\beta_0}$. Finalement, la matrice DR_0 est calculée à partir de ces deux dernières. Nous discutons maintenant d'une condition suffisante pour la *consistance temporelle* de chaque classe accessible $\widetilde{E}_n$. Pour ce faire, le théorème suivant montre comment exploiter la matrice DR_n pour déterminer si une classe accessible est *temporellement consistante* ou non.

Théorème 5 *Soit* $\widetilde{E}_n = (M_n, \widetilde{D}_n)$ *une classe accessible dans* $\widetilde{GR}$ *après le franchissement de la séquence* S_n *et* DR_n *la matrice associée.*
$\widetilde{E}_n \not\models^t Ssi\ \exists r \in Senable_n \quad DR_n[\bullet, r] \prec 0.$

8. telle que t appartient à l'ensemble des transitions sensibilisées et non inhibées.

Preuve. La preuve est donnée en *Annexe A.* ■

Soit $\widetilde{E}_n$ la sur-approximation de la classe E_n. Le fait que $\widetilde{E}_n$ soit *temporellement consistante* ne garantit pas que E_n l'est. Au fait, nous garantissons que si $\widetilde{E}_n$ est *temporellement inconsistante*, alors E_n l'est aussi. Ceci est du au fait que $\widetilde{E}_n$ peut contenir au moins un état supplémentaire qui soit *temporellement consistant* pendant que les états de E_n sont tous *temporellement inconsistants*.

Nous cherchons maintenant à exploiter la matrice DR_n dans le test de franchissement. Pour cet effet, le théorème suivant établit une condition approximée, pour franchir un rendez-vous au point $(n-1)$. Au fait, cette condition est nécessaire pour la construction de GR et suffisante pour celle de $\widetilde{GR}$.

Théorème 6 *Soit* $\widetilde{E}_{n-1} = (M_{n-1}, \widetilde{D}_{n-1})$ *une classe temporellement consistante accessible dans* $\widetilde{GR}$ *après le franchissement de la séquence* S_{n-1}*, et* DR_{n-1} *est la matrice associée à* $\widetilde{E}_{n-1}$*. Un rendez-vous* r_n *peut pêtre franchi à partir de* $\widetilde{E}_{n-1}$, Ssi *:*

(i) $r_n \in Senable_{n-1}$ *;*

(ii) $\forall r \in Senable_{n-1} \begin{cases} DR_{n-1}[r_n, r] \succ 0 & \text{Si } r \in \Phi_{n-1}(r_n) \\ DR_{n-1}[r_n, r] \succeq 0 & \textit{sinon} \end{cases}$

Preuve. La preuve est donnée en *Annexe A* ■

Si $\widetilde{E}_{n-1}$ est la sur-approximation de la classe E_{n-1}, alors tout rendez-vous franchissable r_n à partir de E_n vérifie le *Théorème 3*, et donc est franchissable à partir de $\widetilde{E}_{n-1}$. Cependant, un rendez-vous non franchissable à partir de E_{n-1} peut vérifier le *Théorème 3* et donc pourrait être franchissable [9] à partir de $\widetilde{E}_{n-1}$. Au fait, comme la classe $\widetilde{E}_{n-1}$ inclut les états de $\widetilde{E}_{n-1}$, il se peut qu'il y ait au moins un état e_{n-1} appartenant à $\widetilde{E}_{n-1}$ et non à E_{n-1} tel que e_{n-1} franchit r_n.

En appliquant le théorème précédent sur le $STPTPN$ de la $Figure.5.2$; nous constatons qu'initialement parmi les rendez-vous fortement sensibilisés seul r_5 est non franchissable puisque $DR_0[r_5, r_1] = -2 \prec 0$. Plus clairement, r_5 est victime de la priorité en temps réel puisque il ne peut se réaliser en dehors des instants de franchissement de r_1 (qui l'inclut strictement).

Par ailleurs, il reste à proposer une technique calculant les paramètres de chaque classe nouvellement accessible dans $\widetilde{GR}$. Plus précisément, nous pensons à la définition d'un processus de calcul des éléments de la matrice $\widetilde{D}_n$. Toutefois, un calcul efficace des classes accessible passe par la sur-approximation des coefficients de la matrice $\widetilde{D}_n$. Nous présentons ci-après ce processus de calcul récursif permettant de calculer la surapproximation de la matrice $\widetilde{D}_n$ à partir de la matrice $\widetilde{D}_{n-1}$.

Proposition 2 *Soit* $\widetilde{E}_{n-1} = (M_{n-1}, \widetilde{D}_{n-1})$ *une classe temporellement consistante accessible dans* $\widetilde{GR}$ *après franchissement de la séquence* S_{n-1}*. La classe* $\widetilde{E}_n = (M_n, \widetilde{D}_n)$ *accessible à partir de* $\widetilde{E}_{n-1}$ *en franchissant le rendez-vous* r_n *peut être calculée par sur-approximation comme suit :*

1. *Le marquage* M_n *est calculé comme montré dans la* $Définition\ 14$*.*
2. *La matrice* $\widetilde{D}_n$ *est calculée à partir de la matrice* $\widetilde{D}_{n-1}$ *en effectuant une sur-approximation, comme suit :*

   $\forall t \in Te_n$

   Si t est persistante

9. Inversement, si r_n est non franchissable à partir de $\widetilde{E}_{n-1}$ alors il l'est aussi à partir de E_{n-1}.

Si $t \in Ti_{n-1}$ *(t est inhibée au point* $n-1$*)*

$$\widetilde{D}_n[\bullet,t] \preceq \widetilde{D}_{n-1}[\bullet,t]$$
$$\widetilde{D}_n[t,\bullet] \preceq \widetilde{D}_{n-1}[t,\bullet]$$

Si $t \notin Ti_{n-1}$ *(t est non inhibée au point* $n-1$*)*

$$\widetilde{D}_n[\bullet,t] :=$$

$$\begin{cases} \underset{\forall t' \in Trans(r_n)}{MIN} \left\{\widetilde{D}_{n-1}[t',t]\right\} & Si\ typ(r_n) \in \{And, Wand, Amas, Async\} \\ \underset{\forall t' \in Trans(r_n)}{MAX} \left\{\widetilde{D}_{n-1}[t',t]\right\} & Si\ typ(r_n) \in \{Or, Sor, Omas\} \\ \widetilde{D}_{n-1}[t_m,t] & Si\ typ(r_n) \in \{Mas, Smas, Wmas\} \wedge\ t_m = MA(r_n) \end{cases}$$

$$\widetilde{D}_n[t,\bullet] := MAX \begin{pmatrix} MIN\left(0, \quad \underset{\forall r \in Senable_{n-1}}{MIN} \left\{\beta_{n-1}[t,r](r_n)\right\}\right) \\ MIN\left(0, \quad -\widetilde{D}_n[\bullet,t] - \alpha(t)\right) \end{pmatrix}$$

Si t *est nouvellement sensibilisée*

$$\widetilde{D}_n[\bullet,t] := LFT(t)$$
$$\widetilde{D}_n[t,\bullet] := -EFT(t)$$

$\forall (t,t') \in (Te_n)^2 \wedge (t \neq t')$

Si t *ou* t'*sont nouvellement sensibilisées.*

$$\widetilde{D}_n[t,t'] := \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet]$$

Si t *et* t' *sont persistantes*

Si $(t,t') \notin (Ti_{n-1})^2$ *(*t *et* t' *ne sont pas inhibées au point* $n-1$*)*

$$\widetilde{D}_n[t,t'] := MIN(\widetilde{D}_{n-1}[t,t'], \quad \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet])$$

Si $(t,t') \in (Ti_{n-1})^2$ *(*t *et* t' *sont inhibées au point* $n-1$*)*

$$\widetilde{D}_n[t,t'] := MIN(\widetilde{D}_{n-1}[t,t'], \quad \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet])$$

Si $(t \in Ti_{n-1}) \wedge (t' \notin Ti_{n-1})$ *(seule* t *est inhibée au point* $n-1$*)*

$$\widetilde{D}_n[t,t'] \quad \underline{\prec} \quad MIN\left(\widetilde{D}_{n-1}[t,t'] + DR_{n-1}[r_n,\bullet], \quad \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet]\right)$$

Si $(t \notin Ti_{n-1}) \wedge (t' \in Ti_{n-1})$ *(seule* t' *est inhibée au point* $n-1$*)*

$$\widetilde{D}_n[t,t'] \quad \underline{\prec} \quad MIN\left(\widetilde{D}_{n-1}[t,t'] + DS_n[n-1,n], \quad \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet]\right)$$

Preuve. La preuve est donnée en *Annexe A.* ■

Remarque : Le paramètre $DS_n[n-1,n]$ dénote le temps maximum écoulé entre les points $n-1$ et n. La formalisation de ce paramètre est donnée en *Chapitre 6*, et ses formules de calcul détaillées dans la *Proposition.3*.

Les formules précédentes permettent de calculer une surapproximation de la matrice $\widetilde{D}_n$. A cet effet, le signe " $\preceq$ " désigne que la formule utilisée calcule une valeur supérieure à celle du

coefficient, alors que le signe ":= " dénote que la formule calcule la valeur exacte. D'ailleurs, en analysant ces formules, nous remarquons que la sur-approximation concerne exclusivement les écarts impliquant les transitions anciennement inhibées. Autrement dit, la matrice $\widetilde{D}_n$ calculée est exacte lorsque le modèle ne contient pas d'ars inhibiteurs. De plus, dans certains réseaux contenant des arcs inhibiteurs, cette sur-approximation n'a pas d'incidence sur le calcul du graphe, mais dans d'autres cas (voir § *II.4.2*) il se pourrait qu'un rendez-vous non franchissable dans le modèle le devient suite à cette à surapproximation. Au fait, le système polyédrique $\widehat{D}_n$ n'est généré que lorsque le processus d'énumération rend persistant certaines transitions inhibées. L'élimination de ce sous système lorsqu'il n'est pas redondant implique que le calcul du système réduit $\widetilde{D}_{n+1}$ à partir de $\widetilde{D}_n$ ne correspond pas à celui que l'on obtiendrai à partir du système global $\widehat{D}_n \wedge \widetilde{D}_n$. Toutefois, cela reste le prix à payer pour maintenir la complexité de calcul d'une classe à un degré polynômial.

Théorème 7 ***(Condition d'equivalence).*** *Deux classes temporellement consistantes* $\widetilde{E}_{n-1} = (M_{n-1}, \widetilde{D}_{n-1})$ *et* $\widetilde{E}_{m-1} = (M_{m-1}, \widetilde{D}_{m-1})$ *obtenues après le franchissements respectivement, des séquences* S_{n-1} *et* S_{m-1} *sont fortement équivalentes, si :*

(i) $M_{n-1} = M_{m-1}$

(ii) $\widetilde{D}_{n-1} = \widetilde{D}_{m-1}$

5.3.1 Algorithme d'énumération du graphe sur-approximé

Pour construire le graphe sur-approximé d'un $STPTPN$, nous appliquons le test de franchissement du *Théorème.*6 sur la classe initiale $\widetilde{E}_0$, ensuite sur toutes les classes accessibles. Les conditions suffisantes exprimées dans le *Théorème.*7 permettent de regrouper dans un même noeud toutes les classes consistantes équivalentes. Par ailleurs, les classes inconsistantes sont regroupées dans un même noeud ; décrivant l'état du $STPTPN$ où les contraintes de temps sont violées. Pour chaque classe à explorer, l'algorithme doit procéder d'abord au calcul des matrices, $\overleftarrow{\beta_{n-1}}$ et $\overrightarrow{\beta_{n-1}}$ avant celle de DR_{n-1}. Comme les deux premières matrices sont utilisées dans le calcul des classes directement accessibles à partir de $\widetilde{E}_{n-1}$, il est nécessaire de les sauvegarder en mémoire jusqu'à l'exploration totale des successeurs voisins de $\widetilde{E}_{n-1}$, autrement il faudrait les recalculer à chaque exploration. Par conséquent, pour une gestion optimale de l'espace mémoire, il est nécessaire d'adopter une stratégie d'exploration en largeur puis en profondeur ; ceci permet de ne garder en mémoire ces matrices que le temps de l'exploration des successeurs. L'algorithme d'énumération adoptant cette stratégie décrit ci-après utilise une procédure récursive **Explorer-classe** dont le rôle est d'explorer les successeurs voisins de la classe courante en entrée de la procédure. Des structures de pile *CE* et de liste *CC* permettent de sauvegarder temporairement respectivement les classes calculées et les classes à explorer. Le graphe des classes est calculé de manière incrémentale jusqu'à qu'il n y ait plus de classes à explorer où que la profondeur du graphe ait surpassé une valeur prédéfinie *Prof.* Cette condition permet de mettre fin au processus d'énumération dans le cas d'un processus infini du par exemple à un réseau non borné.

DEBUT DE L'ALGORITHME

Déclaration

CE : pile de type classe ; (*contient les classes non encore totalement explorées*)
Gr : structure graphe ; (*contient le graphe à calculer*)
CC : liste de type classe ; (*contient les classes calculées*)
$\widetilde{D}_{n-1}, \overleftarrow{\beta_{n-1}}, \overrightarrow{\beta_{n-1}}, DR_{n-1}, \widetilde{D}_n$:matrices ;
Prof : constante entière ; (*détermine le profondeur maximale du graphe*)

Procédure Explorer-classe ($\widetilde{E}_{\mathbf{n-1}}$:type classe)
Début

Calculer-trans (Te_{n-1}, Td_{n-1}, Ti_{n-1}, Tp_{n-1}, $Tv_{n-1}, Tf_{n-1}, Ts_{n-1}$) ;
Calculer-Rdv ($Enable_{n-1}$, $Inihib_{n-1}$, $Interdit_{n-1}$ $Senable_{n-1}$) ;
Calculer-les-2-matrices($\overrightarrow{\beta_{n-1}}, \overleftarrow{\beta_{n-1}}$); (**Utilisation des formules de la Proposition 1**)
Calculer-matrice(DR_{n-1}).

Si $\widetilde{E}_{n-1}$ est consistante selon le *Théorème.5* **alors** (*La classe $\widetilde{E}_{n-1}$ peut être explorée*)

Pour chaque r_n **appartenant à** $Senable_{n-1}$
Faire
Si r_n est franchissable en appliquant le *Théorème.6* **alors**
$\widetilde{E}_n$:=*Classe-accessible* ($\widetilde{E}_{n-1}$, r_n) (* *Utilisation de la Proposition.2* *)
Calculer-event(r_n, L) ; (* *Calcul du label L du rendez-vous* r_n *)
Si $\widetilde{E}_n \notin CC$ **alors** (* *Non équivalente selon le Théorème.7*)
Empiler(*CE*, $\widetilde{E}_n$) ;
$CC \leftarrow \widetilde{E}_n + CC$;
Créer-arc-graphe($\widetilde{GR}$, $\widetilde{E}_n$, *L*)
Fsi

Fsi
Fait

$\widetilde{E}$:=*Dépiler*(*CE*) ;
Tant que (*Prof*$\neq 0$) $\wedge$ ($\widetilde{E} \neq \widetilde{E}_{n-1}$)
Faire
Prof :=*Prof-1* ;
Explorer-classe($\widetilde{E}$) ;
$\widetilde{E}$:=*Dépiler*(*CE*) ;
Prof :=*Prof+1* ;
Fait

FIN
(*————————————*Fin de déclaration*————————————*)

DEBUT

Calculer la classe initiale $\widetilde{E}_0$=(M_0,D_0) ;
Créer-structure-graphe($\widetilde{GR}$, $\widetilde{E}_0$) ;
$CC \leftarrow \widetilde{E}_0 + CC$;
Empiler(*CE*, $\widetilde{E}_0$) ;

Explorer-classe($\widetilde{E}_0$) ;
$\widetilde{E}$ *:=Dépiler(CE) ;*

Si *CE* est non vide **alors** le graphe $\widetilde{GR}$ n'est pas complètement exploré.

FIN DE L'ALGORITHME

5.3.2 Etude de la complexité de l'algorithme

L'étude de la complexité du calcul de chaque classe est présentée ci-après en considérant le pire des cas. Soit $\widetilde{E}_{n-1}$ une classe accessible dans $\widetilde{GR}$, nous assumons $l =| Te_{n-1} |$ et $m =| Senable_{n-1} |$ respectivement, le nombre de transitions sensibilisées et le nombre de rendez-vous fortement sensibilisés pour $\widetilde{E}_{n-1}$.

En se basant sur les formules données dans la *Proposition*.1, les complexités de calcul des deux matrices $\overrightarrow{\beta_{n-1}}$ et $\overleftarrow{\beta_{n-1}}$ sont estimées respectivement à $o(l^2)$ et $o(3l^2)$. De plus, le calcul des éléments de la matrice DR_{n-1} est effectué en un temps égal à $o(l(m+1))$. Donc, le calcul de la matrice DR_{n-1} est la somme des précédentes complexités, soit *(A)* : $o(4l^2 + l \times m + l)$

Par ailleurs, la complexité de calcul de la classe $\widetilde{E}_n$ accessible à partir de $\widetilde{E}_{n-1}$ en franchissant le rendez-vous r_n est due pricipalement au calcul de la matrice $\widetilde{D}_n$. La détermination de cette dernière implique des calculs supplémentaires réalisés par les formules de la *Proposition*.2. Toutefois, la complexité de l'élément $DS_n[n-1, n]$ est obtenue en utilisant les formules de la *Proposition 3* et est égale à $o(2m+2l)$. Par conséquent, la complexité du calcul de la matrice $\widetilde{D}_n$ est estimée à *(B)*: $o(l^2(\frac{1}{m}+4)+l(6+m)+2m)$. Donc, la complexité golbale du calcul d'une classe est la somme des complexités *(A)* et *(B)* soit de degré deux égale à $o(l^2(\frac{1}{m}+8)+l(7+2m)+2m)$ et corrigible à $o(l^2 + l \times m + m)$.

A d'autres égards, le test de consistance donné dans le *Théorème*.5 peut être estimé à $o(l)$, et les tests de franchissement à partir de la classe $\widetilde{E}_{n-1}$ prennent un temps de complexité égal à $o(2(m^2 - m)$. Enfin, le test d'équivalence entre deux classes accessibles de marquages identiques, est de complexité égale à $o(l^2)$.

En conclusion, la complexité du calcul d'une classe induite par notre algorithme en considérant le pire des cas est polynômiale d'ordre deux. De plus, elle est sensiblement inférieure à celle que l'on pourrait obtenir dans le meilleur des cas en utilisant l'algorithme de *Floyd/Warshall* estimée à $o((l+m)^3)$.

5.4 Exemple

En appliquant la technique d'énumération développée précédemment sur le $STPTPN$ de la *Figure*.5.2, nous sommes en mesure de recenser toutes les classes accessibles dans $\widetilde{GR}$. Pour rappel nous avons : $r_1 = (Sor, \{t_4, t_5\}), r_2 = (Mas, \{t_1\} \implies \{t_3\}), r_3 = (Or, \{t_1, t_2\})$, $r_4 = (And, \{t_2, t_5\})$, et $r_5 = (And, \{t_5\})$.

Comme prévu, le graphe sur-approximée obtenu est exact malgrè l'existence de l'arc inhibiteur induisant la sur-approximation des valeurs $\widetilde{D}_1[t_1, t_2]$ et $\widetilde{D}_1[t_2, t_1]$; Aucun comportement additif n'est constaté par rapport au graphe exact donné en *Figure*.5.3. Par conséquent, à partir de la

classe initiale $\widetilde{E}_0 = (M_0, \widetilde{D}_0)$ le modèle peut évoluer seulement par le franchissement du rendez-vous r_1, puisque les rendez-vous r_4 et r_5 sont non franchissables : $DR_0[r_4, r_1] = -2 \prec 0$ et $DR_0[r_5, r_1] = -2 \preceq 0$. La classe accessible $\widetilde{E}_1 = (M_1, D_1)$ permet de franchir les rendez-vous [10] r_3 ou r_4. Par contre, le rendez-vous r_5 ne peut se réaliser [11] car nous constatons : $DR_1[r_5, r_4] = 0$.

Le franchissement de r_4 mène vers la classe $\widetilde{E}_{21} = (M_{21}, D_{21})$. Cette dernière est *temporellement inconsistante* du fait de la violation des contraintes de r_2 : $DR_0[\bullet, r_2] = -1 \prec 0$. Par ailleurs, le franchissement de r_3 produit la classe consistante $\widetilde{E}_{22} = (M_{22}, \widetilde{D}_{22})$. Le processus d'énumération se poursuit jusqu'à sa terminaison par regroupement des classes équivalentes en assumant les hypothèses du *Théorème*.7. Par exemple, les classes $\widetilde{E}_{54}, \widetilde{E}_{52}$, et $\widetilde{E}_{41}$ d'une part et $\widetilde{E}_{31}, \widetilde{E}_{42}, \widetilde{E}_{43}$, et $\widetilde{E}_{53}$ d'autre part sont non équivalentes alors qu'elles ont le même marquage [12].

$\widetilde{D}_1$	$\bullet$	t_1	t_2	t_5
$\bullet$	0	1	5	4
t_1	0	1	5	4
t_2	−2	−1	1	2
t_5	−2	−1	3	2

$\overrightarrow{\beta_1}$	r_3	r_4	r_5
t_1	1	4	4
t_2	1	1	2
t_5	3	2	2

$\overleftarrow{\beta_1}$	r_3	r_4	r_5
t_1	4	0	2
t_2	0	0	0
t_5	2	0	0

DR_1	$\bullet$	r_3	r_4	r_5
$\bullet$	0	5	4	4
r_3	0	0	1	4
r_4	−2	1	0	2
r_5	−2	3	0	0

$\widetilde{D}_{21}$	$\bullet$	t_1	t_3	t_5
$\bullet$	0	-1	2	4
t_1	0	1	2	4
t_3	−1	−2	1	3
t_5	−2	−1	0	2

$\overrightarrow{\beta_{21}}$	r_2	r_5
t_1	1	4
t_3	−2	3
t_5	−1	2

$\overleftarrow{\beta_{21}}$	r_2	r_5
t_1	0	2
t_3	−1	1
t_5	−2	0

DR_{21}	$\bullet$	r_2	r_5
$\bullet$	0	−1	4
r_2	0	0	4
r_5	−2	−1	0

$\widetilde{D}_{22}$	$\bullet$	t_1	t_3	t_5
$\bullet$	0	1	2	4
t_1	0	1	2	4
t_3	−1	0	1	3
t_5	0	1	2	2

$\overrightarrow{\beta_{22}}$	r_2	r_5
t_1	1	4
t_3	0	3
t_5	1	2

$\overleftarrow{\beta_{22}}$	r_2	r_5
t_1	0	2
t_3	−1	1
t_5	0	0

DR_{22}	$\bullet$	r_2	r_5
$\bullet$	0	1	4
r_2	0	0	4
r_5	0	1	0

$\widetilde{D}_{31}$	$\bullet$	t_1	t_5	t_6
$\bullet$	0	1	4	1
t_1	0	1	4	1
t_5	0	1	2	1
t_6	−1	0	3	0

$\overrightarrow{\beta_{31}}$	r_5	r_6
t_1	4	1
t_5	2	1
t_6	3	0

$\overleftarrow{\beta_{31}}$	r_5	r_6
t_1	2	1
t_5	0	1
t_6	1	1

DR_{31}	$\bullet$	r_5	r_6
$\bullet$	0	4	1
r_5	0	0	1
r_6	−1	3	0

$\widetilde{D}_{32}$	$\bullet$	t_1	t_3	t_5
$\bullet$	0	1	2	4
t_1	0	1	2	4
t_3	0	0	1	4
t_5	−2	−1	0	2

$\overrightarrow{\beta_{32}}$	r_2	r_5
t_1	1	4
t_3	1	4
t_5	−1	2

$\overleftarrow{\beta_{32}}$	r_2	r_5
t_1	0	2
t_3	0	2
t_5	−2	0

DR_{32}	$\bullet$	r_2	r_5
$\bullet$	0	1	4
r_2	0	0	4
r_5	−2	−1	0

10. La transition t_1 n'est plus inhibée après la consommation du jeton de p_5 suite au tir de t_4.
11. Le rendez-vous r_5 ne peut se produire en dehors des instants de franchissement de r_4.
12. Les matrices $\widetilde{D}$ sont non égales.

$\widetilde{D}_{41}$	$\bullet$	t_5
$\bullet$	0	3
t_5	0	2

$\overrightarrow{\beta}_{41}$	r_5
t_5	2

$\overleftarrow{\beta}_{41}$	r_5
t_5	0

DR_{41}	$\bullet$	r_5
$\bullet$	0	3
r_5	0	0

$\widetilde{D}_{42}$	$\bullet$	t_1	t_5	t_6
$\bullet$	0	1	4	1
t_1	0	1	4	1
t_5	-2	-1	2	-1
t_6	0	0	4	0

$\overrightarrow{\beta}_{42}$	r_5	r_6
t_1	4	1
t_5	2	-1
t_6	4	0

$\overleftarrow{\beta}_{42}$	r_5	r_6
t_1	-1	1
t_5	0	-1
t_6	2	0

DR_{42}	$\bullet$	r_5	r_6
$\bullet$	0	4	1
r_5	-2	0	-1
r_6	0	4	0

$\widetilde{D}_{43}$	$\bullet$	t_1	t_5	t_6
$\bullet$	0	1	4	1
t_1	0	1	4	1
t_5	-1	0	2	0
t_6	-1	0	3	0

$\overrightarrow{\beta}_{43}$	r_5	r_6
t_1	4	1
t_5	2	0
t_6	3	0

$\overleftarrow{\beta}_{43}$	r_5	r_6
t_1	2	1
t_5	0	0
t_6	1	0

DR_{43}	$\bullet$	r_5	r_6
$\bullet$	0	4	1
r_5	-1	0	0
r_6	-1	3	0

$\widetilde{D}_{51}$	$\bullet$	t_5
$\bullet$	0	4
t_5	-2	2

$\overrightarrow{\beta}_{51}$	r_5
t_5	2

$\overleftarrow{\beta}_{51}$	r_5
t_5	0

DR_{51}	$\bullet$	r_5
$\bullet$	0	4
r_5	-2	0

$\widetilde{D}_{52}$	$\bullet$	t_5
$\bullet$	0	4
t_5	-1	2

$\overrightarrow{\beta}_{52}$	r_5
t_5	2

$\overleftarrow{\beta}_{52}$	r_5
t_5	0

DR_{52}	$\bullet$	r_5
$\bullet$	0	4
r_5	-1	0

$\widetilde{D}_{53}$	$\bullet$	t_1	t_5	t_6
$\bullet$	0	0	4	0
t_1	0	0	4	0
t_5	-2	-2	2	-2
t_6	0	0	4	0

$\overrightarrow{\beta}_{53}$	r_5	r_6
t_1	4	0
t_5	2	-2
t_6	4	0

$\overleftarrow{\beta}_{53}$	r_5	r_6
t_1	2	0
t_5	0	-2
t_6	2	0

DR_{53}	$\bullet$	r_5	r_6
$\bullet$	0	4	0
r_5	-2	0	-2
r_6	0	4	0

$\widetilde{D}_{54}$	$\bullet$	t_5
$\bullet$	0	3
t_5	-1	2

$\overrightarrow{\beta}_{54}$	r_5
t_5	2

$\overleftarrow{\beta}_{54}$	r_5
t_5	0

DR_{54}	$\bullet$	r_5
$\bullet$	0	3
r_5	-1	0

5.5 Conclusion

Nous avons développé durant ce chapitre une technique de construction du graphe des classes d'un $STPTPN$. Pour cet effet, il fallait faire avec les sémantiques complexes induites par les mécanismes de synchronisation[53], des chronomètres[46] et de la priorité en temps réel [10]. Une première approche permettant d'énumérer le graphe des classes exact d'un $STPTPN$ noté GR, a été présentée. Pour ce faire, chaque classe accessible est représentée par un marquage M et un système des contraintes D sous forme polyédrique. Comme les coûts de calcul du système D et des différents tests étant prohébitifs (exponentiel au nombre de variables) ; nous avons exploré une seconde approche calculant la sur-approximation DBM du graphe des classes d'un $STPTPN$, noté $\widetilde{GR}$. L'implémentation de cette technique de sur-approximation permet d'optimiser le calcul d'une classe dont la complexité est ramenée à un degré polynômial d'ordre deux. Par conséquent,

le graphe obtenu $\widetilde{GR}$ est un graphe exact identique à GR si le modèle ne contient pas d'arcs inhibiteur, sinon il *peut être* une sur-approximation de GR préservant néanmoins un sous ensemble de propriétés linéaires notamment celles liées à la "*sûreté*". D'ailleurs, nous discutons dans le chapitre suivant les propriétés que préserve cette construction et comment les déterminer.

Chapitre 6

Analyse de l'espace d'état d'un $STPTPN$

Nous discutons dans ce chapitre comment exploiter les graphes des classes exact et sur-approximé pour l'analyse des propriétés linéaires et temps réel du système modélisé. Nous montrons également comment calculer une surapproximation des délais minimaux et maximaux de toutes sous-séquences contenues dans un chemin ou un cycle du graphe. Nous proposons enfin un nouvelle conceptualisation de la consistance des documents multimédia, inférée du processus de modélisation développé dans le $Chapitre.III$; L'approche en question est appliquée sur l'exemple présenté dans §.IV.4.2.

6.1 Les propriétés générales du graphe des classes d'un $STPTPN$

Le but de construire le graphe des classes d'un $STPTPN$ est de déduire les propriétés les plus importantes du modèle (e.g. accessibilité, blocage, vivacité,...etc). En considérant le graphe exact GR, toutes les propriétés linéaires du modèle (celles qui peuvent être vérifiées en utilisant des logiques linéaires telles que LTL) sont préservées. Par conséquent, pour vérifier si un $STPTPN$ satisfait une propriété donnée ; il suffit de la vérifier sur son graphe GR. Ainsi, les propriétés rattachées à l'atteignabilité peuvent être immédiatement déduites en explorant le graphe de la même manière que pour les réseaux de Petri standards. Cependant, pour le graphe sur-approximé $\widetilde{GR}$, un sous ensemble de propriétés du $STPTPN$ sont préservées ; les principales d'entre elles sont énumérées ci-après :

- Tout marquage M_n respectivement, état e_n accessible dans GR, l'est aussi dans $\widetilde{GR}$.
- Tout marquage M_n respectivement, état e_n non accessible dans $\widetilde{GR}$, est non atteignable dans GR.
- Toute séquence S_n franchissable dans GR, l'est aussi dans $\widetilde{GR}$.
- Toute séquence S_n non franchissable dans $\widetilde{GR}$, l'est aussi dans GR.

A partir des relations précédentes, on peut vérifier si une propriété de *sûreté* i.e, "*quelque chose de mauvais n'arrive jamais",* est vraie ou non. Par conséquent, si cette dernière est vraie dans $\widetilde{GR}$ elle est aussi dans GR. Par ailleurs, si cette propriété est fausse dans $\widetilde{GR}$ (i.e, le modèle n'est pas sur) ; alors on peut extrapoler et dire qu'il y a une grande probabilité que GR et donc le modéle

ne soit pas sur avec le risque d'être pessimiste [1].

Pour vérifier une propriété particulière (discrete ou quantitative) sur le graphe sur-approximé $\widetilde{GR}$, on peut étendre le $STPTPN$ en amont avec des places ou des transitions d'observation permettant de modéliser la propriété à analyser. Ensuite la vérification de cette propriété sur le graphe $\widetilde{GR}$ se fait en vérifiant par exemple l'atteignabilité ou non du marquage des places de l'observateur[83][93].

Par ailleurs, concernant les propriétés temps réel (e.g. propriété du temps borné) réquiérant la détermination des temps minimum et maximum d'une séquence de franchissement, elles peuvent être calculées à partir du graphe des classes exact GR. Pour ce faire, *Bucci et autres* [42] ont démontré que pour un *RdPT* à chronomètre, le calcul du temps maximum ou minimum d'une séquence de franchissement nécessite la résolution d'un système linéaire dont la complexité est exponentielle à la longueur de la séquence et au nombre de transitions sensibilisées. Ce coût de calcul étant encore plus exacerbé pour un $STPTPN$ [2], nous explorons par la suite une technique sur-approximant ces délais. L'algorithme en question s'applique aussi bien sur le graphe exact [3] que le sur-approximé. Cependant, nous avons besoin de réaliser des calculs supplémentaires pour déterminer ces informations. Pour ce faire, nous montrons ci-après comment calculer en parcourant le long d'un chemin du graphe, les temps minimum et maximum sur-approximés de n'importe quelle sous séquence du chemin exploré.

6.2 Temps minimal et maximal d'une séquence de franchissement

Dans ce qui suit, les notations supposent que l'approche s'applique sur le graphe sur-approximé $\widetilde{GR}$. Cependant, la formalisaion et la technique developpées ci-après s'appliquent aussi sur le graphe exact GR, pour cela il suffit de considérer au niveau de chaque classe E_n accessible dans GR le système réduit $\widetilde{D_n}$ en place du système de contraintes D_n.

Afin de formaliser le processus de calcul, nous considérons une séquence de franchissement $S_i^n = (r_{i+1}, .., r_n)$. S_i^n décrivant un chemin dans le graphe $\widetilde{GR}$ partant du noeud représentant la classe $\widetilde{E}_i$ au noeud représentant la classe $\widetilde{E}_n$. Par conséquent, les temps minimum et maximum de la séquence $(r_{i+1}, .., r_n)$ sont donnés par la somme $\underline{r_{i+1}} + .. + \underline{r_n}$, et dont les valeurs optimales sont prises dans l'espace de franchissement capturant toutes les solutions admissibles de $(Q^+)^{n-i}$, et défini comme suit :

Définition 21 *Soit $\widetilde{E}_n$ une classe accessible dans $\widetilde{GR}$ à partir de la classe $\widetilde{E}_i$ après le franchissement de la séquence $S_i^n = (r_{i+1}, .., r_n)$. On dénote par space$(S_i^n)$ l'espace de franchissement défini sur $(Q^+)^{n-i}$ des vecteurs $(\theta_{i+1}, .., \theta_n)$ telle que la séquence S_i^n peut être franchie aux dates relatives $(\underline{r_{i+1}}, .., \underline{r_n}) := (\theta_{i+1}, .., \theta_n)$:*

$$space(S_i^n) := \left\{ \begin{array}{l} (\underline{r_{i+1}}, .., \underline{r_n}) := (\theta_{i+1}, .., \theta_n) \in (Q^+)^{n-i} \mid \\ \forall e_n \in \widetilde{E}_n, \forall j \in \{i, .., n-1\} \quad \exists e_j \in \widetilde{E}_j \ , \\ \qquad e_i \stackrel{L(r_{i+1})}{\longrightarrow} e_{i+1} .. e_j \stackrel{L(r_{j+1})}{\longrightarrow} ..e_{n-1} \stackrel{L(r_n)}{\longrightarrow} e_n \end{array} \right\}$$

1. Il se peut qu'il existe un chemin additif dans $\widetilde{GR}$ qui ne vérifie pas la propriété de sureté, alors que tous les chemins de GR la vérifient.

2. La complexité depend en plus du nombre de rendez-vous fortement sensibilisés.

3. Dans ce cas, la technique exploite seulement le sous-système $\widetilde{D}_n$ de D_n associé à chaque classe accessible E_n.

Notons que les variables $\underline{r_{i+1}}, .., \underline{r_n}$ sont dépendantes temporellement. De là, le vecteur $(\underline{r_{i+1}}, .., \underline{r_n})$ $:=(\theta_{i+1}, .., \theta_n)$ est une solution admissible, s'il y a un état e_i dans $\widetilde{E}_i$ qui peut franchir consécutivement : le rendez-vous r_{i+1} à la date relative $\underline{r_{i+1}} := \theta_{i+1}$, r_{i+2} à la date relative $\underline{r_{i+2}} := \theta_{i+2}, ..$, et finallement le rendez-vous r_n à la date relative $\underline{r_n} := \theta_n$ pour atteindre l'état e_n accessible dans $\widetilde{E}_n$. De plus, puisque une classe contient tous les états accessibles après franchissement de la même séquence, alors chaque vecteur $(\theta_{i+1}, .., \theta_n)$ de $space(S_i^n)$ dénotera un état accessible dans $\widetilde{E}_n$.

Définition 22 *L'espace $space(S_i^n)$ peut être exprimé en fonction de $space(S_i^{n-1})$, comme suit : Si $space(S_i^{n-1}) \neq \varnothing$ alors*

$$space(S_i^n) := \left\{ \begin{array}{l} (\underline{r_{i+1}}, .., \underline{r_n}) \mid (\underline{r_{i+1}}, .., \underline{r_{n-1}}) \in space(S_i^{n-1}) \wedge \\ 0 \preceq Low_{n-1}(r_n) \preceq \underline{r_n} \preceq \underset{\forall r \in Senable_{n-1}}{MIN} \{Up_{n-1}(r)\} \\ 0 \preceq Low_{n-1}(r_n) \preceq \underline{r_n} \prec \underset{\forall r' \in \Phi_{n-1}(r_n)}{MIN} Low_{n-1}(r') \end{array} \right\}$$

En d'autres termes, si r_n est franchissable dans $space(S_i^{n-1})$, alors $space(S_i^n)$ représente tous les vecteurs de $space(S_i^{n-1})$ (états de $\widetilde{E}_{n-1}$) satisfaisant la condition de franchissement donnée en $Définition.13$ et la restriction de l'espace $space(S_i^n)$ aux variables $(\underline{r_{i+1}}, .., \underline{r_{n-1}})$ est un sous ensemble de $space(S_i^{n-1})$. Par conséquent, le calcul des temps de performance est déterminé récursivement en faisant des projections sur les espaces de franchissement calculés antérieurement, et nous calculons les temps de performances relatifs à la séquence S_i^n gràce à la fonction DS_n définie

comme suit :

Définition 23 ***(Fonction des distances de temps)*** *Soit $\widetilde{E}_n$ une classe accessible dans $\widetilde{GR}$, après le franchissement de la séquence $S_i^n = r_{i+1}, .., r_n$. On définit au point (n) la fonction DS_n appelée fonction des distances de temps calculant les distances minimum et maximum de tout chemin allant du point $(i) \in \{0, .., n\}$ au point (n).*

$DS_n : (\{0, .., n\} \cup Te_n \cup Senable_n)^2 \longrightarrow Q \cup \{\infty\}$

$\forall i \in \{0, .., n-1\}, \qquad \forall r \in Senable_n \qquad \forall t \in Te_n$

$DS_n[i,n] := \underset{space(S_i^n)}{MAX} \{\underline{r_{i+1}} + .. + \underline{r_n}\}$

$DS_n[n,i] := - \underset{space(S_i^n)}{MIN} \{\underline{r_{i+1}} + .. + \underline{r_n}\}$

$DS_n[i, \overrightarrow{r}] := \underset{space(S_i^n)}{MAX} \{\underline{r_{i+1}} + .. + \underline{r_n} + Up_n(r)\}$

$DS_n[i, \overleftarrow{r}] := \underset{space(S_i^n)}{MAX} \{\underline{r_{i+1}} + .. + \underline{r_n} + Low_n(r)\}$

$DS_n[r,i] := - \underset{space(S_i^n)}{MIN} \{\underline{r_{i+1}} + .. + \underline{r_n} + Low_n(r)\}$

$DS_n[i,t] := \underset{space(S_i^n)}{MAX} \{\underline{r_{i+1}} + .. + \underline{r_n} + y_n(t)\}$

$DS_n[t,i] := - \underset{space(S_i^n)}{MIN} \{\underline{r_{i+1}} + .. + \underline{r_n} + x_n(t)\}$

$DS_n[i,i] := 0$

$DS_n[n,n] := 0$

Le calcul des précédentes distances est réalisé en considérant le vecteur optimal $(\theta_{i+1}, .., \theta_n)$ dans l'espace $space(S_i^n)$ tel que il existe un état de $\widetilde{E}_n$ accessible à partir d'un état de $\widetilde{E}_i$ après

franchissement de la séquence $S_i^n = (r_{i+1}, .., r_n)$ aux dates relative $(\underline{r_{i+1}}, .., \underline{r_n}) := (\theta_{i+1}, .., \theta_n)$. Par conséquent, si t est un transition sensibilisée pour $\widetilde{E}_n$, alors $DS_n[i,t]$ représente l'écart maximum entre le point (i) et la borne supérieure de t. $DS_n[t,i]$ dénote la valeur opposée de l'écart minimum entre le point (i) et la borne inférieure de t. De plus, si r est un rendez-vous fortement sensibilisé pour $\widetilde{E}_n$, alors $DS_n[i, \overrightarrow{r}]$ (respectivement, $DS_n[i, \overleftarrow{r}]$), représente l'écart maximum entre le point (i) et la borne supérieure de r (respectivement, la borne inférieure de r). $DS_n[r,i]$ dénote la valeur opposée de l'écart minimum entre le point (i) et la borne inférieure de r. Finallement, $DS_n[i,n]$ (respectivement $DS_n[n,i]$), calcule la distance de temps maximum (respectivement, la valeur opposée de la distance de temps minimum), entre les points (i) et (n).
Les formules de calcul de la fonction DS_n sont données par la proposition suivante, et permettent de calculer une sur-approximation des distances de temps.

Proposition 3 *Soit $\widetilde{E}_n = (M_n, \widetilde{D}_n)$ une classe accessible dans $\widetilde{GR}$, à partir de la classe $\widetilde{E}_{n-1} = (M_{n-1}, \widetilde{D}_{n-1})$ après le franchissement du rendez-vous r_n. La fonction DS_n associée à $\widetilde{E}_n$ est calculée récursivement par sur-approximation à partir de DS_{n-1} comme suit :*

$DS_n[n,n] := 0.$

$DS_n[n, \overrightarrow{r}] := DR_n[\bullet, r].$

$$DS_n[n, \overleftarrow{r}] := MAX \left(\begin{cases} \underset{\forall t \in Trans(r)}{MAX} \left\{ \widetilde{D}_n[\bullet,t] + \alpha(t) \right\} & Si\ typ(r) \in \{And, Wand, Amas, Async\} \\ \underset{\forall t \in Trans(r)}{MIN} \left\{ \widetilde{D}_n[\bullet,t] + \alpha(t) \right\} & Si\ typ(r) \in \{Or, Sor, Omas\} \\ \widetilde{D}_n[\bullet,t_m] + \alpha(t_m) & Si\ typ(r) \in \{Mas, Smas, Wmas\} \\ & \wedge (t_m = MA(r)) \end{cases} \right.$$
$$\left. 0 \right)$$

$DS_n[r,n] := DR_n[r,\bullet].$

$DS_n[n,t] := \widetilde{D}_n[\bullet,t].$

$DS_n[t,n] := \widetilde{D}_n[t,\bullet].$

$\forall i \in \{0, .., n-1\},$

$$DS_n[i,n] := MIN \begin{cases} \underset{\forall r \in Senable_{n-1}}{MIN} DS_{n-1}[i, \overrightarrow{r}] \\ \underset{\forall r \in \Phi_{n-1}(r_n)}{MIN} DS_{n-1}[i, \overleftarrow{r}] \end{cases}$$
$$DS_n[n,i] := DS_{n-1}[r_n, i]$$

$\forall i \in \{0, .., n-1\}, \quad \forall r \in Senable_n$

$$DS_n[i, \overrightarrow{r}] := \begin{cases} \underset{\forall t \in Trans(r)}{MAX} \{DS_n[i,t]\} & Si\ typ(r) \in \{Or, Wand, Wmas\} \\ \underset{\forall t \in Trans(r)}{MIN} \{DS_n[i,t]\} & Si\ typ(r) \in \{Sor, And, Smas, Async\} \\ DS_n[i,t_m] & Si\ typ(r) \in \{Mas, Omas, Amas\} \wedge (t_m = MA(r)) \end{cases}$$

$$DR_n[i, \overleftarrow{r}] := MAX \left(\begin{cases} \underset{\forall t \in Trans(r)}{MAX} \{DS_n[i,t] + \alpha(t)\} & Si\ typ(r) \in \{And, Wand, Amas, Async\} \\ \underset{\forall t \in Trans(r)}{MIN} \{DS_n[i,t] + \alpha(t)\} & Si\ typ(r) \in \{Or, Sor, Omas\} \\ DS_n[i,t_m] + \alpha(t_m) & Si\ typ(r) \in \{Mas, Smas, Wmas\} \\ & \wedge (t_m = MA(r)) \end{cases} \right.$$
$$\left. DS_n[i,n] \right)$$

$$DR_n[r,i] := \begin{cases} \underset{\forall t \in Trans(r)}{MIN} \{DS_n[t,i]\} & Si\ typ(r) \in \{And, Wand, Amas, Async\} \\ \underset{\forall t \in Trans(r)}{MAX} \{DS_n[t,i]\} & Si\ typ(r) \in \{Or, Sor, Omas\} \\ DS_n[t_m,i] & Si\ typ(r) \in \{Mas, Smas, Wmas\} \wedge (t_m = MA(r)) \end{cases}$$

$\forall i \in \{0,..,n-1\}, \quad \forall t \in Te_n$

Si t *est persistante*

Si $t \notin Ti_{n-1}$ (t *est non inhibée au point* $n-1$)

$$DS_n[i,t] := MIN(DS_{n-1}[i,t], \quad DS_n[i,n] + \widetilde{D}_n[\bullet,t]).$$

$$DS_n[t,i] := MIN(DS_{n-1}[t,i], \quad DS_n[n,i] + \widetilde{D}_n[t,\bullet]).$$

Si $t \in Ti_{n-1}$ (t *est inhibée au point* $n-1$).

$$DS_n[i,t] \preceq MIN \begin{cases} DS_{n-1}[i,t] + DS_n[n-1,n] \\ DS_n[i,n] + \widetilde{D}_n[\bullet,t] \end{cases}$$

$$DS_n[t,i] \preceq MIN \begin{cases} DS_{n-1}[t,i] + DR_{n-1}[r_n,\bullet] \\ DS_n[n,i] + \widetilde{D}_n[t,\bullet] \end{cases}$$

Si t *est nouvellement sensibilisée.*

$$DS_n[i,t] := DS_n[i,n] + LFT(t)$$
$$DS_n[t,i] := DS_n[n,i] - EFT(t)$$

Preuve. La preuve est donnée en *Annexe A* ■

TABLE 6.1 – Les distances de temps minimales et maximales du chemin $0 \longrightarrow 41 \longrightarrow 51$

DS[i,j]	**0**	1	**22**	**31**	41	51
0	0	2	5	6	6	6
1	0	0	4	4	4	4
22	0	0	0	4	4	4
31	0	0	0	0	1	4
41	-1	-1	-1	-1	0	3
51	-3	-3	-1	-1	0	0

Le calcul des temps de performance de la séquence S_i^n est entrepris en parcourant le chemin du graphe, partant du noeud représentant la classe (i) en réalisant à chaque noeud intermédiaire le calcul des éléments de la fonction $DS_j(j = i..n)$ jusqu'à atteindre le noeud final (n). A ce point, nous aurons calculé les coefficients $DS_n[n,i]$ et $DS_n[i,n]$ dénotant respectivement, la valeur opposée de la distance minimum de temps, et la distance maximum de temps entre les noeuds (i) et (n). Toutefois, il est à noter que les valeurs obtenues peuvent une sur-aproximation des valeurs exactes [4], lorsque le modèle contient des arcs inhibiteurs. Dans le cas contraire, la proposition précédente calcule les temps de performance exacts.

Par exemple, la *Table*.6.1 reproduit les temps de performances de n'importe quelle sous-séquence du chemin $(\widetilde{E}_0 \longrightarrow \widetilde{E}_{41} \longrightarrow \widetilde{E}_{51})$ du graphe donné en *Figure*.5.3. Pour obtenir cette

4. En clair, l'intervalle des temps calculés inclut l'intervalle exact.

table, nous appliquons la proposition précédente pour calculer les éléments $DS[i,j]$ en parcourant le chemin $(\widetilde{E}_0 \longrightarrow \widetilde{E}_{41} \longrightarrow \widetilde{E}_{51})$.

Par ailleurs, la complexité de calcul des éléments $DS_n[n,i]$ et $DS_n[i,n]$ depend de la longueur du chemin égale à $(n-i)$, et est estimée à $o((n-i)(12l+2m+1))$ où $m = \frac{\sum_{j=i}^{n-1}|Senable_j|}{n-i}$ et $l = \frac{\sum_{j=i}^{n-1}|Te_j|}{n-i}$ sont respectivement le nombre moyen de rendez-vous fortement sensibilisés et le nombre moyen de transitions sensibilisées pour chaque point de la séquence S_i^n. Cette complexité est donc fonction de la longueur du chemin et du nombre d'éléments sensibilisés.

Remarque : Le calcul de l'élément $DS_n[n-1,n]$ a été déja réalisé lors du calcul de chaque classe du graphe (voir *Proposition 2*), et peut êre sauvegardé comme paramètre de la classe pour éviter un calcul redondant.

6.3 Vérification de la consistance des documents multimédia

Dans ce qui suit, nous déclinons une approche de vérification exhaustive de la consistance des documents multimédia, basée sur le graphe d'accessibilité calculé auparavant [19][20]. Nous restreignons cette étude seulement aux graphes exacts. Les graphes sur-approximés nécessitent quand-t à eux un traitement différent permettant de prendre en compte les chemins additifs ; Cet aspect néanmoins important n'est pas à l'ordre du jour de cette thèse.

La vérification formelle de la "*consistance temporelle*" d'un document multimédia en utilisant l'analyse par énumération a été traitée dans [85][86]. Un document est qualifié de *temporellement consistant,* si le graphe obtenu à partir de la spécification formelle capturant ses contraintes de temps et de synchronisation est "*temporellement consistant*". Cette dernière propriété est vérifiée sur le graphe, si "*l'événement caractérisant le début de la présentation multimédia est nécessairement suivie (et ce pour chaque chemin du graphe) par un événement caractérisant sa fin*".

Une "*présentation temporellement consistante*" assume que les contraintes de temps et de synchronisation de chaque média impliqué dans la présentation multimedia soient satisfaites quelque soit le scénario exécuté par le lecteur. Donc, étant donné un scénario consistant formalisé par un chemin du graphe, l'exécution de chaque média O dans cela est explicité par l'occurrence de son événement de début *B(O)* et sa bonne terminaison est renseignée par l'occurrence de soit son événement de fin *E(O)* ou l'événement de l'interaction utilisateur [5] sur le lien qui lui est rattaché *Anc*.

Cependant, le "*concept de consistance temporelle*" ne permet de vérifier que les incohérences temporelles ; les disfonctionnements dues aux conflits de ressources par exemple sont totalement ignorés par cette méthodologie et ne peuvent être détectés par l'utilisation de ce concept [6]. Nous proposons par conséquent d'étendre ce concept de consistance à la prise en charge de cette problématique en considérant les principaux fonctionnements des lecteurs [27][81].

Avant d'aborder cette approche de vérification, nous avons besoin préalablement d'introduire quelques notations.

5. Dans le respet de notre modélisation, la transition *E(O)* ne peut être interdite de tir, et ne peut être en conflit qu'avec une transition *Anc* modélisant le lien rattaché à *O*.

6. Particulièrement dans le cas d'interactions d'utilisateurs nécessitant la disponibilité des ressources d'affichage.

Notation 4 *Soit GR le graphe d'atteignabilité exact "temporellement consistant" obtenu à partir de la spécification STPTPN correspondant à la présentation multimédia et soit S un chemin dans Gr partant du noeud initial. On note Seq(O, S) dénoté par la séquence d'événement* $B(O) \xrightarrow{s} e_2$, *la trace observée dans S partant de l'événement du début de O, à l'événement dénotant sa bonne terminaison (donné par* e_2*), qui pourrait être soit son événement de fin* $e_2 = E(O)$ *ou l'événement relatif au lien qui lui est rattaché noté* $e_2 = Anc(O)$. *Par ailleurs, nous assumons que* $Seq(O, S) = \varnothing$, *si O n'apparaît pas dans S.*

Nous cherchons maintenant à inférer des propriétés générales décrivant les exécutions d'une présentation multimédia, et ce en analysant tous les comportements possibles qui peuvent être induits à partir du processus de modélisation développé dans le *Chapitre III*. Pour ce faire, nous énumérons ci-après dans le contexte d'une présentation multimédia "*temporellent consistante*", toutes les exécutions possibles d'un média de base, formalisées en terme de séquences d'événement $Seq(O, S)$. A ces dernières, on associe les interprétations exprimées selon le fonctionnement du lecteur considéré. Cet inventaire des configurations d'exécutions rendra possible par la suite la définition d'un nouveau concept de vérification.

Selon le mode de fonctionnement du lecteur *Real one* : Dans le contexte du lecteur *Real one*, nous observons sept configurations d'exécutions possibles :

- $B(O) \xrightarrow{s} E(O)$: dénotant une exécution conforme avec acquisition de la ressource tout au long de la durée d'exécution.
- $B(O) \xrightarrow{s} E(O)^*$: modélisant le cas où la ressource est acquise au lancement de O avant qu'elle ne lui soit retirée définitivement.
- $B(O)^* \xrightarrow{s} E(O)^*$: spécifiant le cas où O est dans l'impossibilité d'acquérir la ressource pour la totalité de son temps d'exécution.
- ${}^*B(O) \xrightarrow{s} E(O)$: dénotant l'acquisition de la ressource par violation active lors du lancement de O, et sa sauvegarde jusqu'à la fin de son temps d'exécution.
- ${}^*B(O) \xrightarrow{s} E(O)^*$: modélisant le cas où O acquiert la ressource par violation lors de son lancement, avant de se faire retirer définitivement la ressource par un média de plus haute priorité.
- $B(O) \xrightarrow{s} Anc(O)$: Modélisant une exécution consistante de O avec acquisition de ressource jusqu'à la terminaison de O suite à l'interaction d'un utilisateur.
- ${}^*B(O) \xrightarrow{s} Anc(O)$: dénotant le cas où O acquiert la ressource par violation lors de son lancement, et sa sauvegarde jusquà sa terminaison suite à l'interaction d'un utilisateur.

Selon le mode de fonctionnement du lecteur *Ambulant* : Dans le contexte du lecteur *Ambulant*, nous observons onze configurations d'exécutions possibles :

- $B(O) \xrightarrow{s} E(O)$: dénotant une exécution conforme, avec acquisition de la ressource tout au long de la durée d'exécution.
- $B(O) \xrightarrow{s} E(O)^*$: inférant le cas où la ressource est acquise au lancement de O sans qu'il puisse la garder définitivement jusqu'à la fin de son exécution.
- $B(O)^* \xrightarrow{s} E(O)^*$: spécifiant le cas où O est dans l'impossibilité d'acquérir la ressource initialement. Cependant la ressource pourrait être récupérée en cours de présentation mais

sans pour autant être préservée jusqu'à la fin du temps d'exécution.

- $B(O) \overset{s}{\rightarrow}^{*} E(O)$: dénotant que O ait obtenu la ressource lors de son lancement, mais qui a du la céder une fois, voire plusieurs fois, avant de pouvoir la récupérer et de la garder définitivement jusqu'à la fin de son temps d'exécution.
- ${}^{*}B(O) \overset{s}{\rightarrow} {}^{*}E(O)$: modélisant le cas où O procède à la violation de la ressource à son début, avant de se faire retirer la ressource durant son exécution, et de pouvoir enfin la récupérer définitivement pour la fin de sa présentation.
- ${}^{*}B(O) \overset{s}{\rightarrow} E(O)$: dénotant le cas où O viole la ressource lors de son lancement, en la sauvegardant jusqu'à la fin de son temps d'exécution.
- ${}^{*}B(O) \overset{s}{\rightarrow} E(O)^{*}$: modélisant le cas où O viole la ressource lors de son lancement, avant de la céder en cours d'exécution avec possibilité de la réupérer entre temps, sans pour autant pouvoir la garder jusqu'à la fin de son temps de présentation.
- $B(O)^{*} \rightarrow^{*} E(O)$: spécifiant le cas où O initie son exécution sans pouvoir acquérir la ressource, avant de la recouvrer définitivement pour la fin de sa présentation.
- $B(O) \overset{s}{\rightarrow} Anc(O)$: représentant l'exécution où O a aquis la ressource au début, *pouvant* la céder par la suite avant de la récupérer et la garder jusqu'à sa terminaison achevée par l'occurrence de l'interaction d'un utilisateur.
- ${}^{*}B(O) \overset{s}{\rightarrow} Anc(O)$: spécifiant le cas où le média O ait violé la ressource à son commencement, *avec possibilité* de la céder par la suite avant de la récupérer et de la garder jusqu'à sa terminaison achevée par l'occurrence de l'interaction d'un utilisateur.
- $B(O)^{*} \overset{s}{\rightarrow} Anc(O)$: modélisant le cas où O ne peut acquérir la ressource à son commencement avant de la récupérer et de la garder jusquà sa terminaison dénotée par l'occurrence de l'interaction d'un utilisateur.

A première vue, nous pouvons dire qu'une présentation multimédia *temporellement consistante* est sans conflit de ressources, si le graphe correspondant ne contient pas l'occurrence d'un *événement de violation passive ou active*. En d'autres termes, une présentation multimédia peut être appréciée comme *qualitative* si tous les médias exécutées dans cela, auront été perçus par le client durant la totalité de leurs exécutions.

Cette ébauche de définition n'est qu'un premier pas d'une démarche menant vers la reformulation du concept de consistance, et qui sous cette forme est encore à l'état brut et nécessité d'être raffinée encore plus. D'ailleurs, nous pouvons noter que l'hypothèse précédente est assez restrictive, partant du constat que nous ne pouvons adopter la même politique pour les médias continus (e.g. vidéo, animation, ou audio) que pour les médias discrets (e.g. texte ou image). Plus clairement, un média discret détermine un objet inanimé qui nécessité d'être visualisé durant une partie de son temps d'exécution pour permettre à l'information véhiculée d'être perçue par l'utilisateur, alors que la non perception d'un segment d'un média continu entraine une perte d'information qu'on pourrait difficilement restaurer à l'utilisateur.

Considérons le *Tableau 6.2,* qui donne une appréciation des configurations d'exécution possibles en fonction du lecteur considéré. Nous pouvons remarquer que pour les deux lecteurs, les exécutions consistantes avec acquisition totale de la ressource sont celles qui infèrent les séquences $B(O) \overset{s}{\rightarrow} E(O)$,${}^{*}B(O) \overset{s}{\rightarrow} E(O), B(O) \overset{s}{\rightarrow} Anc(O)$ ou ${}^{*}B(O) \overset{s}{\rightarrow} Anc(O)$. Cependant, dans le cas du lecteur *Ambulant,* les configurations $B(O) \overset{s}{\rightarrow} Anc(O)$ et ${}^{*}B(O) \overset{s}{\rightarrow} Anc(O)$ peuvent aussi bien inférer des exécutions avec acquisition partielle de la ressource ; pour résoudre cette ambivalance nous avons

TABLE 6.2 – Appréciation des configurations d'exécution

Lecteur	*Real one*			*Ambulant*		
Configuration	**aucune**	**partielle**	**totale**	**aucune**	**partielle**	**totale**
$B(O) \xrightarrow{s} E(O)$			x			x
$B(O) \xrightarrow{s} E(O)^*$		x			x	
$B(O)^* \xrightarrow{s} E(O)^*$	x			x	x	
$^*B(O) \xrightarrow{s} E(O)^*$		x			x	
$^*B(O) \xrightarrow{s} E(O)$			x			x
$B(O) \xrightarrow{s} Anc(O)$			x		x	x
$^*B(O) \xrightarrow{s} Anc(O)$			x		x	x
$^*B(O) \xrightarrow{s} {}^*E(O)$	Impossible				x	
$B(O) \xrightarrow{s^*} E(O)$	Impossible				x	
$B(O)^* \rightarrow^* E(O)$	Impossible				x	
$B(O)^* \xrightarrow{s} Anc(O)$	Impossible				x	

besoin de procéder à une analyse plus approfondie. Par ailleurs, une exécution sans acquisition de ressource induit la séquence $B(O)^* \xrightarrow{s} E(O)^*$ dans le cas des deux lecteurs. Toutefois, pour le lecteur *Ambulant*, cette configuration peut dénoter aussi une exécution avec acquisition partielle de la ressource ; là aussi une étude approfondie de la séquence s'impose.

A d'autres égards, la vérification devrait aussi explorer les inconsistances résultant lorsque différentes régions (fenêtres), sont en conflit pour le même espace d'écran. Par exemple, le langage *SMIL [90]* établit une priorité[7] pour décider quelle région doit être affichée en premier plan, lorsque plusieurs régions utilisées simultanément, sont en conflit pour le même espace écran. Il en résulte que les présentations d'un voire de plusieurs médias pourraient être masquées durant une partie ou la totalité de leurs exécutions si les régions qui leur sont dédiées sont recouvertes par d'autres régions plus prioritaires, évoluant en même temps. Faisant ce constat, la méthodologie envisagée devrait avoir en plus la capacité de prendre en charge la vérification de ce type de disfonctionnements.

Pour ce faire, nous discutons par la suite d'un nouveau concept appelé "*consistance qualitative*"[20][19] qui étend les concepts précédemment introduits [85][86] à la vérification des conflits de ressources.

Auparavant, nous aurons besoin d'introduire quelques notations.

Notation 5 *Soit R respectivement O une région respectivement, un média de base.*
- On note par $\Omega(R)$ l'ensemble des régions définies de priorité supérieure, qui sont conflituelles avec R pour l'espace écran.
- On note par $\Psi(O)$, l'ensemble des médias ayant une priorité supérieure à celle de O.

Définition 24 ***(Consistance Qualitative).*** *Un présentation multimédia est dite "qualitativement consistante", si le graphe correspondant GR est "temporellement consistant" et pour chaque chemin S dans GR, et pour chaque média O qui apparait dans la séquence S $(Seq(O,S) \neq \varnothing)$, nous vérifions : Soit R la ressource associée au média O.*

Si** un fonctionnement du lecteur Real one, **alors

7. Cette priorité est dénotée par un paramètre *Zindex* associé à la déclaration d'une région.

– ***Si*** *O est un média discret (i.e. image ou text),* ***alors***

1. *Il existe au moins un marquage accessible M dans la séquence* $Seq(O,S)$*, tel que toutes les places ressource* $\{fr(R')/ \quad R' \in \Omega(R)\}$ *sont marquées :* $\exists M \in Seq(O,S), \forall R' \in \Omega(R)$,
$$\exists l = (fr(R'), \{bs_1(R'), .., bs_q(R')\}) \in RAS, \quad M(fr(R')) = 1\,;$$
2. $Seq(O,S) \neq B(O)^* \stackrel{s}{\rightarrow} E(O)^*$.

– ***Si*** *O est un média continu (i.e., animation, vidéo ou audio),* ***alors***

1. ***Si*** *le media n'est pas Audio,* ***alors*** *Toutes les places ressource* $\{fr(R')/ \quad R' \in \Omega(R)\}$ *doivent être marquées quelque soit le marquage accessible dans la séquence* $Seq(O,S)$: $\forall M \in Seq(O,S), \forall R' \in \Omega(R)$,
$$\exists l = (fr(R'), \{bs_1(R'), .., bs_q(R')\}) \in RAS, \quad M(fr(R')) = 1\,;$$
2. $$Seq(O,S) \in \left\{ \begin{array}{l} B(O) \stackrel{s}{\rightarrow} E(O), {}^* B(O) \stackrel{s}{\rightarrow} E(O), \\ B(O) \stackrel{s}{\rightarrow} Anc(O), \ {}^*B(O) \stackrel{s}{\rightarrow} Anc(O) \end{array} \right\}$$

Si *un fonctionnement du lecteur Ambulant,* ***alors***

– ***Si*** *O est un média discret,* ***alors***

1. *Il existe au moins un marquage M dans la séquence* $Seq(O,S)$*, tel que toutes les places centrales des unités impliquées dans les shémas d'allocation des ressources* $\{R' \in \Omega(R)\}$ *sont non marquées :*
$\exists M \in Seq(O,S), \forall R' \in \Omega(R), \exists l = (fr(R'), \{bs_1(R'), .., bs_q(R')\}) \in RAS,$
$\forall u \in units(l), \quad M(pc(u)) = 0\,;$
2. ***Si*** $Seq(O,S) = B(O)^* \stackrel{s}{\rightarrow} E(O)^*$, ***alors*** $Seq(O,S) \subsetneq \ \cup_{O' \in \Psi(O)} \{Seq(O',S)\}$

– ***Si*** *O est un média continu,* ***alors***

1. ***Si*** *le media n'est pas Audio,* ***alors*** *toutes les places centrales des unités impliquées dans les shémas d'allocation des ressources* $\{R' \in \Omega(R)\}$ *sont non marquées, quelque soit le marquage accessible M dans la séquence* $Seq(O,S)$ *:*
$\forall M \in Seq(O,S), \forall R' \in \Omega(R), \exists l = (fr(R'), \{bs_1(R'), .., bs_q(R')\}) \in RAS,$
$\forall u \in units(l), \quad M(pc(u)) = 0\,;$
2. $Seq(O,S) \in \left\{ B(O) \stackrel{s}{\rightarrow} E(O), {}^* B(O) \stackrel{s}{\rightarrow} E(O) \right\}$ ***ou***
$Seq(O,S) \in \left\{ B(O) \stackrel{s}{\rightarrow} Anc(O), \ {}^*B(O) \stackrel{s}{\rightarrow} Anc(O) \right\}$, ***tel que***
$Seq(O,S) \cap \left(\cup_{O' \in \Psi(O)} \{Seq(O',S)\} \right) = \varnothing$

Cette dernière définition énonce qu'une présentation multimédia est "*qualitativement consistante"* si elle permet aux médias discrets d'être visualisés durant une partie de leur temps d'exécution tandis qu'elle oblige les médias continus à être perçus pour la totalité de leurs présentations. Plus clairement, concernant les médias discrets, la première clause permet de contrôler si la région associée est cachée par une autre région (de plus haute priorité), conflictuelle pour le même espace. Formellement, dans le cas d'une modélisation épousant le fonctionnement du lecteur *Real one*, il faut trouver une sous-séquence dans $Seq(O,S)$ (dénotant une portion de l'exécution de O),

telle que les *places ressource* $fr(R')$) modélisant les régions conflictuelles de plus haute priorité, sont toutes marquées (i.e., ces régions sont non utilisées durant cette période de temps). Dans le contexte du lecteur *Ambulant*, l'exploration est plus complexe ; il suffit de trouver un marquage dans la séquence tel que toutes les places centrales des unités utilisant les schémas d'allocation des régions de haute priorité, soient non marquées.

Par ailleurs, la seconde clause permet de traiter la problématique relative aux média conflictuels envers les mêmes ressources (region, canal audio). Donc, nous avons besoin de vérifier que tous les médias discrets sont au moins perçus par l'utilisateur pour une partie de leurs exécutions. Dans un contexte de fonctionnement semblable à celui du lecteur *Real one[81]*, nous avons besoin de vérifier que les séquences d'événements associées à chaque média discret soient différentes de $B(O)^* \xrightarrow{s} E(O)^*$. Par contre, dans un contexte de fonctionnement épousant celui du lecteur *Ambulant* [27], nous devons vérifier en plus que la séquence d'événement $B(O)^* \xrightarrow{s} E(O)^*$ n'est pas totalement incluse dans la concaténation des séquences d'événements associées aux média conflictuels de haute priorité [8]. Concernant les médias continus, nous vérifions les mêmes clauses que pour les médias discrets, mais en assumant que les médias continus doivent être perçus pour la totalité de leurs temps d'exécution. Par conséquent, en se basant sur les interprétations des configurations données précédemment, un média est perçu pour la totalité de sa durée si sa séquence d'événement appartient à
$\left\{ B(O) \xrightarrow{s} E(O), {}^*B(O) \xrightarrow{s} E(O), B(O) \xrightarrow{s} Anc(O),\ {}^*B(O) \xrightarrow{s} Anc(O) \right\}$.

Toute fois, dans le contexte du lecteur *Ambulant,* nous vérifions de plus pour les séquences $B(O) \xrightarrow{s} Anc(O)$ et ${}^*B(O) \xrightarrow{s} Anc(O)$ qu'elles ne chevauchent pas avec aucune autre séquence relative à un média conflictuel de haute priorité.

Au vue de la définition de ce nouveau concept, la vérification de la consistance d'un document multimédia est obtenue en analysant le graphe correspondant. Ainsi, si ce dernier ne respecte pas les clauses de la *Définition 24*, la correction du document est effectuée en identifiant les origines de ces inconsistances en localisant les medias impliquées dans les conflits de ressources. De là, nous pouvons éliminer ces situations conflictuelles soit par rajout de nouvelles régions ou par formattages spatiale et temporelle du document. Cette méthodologie peut être donc utilisée lors de la phase d'édition, dans l'élaboration de documents multimédia qualitativement consistants, en exploitant le graphe dérivé de la spécification *STPTPN*. Par contre, au niveau de la phase de lecture, le lecteur peut procéder à la programmation des scénarios consistants lorsqu'ils existent. Dans le cas où tous les chemins du graphe sont qualitativement inconsistants, le lecteur pourrait alerter l'utilisateur des anomalies (dues aux conflits de ressource), occurant durant la présentation.

A d'autres égards, le graphe d'accessibilité peut aussi être utilisé dans le calcul d'un pattern d'accés adapté[12] et précis qui pourrait être exploité efficacement dans la gestion d'un schéma de préchargement ou d'un mécanisme de délivrance, dans le même esprit que ce que nous avons proposé pour les présentations temporellement consistantes [9][13][22]. Pour ce faire, le schéduler des requêtes doit déterminer les médias discrets qui ne pourront être perçues faute de ressources, et d'éviter ainsi de les quémander. Aussi, le schéduler devrait réquérir seulement les portions des médias continus qui seront perçus par l'utilisateur. Le graphe permet ainsi de déterminer les médias continus concernés par ce traitement spécifique, et les différentes portions à demander.

8. Cela implique de trouver une sous-séquence dans $Seq(O, S)$ qui n'est incluse dans aunce séquence d'événement associée à une media conflictuel de haute priorité.

Plus encore, le graphe pourrait être exploité pour la gestion d'un protocole de délivrance adapté, permettant de garantir une qualité de service continue et optimale pour ce type d'applications, et ce à l'image de ce qui a été proposé précédemment[52][88].

6.4 Exemple

Pour illustrer la méthodologie de vérification présentée auparavant, nous considérons l'exemple du document multimédia *SMIL* présenté dans § *4.4.2*.

La vérification de la consistance temporelle de ce dernier se fait en capturant seulement ses contraintes temporelles et de synchronisation. Cela revient à considérer les *STPTPN* des figures *Figure*.4.4 et *Figure*.4.5, dépourvus de toutes les places ressource. Par exemple dans le cas d'un fonctionnement semblable à celui du lecteur *Real One*[81], nous obtenons à partir de la spécification *STPTPN* correspondante (après application de l'algorithme d'énumération présenté dans le *Chapitre V)*, le graphe de la figure *Figure*.6.1. Dans ceci, nous remarquons que les chemins $0 \longrightarrow 6, 0 \longrightarrow 11$ *et* $0 \longrightarrow 15$ sont *temporellement inconsistants* puisque l'ensemble des événements étiquetant leurs arcs terminaux ne contiennent aucune occurrence de l'événement $E(Aud_2)$. Ces chemins décrivent les scénarios inconsistants induits par l'interaction de l'utilisateur intervenant entre *4* et *5* secondes. Dans ces configurations, le bloc *"par*$_0$*"* commence avant *"*Aud_1*"* empêchant ainsi ce dernier d'achever sa propre exécution [9]. Par conséquent, le média *"*Aud_2*"* n'est pas autorisé à commencer, car *"*Aud_1*"* ne peut se terminer [10].

Les chemins restants menant au noeud 22, décrivent les scénarios consistants de la présentation multimédia. Par exemple, le chemin $0 \longrightarrow 13 \longrightarrow 22$ décrit le cas où l'utilisateur interagit à l'instant *5s*, tandis que le chemin $0 \longrightarrow 24 \longrightarrow 22$ modélise l'exécution dans laquelle l'utilisateur intervient après cette date et avant l'instant *6s*. Le dernier scénario est représenté par le chemin $0 \longrightarrow 30 \longrightarrow 22$, et indique le cas où l'utilisateur n'a pas interagi avec la présentation. Ce comportement est illustré par l'occurrence de l'événement time out $Tout(Anc)$ associé au lien [11].

Malgré que le document est *temporellement inconsistant,* son graphe d'accessibilité peut être utilisé au niveau du lecteur comme un outil de décision pour la programmation exclusive des scénarios consistants. Ceci nécessité d'ignorer tous les chemins inconsistants du graphe, ce qui dans notre cas implique de forcer l'utilisateur à interagir au moins 5 seconds aprés le début du média *"*Vid_1*"*. Ainsi, le lecteur sera amené à interdire toute interaction entre 4 et 5 secondes, mais en autorisant celle qui interviendrait ultérieurement.

En considérant maintenant le même fonctionnement mais avec la spécification des conflits de ressources, nous obtenons le graphe décrit dans la *Figure*.6.2.*a* où les chemins *temporellement inconsistants* ont été élagués. Au fait, nous cherchons à explorer comment se comportent les différents scénarios (prouvés comme *temporellement consistants*), lorsque on intégre en plus la problématique des conflits de ressource. Par conséquent, on peut remarquer que le média Vid_2 est forcé de violer la région R_1 (détenue par Img_1), pour initier sa propre présentation ; Ceci intervient à la date *4s*, et est dénoté par l'occurrence de l'événement de violation active ${}^*B(Vid_2)$ étiquetant l'arc $(1-7)$. Cette configuration implique (quelque soit le scénario considéré), que Img_1 ne pourra

9. Dans le respect de la valeur de synchronisation de son événement de terminaison
10. Dans le respect de la valeur de synchronisation associée à l'événement de début de *"*Aud_2
11. L'occurrence de cet événement dénote que le lien n'est plus visualisé et par conséquent l'utilisateur ne peut interagir à partir de cet instant avec la présentation.

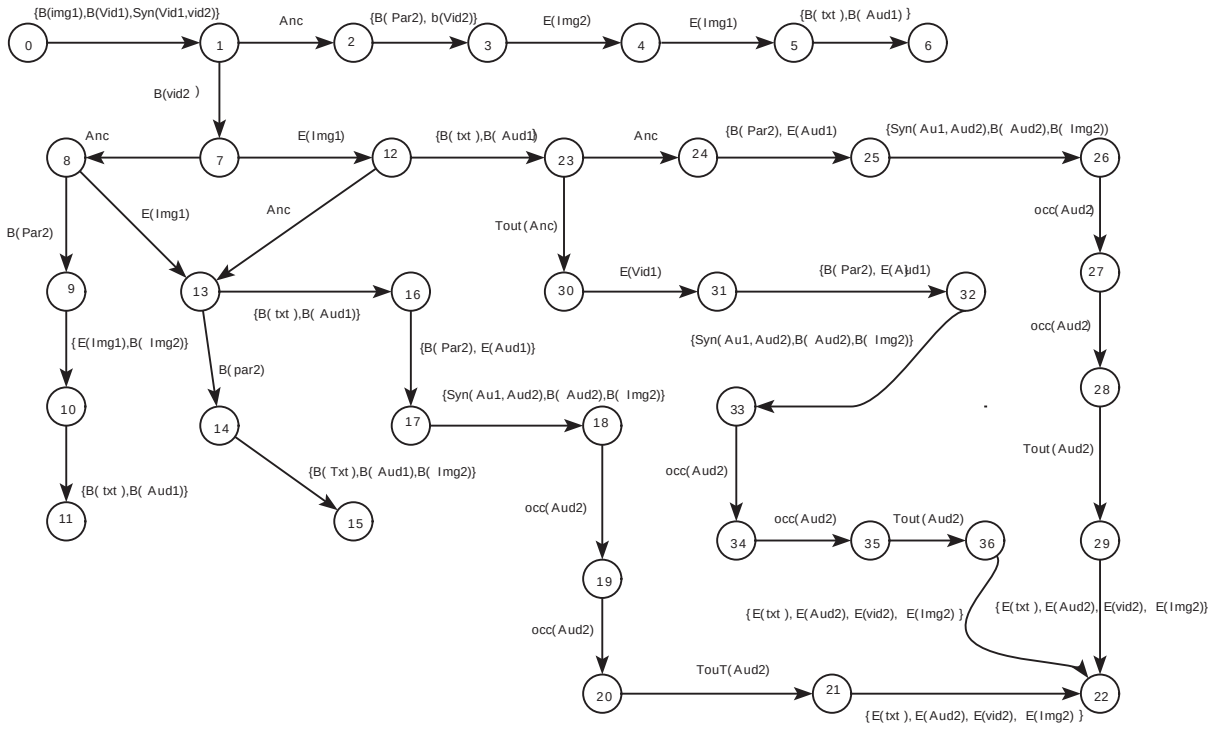

FIGURE 6.1 – Graphe d'accessibilité capturant les contraintes temporelles et de synchronisation de l'exemple de la *Figure.4.3*

être perçu pour au plus la dernière seconde de son exécution. Ce comportement est spécifié par la génération de l'événement de violation passive $E(Img_1)^*$ étiquetant les arcs $(8-13)$ et $(7-12)$.

A d'autres égards, il est à noter que le média Vid_1 (qui a acquis initiallement la région R_2 est en position de la garder jusqu'à sa fin), empêche le média Txt d'être visualisé malgré que Vid_1 est en mesure de libérer la ressource région avant que Txt ne se termine. Ces cas sont illustrés par les scénarios décrits par les chemins $(0 \longrightarrow 30 \longrightarrow 22)$ et $(0 \longrightarrow 24 \longrightarrow 22)$, modélisant les scénarios où Txt est imperceptible pour la durée totale de sa présentation [12]. Par ailleurs, seulement le scénario décrit par le chemin $(0 \longrightarrow 13 \longrightarrow 22)$ permet de percevoir Txt, à condition que l'interaction de l'utilisateur intervient à la date *5s ;* achevant Vid_1 avant que Txt ne commence.

Pour expliciter encore plus l'interet de cette vérification, considérons maintenant la même présentation *SMIL*, mais en assumant le fonctionnement du lecteur *Ambulant*. Au vue de cette dernière hypothèse, l'ordre de priorité adopté est inversé (i.e, les médias Txt, Aud_1, et Img_1 ont la priorité sur respectivement, Vid_1, Aud_2, et Vid_2). De là, le graphe d'accessibilité correspondant à la spécification formelle est présenté en $Figure.6.2.b$. En comparant ce dernier avec celui de la $Figure.6.2.a$, on peut remarquer que le chemin $23 \longrightarrow 24 \longrightarrow 22$ n'existe plus [13]. Plus concrètement, le média Txt qui a la priorité, est en mesure de violer Vid_1 si ce dernier ne se termine pas avant *5s*. Ainsi, Vid_1 sera non perceptible entre *5s* et *8s*, et par conséquent l'utilisateur ne peut interagir avec le lien. Donc, l'occurrence de $^*B(Txt)$ étiquetant l'arc $(12-23)$ dénote la préemption de la ressource R_2 (détenue jusqu'à lors par Vid_1), et ceci se produit seulement dans le scénario décrit par le chemin $0 \longrightarrow 23 \longrightarrow 22$. Par ailleurs, le scénario donné par le chemin $0 \longrightarrow 13 \longrightarrow 22$, décrit le cas où l'utilisateur intervient à l'instant *5s* juste à temps pour achever

12. Ces comportements sont indiqués par l'occurrence des événements de violation passive $B(Txt)^*$ respectivement, $E(Txt)^*$ étiquetant les arcs $(12-23)$ respectivement, $(29-22, 36-22)$.

13. Ce chemin dénote le scénario où l'interaction de l'utilisateur intervient entre 5s et 6s

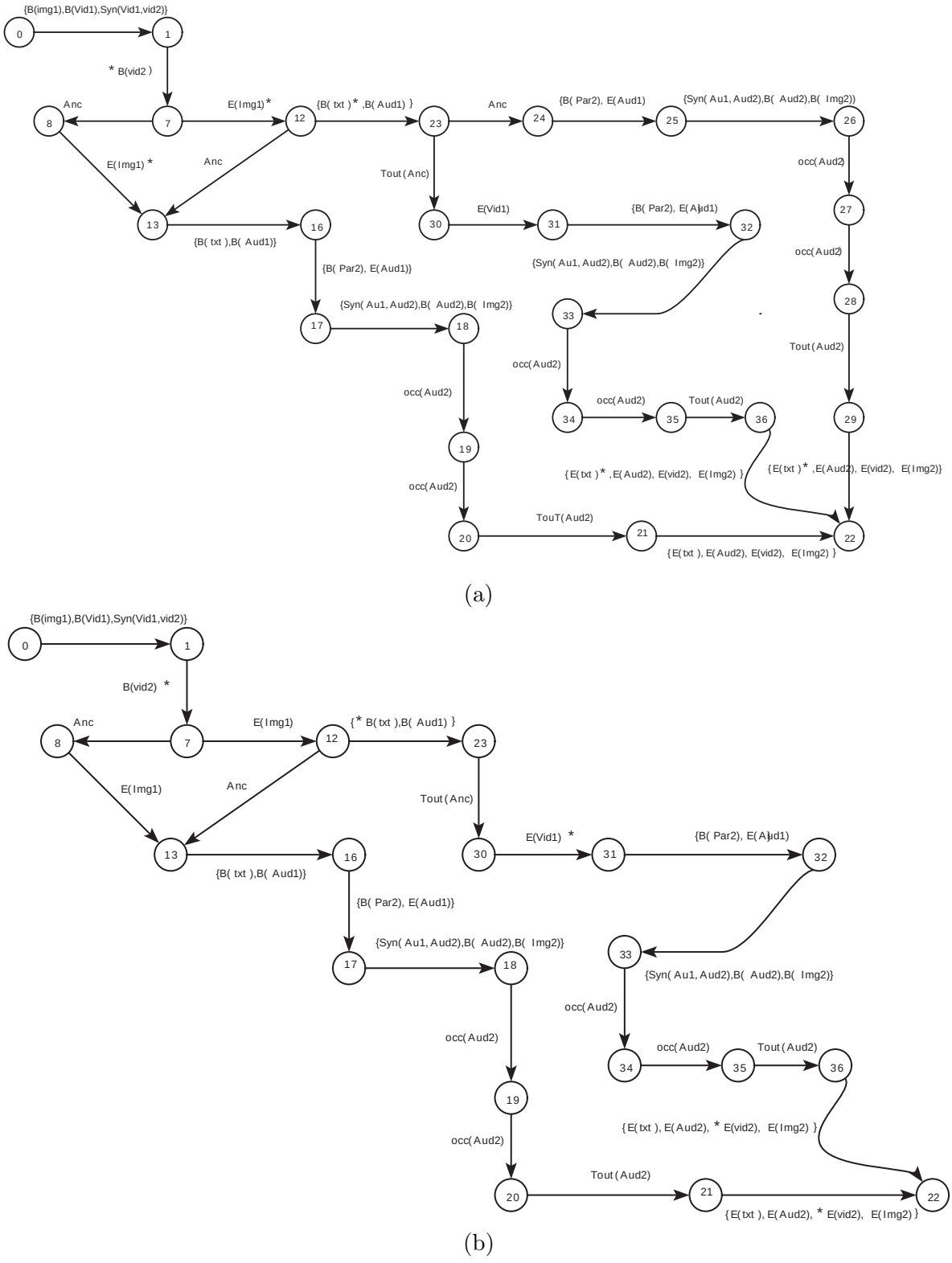

FIGURE 6.2 – Les graphes capturant en plus les contraintes ressources de la présentation multimedia, en adoptant le fonctionnement des deux lecteurs.

Vid_1 avant que Txt ne commence.

De plus, quelque soit le scénario considéré, le média Vid_2 est non perceptible durant la première seconde de sa présentation (entre *4s* et *5s*) puisque la région R_1 est détenue par Img_1 [14]. Toutefois, Vid_1 poura être perceptible à l'instant *5s,* une fois que Img_1 sera achevée. Ceci est dénoté par la génération de l'événement de violation active ${}^*E(Vid_2)$ étiquetant les arcs $(36-22)$ et $(21-22)$.

Remarque : Il est à noter que faisant le constat que les méthodologies de vérification existantes ne traitent pas la problématique des conflits de ressources ; il en résulte que le graphe obtenu, induit des scénarios supplumentaires (chemin $24 \longrightarrow 22$ dans la graphe de la $Figure$.6.1) pouvant fausser la vérification de la consistance.

6.5 Conclusion

Nous avons montré à travers ce chapitre comment exploiter le graphe des classes (obtenu à partir d'une spécification $STPTPN$), pour la vérification de ses propriétés linéaires, et pour le calcul des temps minimimum et maximum de n'importe quel chemin du graphe. Par ailleurs, nous avons développé une méthodologie permettant de vérifier la *consistance qualitative* des documents multimédia. Ce dernier concept a été défini pour étendre les concepts précédemment dédiés à la vérification de la cohérence temporelle, afin qu'ils puissent prendre en charge la détection et le traitement des conflits de ressources dans ces applications.

14. Ceci est illustré par la génération de l'événement de violation passive $B(Vid_2)^*$ étiquetant l'arc $(1-7)$.

Chapitre 7

Conclusion Générale

7.1 Bilan

La vérification et l'analyse des propriétés des systèmes temps réel est une tâche ardue ; d'ailleurs les applications multimédia en sont un exemple concrét du fait des sémantiques complexes qu'elles induisent. Les méthodes formelles ont longtemps été utilisées pour l'analyse de ces systèmes parmi lesquelles celles basés sur les réseaux de Petri. Principalement, les techniques d'analyse basées sur l'exploration de l'espace d'état de ces systèmes sont des solutions exhaustives mais couteuses ; néanmoins elles permettent de répondre indubitablement aux propritétés de "*sûreté*".

Nous avons tenter à travers cette thèse d'apporter un certain nombre de solutions à des problématiques différentes mais néanmoins participant à la proposition d'une méthodologie formelle permettant de modéliser et d'analyser les systèmes temps réel complexes. Les systèmes multimédia étant dans ; nous avons exploré leurs comportements et leurs exigences en termes de temps, de synchronisation, et de ressource. Plus particulièrement, la gestion des ressources fut un point important à explorer dénotant d'une problématique nouvelle qui fallait aborder minutieusement. D'ailleurs, nos investigations ont permi de recenser les besoins en terme de *ressources non critiques* (canal audio, région), et d'explorer leurs schémas d'allocations.

A partir de ces besoins, la proposition d'un modèle appelé $STPTPN$ (*Pre-emptive Time Petri Nets with Synchronizing Transitions*) a été élaborée et déclinée[14][16]. Ce dernier permet d'étendre les $RdPT$ à différents mécanismes dont certains ont été précédemment introduits dans la littérature, tels que :

- la *synchronisation* : permettant le tir simultané et indivisible d'un ensemble de transitions appelé "*rendez-vous*" selon un schéma de synchronisation prédefini ; les stratégies de *Senac et autres* ont été considérés [87] ;
- les *chronomètres* : permettant de modéliser l'arrêt la suspension et la reprise de l'horloge temporelle d'une transition (*chronomètre*). Ces actions sont induites gràce à la sémantique des arcs standards et des arcs inhibteurs [83].

et d'autres nouvellement introduits, tels que :

- *la priorité en temps réel* [11][10] : permettant de résoudre le non déterminisme lors du tir des transitions concurrentes en associant des priorités opérant en temps réel ;
- les *hyperarcs de préemption* [14], et les *arcs de privation de ressource* [20] : permettant de spécifer les schémas d'allocations de *ressources non critiques* ;

– *les arcs de franchissement* : permettant de conditionner le tir d'une transition en fonction de la disponibilité des *ressources non critiques.*

De plus, le raffinement de la notion d'événement (*événement fort, événement de violation active et événement de violation passive*) associé au tir d'une transition a permi d'expliciter les comportements induits par les schémas d'allocation des *ressources non critiques.*

Gràce à ce cadre formel, il est devenu possible de capturer et de modéliser avec plus de fidélité et de précision les comportements les plus complexes induits par les systèmes temps réel et en particulier des applications multimédia. D'ailleurs, nous avons montré comment notre modèle pouvait facilement spécifier de manière naturelle et implicite les exigences des présentations multimédia *SMIL*, et ce au contraire des autres méthodologies existantes [41][47][84][71].

Toutefois, le pouvoir d'expression de ce modèle ne pourrait être exploitable si aucune approche d'analyse de ses propriétés n'aurait été pourvue. Pour ce faire, nous avons développé deux approches pour la construction du graphe des classes d'un $STPTPN$.

– Le première approche permet de construire le graphe des classes exact d'un $STPTPN$. Dans cela, nous avons formulé l'expression d'une classe E au moyen du couple (M, D); où M est le marquage accessible et D est un système de contraintes utilisant les variables transitions. Nous avons montré que D s'écrivait dans ce cas par la conjonction de deux sous-systèmes $\widetilde{D} \wedge \widehat{D}$ tel que $\widetilde{D}$ est de forme *DBM(Difference Bound Matrix),* alors que $\widehat{D}$ regroupe toutes les contraintes de forme générale dite *polyédrique.* Cependant, les tests de consistance et de franchissement nécessitent l'extension du système D aux contraintes relatives aux rendez-vous fortement sensibilisés. La complexité du calcul d'une classe est d'ordre exponentiel facteur du nombre de variables présentes dans le système étendu ; handicapant sérieusement la construction du graphe lorsque ce dernier devient de taille importante. Le graphe ainsi construit préserve toutes les propriétés linéaires du modèle.
– Afin de réduire la complexité de calcul, nous avons exploré dans la seconde approche une technique implémentant la sur-approximation DBM du système D. Cette dernière consiste à éliminer systématiquement toutes les contraintes polyédriques $\widehat{D}$. Ceci permet de réduire la complexité de calcul d'une classe ainsi que des différents tests. Cependant, le relâchement de contraintes non DBM peut induire des séquences de franchissements additifs non accessibles dans le graphe exact. Le graphe ainsi construit est dit par conséquent sur-approximé, tel que chaque classe sur-approximée $\widetilde{E}$ est donnée par le système $\widetilde{D}$, et présentée sous forme d'une matrice des bornes. Les différents tests et le calcul de toute classe accessible nécessitent le calcul d'une autre matrice des bornes notée DR représentant les contraintes relatives aux rendez-vous sensibilisés. Pour ce faire, la complexité de calcul de toute classe $\widetilde{E}$ est polynomiale d'ordre deux estimée à $o(l^2 + l \times m + m)$ évitant l'utilisation de l'algorithme de *Floyd/Warshall* dont le coût est estimé au meilleur des cas à $o((m + l)^3)$, où m et l représentent respectivement, le nombre de transitions sensibilisées et le nombre de rendez-vous fortement sensibilisés. Le graphe ainsi calculé est *exact* si le modèle ne contient pas d'arcs inhibiteurs, et peut être *sur-approximé* dans le cas contraire. Dans ce dernier cas, un sous ensemble des propriétés linéaires sont préservées, notamment celles inhérentes à la *sûreté.* Par ailleurs, nous avons développé une technique permettant de déterminer à partir du graphe obtenu une sur-approximation des temps minimaux et maximaux de n'importe quelle sous séquence de franchissement. Ceci peut être exploité pour l'estimation quantitative

des propriétés temps réel du modèle.

Enfin, tout au long de notre thèse, nous avons motivé nos différentes propositions par leur application dans la spécification des contraintes des documents hypermédias et multimédia. Pour ce faire, nous avons exploré et exploité les mécanismes définis pour ces besoins en se basant sur les fonctionnements les plus répandus ; pour cet effet, les fonctionnements des lecteurs *Real one* [81] et *Ambulant* [27] ont été considérés.

Une méthodologie d'analyse [6][7][20][19] a été développée permettant de détecter et de localiser en plus des erreurs dues aux inconsistances temporelles, celles induites par les conflits de *ressources*. Le concept de "*consistance temporelle*" précédemment introduit dans [85][86] a été étendu par la définition du paradigme de "*consistance qualitative*" [20][19]. Ce dernier permet de vérifier un panel d'erreurs plus large et d'introduire la notion de qualité de perception vis-à-vis de l'utilisateur.

7.2 Perspectives

Plusieurs travaux futures inhérents à cette thèse sont à envisager :

- L'implémentation de l'outil d'édition graphique d'un $STPTPN$ et mise en oeuvre de l'approche proposée ; une estimation des performances de la technique est en cours.
- Exploitation de notre technique pour la réduction de la taille des graphes [1] des classes en supprimant les comportements induits par la sémantique d'*Interleaving* ; Comparaison avec les technique de réduction par ordre pariel[66].
- Adaptation de l'approche d'énumération pour la génération de l'automate temporisé équivalent, voire du graphe des classes atomiques préservant les propriétés de branchement [38][97].
- Etudes de cas de systèmes temps réels complexes incluant entre autres les applications multimédia.

1. En exploitant le regroupement par rendez-vous.

Bibliographie

[1] H.Boucheneb, A.Abdelli, and G.Berthelot. L'automate temporisé d'une spécification TC-LOTOS. in Proc du 3iéme coloque Africain sur la recherche informatique CARI'98 Dakar Senegal - INRIA ORASTROM.

[2] A.Abdelli. Specifying a real time system using the ATN model. Proceedings of the fourth International symposium on programming and systems ISPS'99 Algiers October 1999.

[3] A.Abdelli. Constructing the contracted accessibility graph of real time systems modelled by a language based on timed Petri nets. In Proc of the first International symposium on Software and systems Constantine Algeria I3S2001 - Feb 2001.

[4] A.Abdelli, Y.Bouabdellah, M.Dahmani.Conception d'une plate forme de simulation distribuée pour un langage à base de réseaux de Petri temporels. In proc of SNIB04, the fourth national symposium on informatics- Biskra Algeria may 2004.

[5] A.Abdelli Une méthode formelle pour la vérification de la consistance temporelle et la gestion prédictive de la qualité de service pour la présentation des documents SMIL. In proc the second Symposium of Sciences of Electronic, Technologies and Information Telecommunications- Sousse Tunisie- Mars 2004 ISBN : 9973-41902-2.

[6] A.Abdelli and M.Daoudi "Towards a SMIL document analysis using an algebraic time net," 5th Pacific Rim Confe on Multimedia, Tokyo, Japan, Nov-30- PCM 2004 : LNCS – Vol 3333/2004.

[7] A.Abdelli. and B.Laichi. A formal approach for SMIL1.0 consistent presentation. In proc of 2nd IEEE ICICT'2004 International conference on information and communication technologies - Cairo Egypt December 2004.

[8] A.Abdelli. Managing a Pre fetching Scheduling into a SMIL presentation. In proc of ACIT'04 international Arab conference on information and technology- Constantine Algeria December 2004.

[9] A.Abdelli and N.badache. An adaptive proxy pre-fetch scheme for consistent SMIL presentation. In proc of ACIT05 international Arab conference on information and technology-Amman Jordan December 2005.

[10] A.Abdelli and M.Daoudi. Un nouveau mécanisme pour la résolution du non déterminisme dans les réseaux de Petri temporels. In proc of the 3^{rd} international symposium on telecommunication and information technology- Sousse Tunisia- March 2005.

[11] A.Abdelli. Les réseaux de Petri à priorité temporelle : Un nouveau modèle pour la modélisation et la simulation déterministes des systèmes temps réel. Second Springer school on cybernetics INI -LRIA, 5-7 june 2005 Algiers Algeria.

[12] A.Abdelli and N.badache. A Semantic based pre-fetch scheme for SMIL presentation Proxy-delivery 12th IEEE Multimedia modelling conference, Beigin China 4-6 Jan. 2006 Page(s) :8 pp.

[13] A.Abdelli and N.badache. A Proxy Pre-fetch Scheme for Consistent SMIL Presentation Delivery. In proc of IEEE ACS/AICCSA Dubai(EAU) March 2006 Page(s) :968 - 975.

[14] A.Abdelli and N.Badache. Synchronized Transitions Preemptive Time Petri Nets : A new model towards specifying multimedia requirements ; In proc of IEEE international conference ACS/AICCSA, Dubai March 8, 2006 Page(s) :17-24.

[15] A.Abdelli and N.badache. Une Approche pour la Construction de l'Espace d'Etat d'un Réseau de Petri à Flux. In proc of 7Th MCSEAI International Conference on software engineering and Artificial intelligence. Agadir Morocco 7-9 December 2006.

[16] A.Abdelli and N.badache. Toward Specifying Multimedia Requirements Using a New Time Petri Net Model. CIT :International Journal of computing and Information Technology, 2007 ISSN : 1330-1136 VOL 15-3.

[17] A.Abdelli. Timed State Space Computation of Weakly Synchronous Systems. Internal Report 2007. Submitted to Real Time Systems Journal Springer Verlag (in progress).

[18] A.Abdelli and N.badache. Towards building the state class graph of the TSPN model. Internal Report 2006. Submitted to Journal Fundamenta Informaticae -IOS press (accepted).

[19] A.Abdelli. Dealing with resource requirement in multimedia document consistency. in BCS EWIC series of the first workshop VECoS Algiers May 2007.

[20] A.Abdelli. Towards Qualitative Modeling and Consistency Verification of Multimedia Documents". Internal Report 2007. Submitted to Fundamenta Informaticae- IOS press (in progress).

[21] A.Abdelli. Efficient state space computation and time analysis of non suspensive Preemptive Systems. Internal report 2007.

[22] A.Abdelli and N.Badache Efficient bandwidth and buffer management for multimedia data download. In proc IEEE computer soceity of FGCN'07 (Future Generation Communication and Networking) South Korea Dec 6-8 2007 (To appear).

[23] J.Allen. Maintaining knowledge about temporal interval. Communication of ACM, 1983, 26(11) :832-843,

[24] R.Alur, and, L. Dill. A Theory of Timed Automata. Theoretical Computer Science, 126 :183 :235, 1994.

[25] R.Alur, C.Courcoubetis, N.Halbwachs, T.A.Henzinger, P.-H.Ho, X.Nicollin, A.Olivero, J.Sifakis, and S.Yovine. The algorithmic analysis of hybrid systems. Theoretical Computer Science, 138 :3-34, 1995.

[26] T. Agerwala. A complete model for representing the coordination of asynchronous processes. Technical report- John Hopkins University- Baltimore Maryland 1974.

[27] Ambulant Player http ://ambulantplayer.org.

[28] Avis, D., K. Fukuda and S. Picozzi, On canonical representations of convex polyhedra. in : A. M. Cohen, X.-S. Gao and N. Takayama, editors, Mathematical Software, Proceedings of the First International Congress of Mathematical Software (2002), pp. 350–360.

[29] Y.Atamna. Réseaux de Petri temporisés stochastiques classiques et bien formés : Définition, Analyse et applications aux systèmes distribués temps réel. PhD. Thesis, Univ Paul Sabatier de Toulouse (France),1994.

[30] B. Berthomieu. " La mèthode des classes d'ètats pour l'analyse des rèseaux temporels. Mise en œuvre, extension à la multi sensibilisation ". Colloque Francophone sur la modèlisation des systèmes rèactifs (MSR'2001) Toulouse (France) Oct 2001.

[31] Bernard Berthomieu , Didier Lime, Olivier Henri Roux, François Vernadat. Problèmes d'accessibilitè et espaces d'ètats abstraits des rèseaux de Petri temporels chronometers, RS - JESA ? 39/2005. MSR ?05, pages 223- 238.

[32] B. Berthomieu, and M. Diaz. Modeling and verification of time dependant systems using Time Petri Nets. IEEE Transactions on Software Engineering, 17(3) :(259-273), March 1991.

[33] Bernard Berthomieu, Didier Lime, Olivier H. Roux, François Vernadat : Reachability Problems and Abstract State Spaces for Time Petri Nets with Stopwatches. Discrete Event Dynamic Systems 17(2) : 133-158 (2007).

[34] G.Blakowski, R.Steinmetz. A media synchronisation survey : Reference model, specification, and case studies. IEEE JSAC, 14(1) :5-35 jan 1996

[35] H. Boucheneb, and G. Berthelot. "Toward a simplified building of Time Petri Nets reachability graph". In Proc of the 5th (PNPM93), pages 46-55, Toulouse, France, October 1993. SITEF and LAAS-CNRS, IEEE Computer Society Press.

[36] H. Boucheneb and G. Berthelot. "composition des rèseaux de Petri temporisès". Revue Hermes- TSI Volume 17, n°6 , juin 1998.

[37] H. Boucheneb and J. Mullins : Analyse des rèseaux de Petri temporels : Calcul des classes en $O(n^2)$ et des temps de chemin en $O(m \times n)$; Revue Techniques et Sciences Informatiques, No 4/2003.

[38] H. Boucheneb, R. Hadjidj : CTL* model-checking of Time Petri Nets ; Journal of Theoretical Computer Science, Vol 353/1-3 pp 208-227, November 2005.

[39] Marc Boyer : Translation from timed Petri nets with intervals on transitions to intervals on places (with urgency) ; Electr. Notes Theor. Comput. Sci. 65(6) :(2002).

[40] M. Boyer, and M. Diaz. "Non equivalence between time Petri nets and time stream Petri nets". In Peter Bucholz and Manuel Silva, editors, Proceedings of 9th (PNPM'99), pages 198-207, Zaragoza, Spain, September 1999. IEEE Computer Society.

[41] M. Boyer. Contribution à la modèlisation des systèmes à temps contraints et application au multimèdia. PHD Thesis- LAAS Toulouse III -2001.

[42] G. Bucci, A. Fedeli, L. Sassoli, and E.Vicario. Timed State Space Analysis of Real-Time Preemptive Systems. IEEE Transaction on Software Engeneering, Vol 30, No. 2, Feb 2004.

[43] G. Bucci, L. Sassoli, E. Vicario. ORIS : a tool for state-space analysis of real-time preemptive systems, Proc. QEST 2004, Enschede (Olanda), September 2004, pp. 70- 79.

[44] J.P.Courtiat, M.Diaz. R.C De Oliveira P.Senac. Proving temporal consistency in a new multimedia synchronisation model. In proc of ACM Multimedia- Boston Nov 96.

[45] Jean-Pierre Courtiat, Luiz F. Rust da Costa Carmo, Roberto C. de Oliveira : A General-Purpose Multimedia Synchronization Mechanism Based on Casual Relations. IEEE Journal on Selected Areas in Communications 14(1) : 185-195 (1996)

[46] F. Cassez and K.G. Larsen. The Impressive Power of Stopwatches. LNCS, vol. 1877, pp. 138-152, Aug. 2000.

[47] J.P.Courtiat : Providing consistent SMIL2.0 documents ; ICME'2002, Suisse, 26-29 Août 2002, 4p.

[48] B. Berthomieu and F. Vernadat "State class construction for branching analysis of Time Petri Nets". In Proc. Tools and Algorithms for the construction and Analysis of systems, Warsaw, Poland, Springer LNCS 2619, pages 442-457,2003.

[49] J. P. Courtiat C.A.S. Santos, C. Lohr, B.Outtaj. Experience with RT-LOTOS, a temporal extension of the LOTOS formal description technique. In computer Communication 23 July 2000.

[50] A. Cerone and A. Maggiolo-Schettini. "Time-base expressivity of time petri nets for system specification". Theoretical Computer Science, 216(1-2) :1-53, March 1999.

[51] J.P. Courtiat, M. Diaz, R. Cruz, P. Sènac, Formal Models for the Description of Time Behaviors of Multimedia and Hypermedia Distributed Systems, Computer Communications, 1996.

[52] M. Diaz, André Lozes, Christophe Chassot, Paul D. Amer, Partial order connections : a new concept for high speed and mu ltimedia services and protocols, Ann. Tèlècommun., 49, n° 5-6, 1994.September 2001.

[53] M. Diaz, and P. Senac. Time Stream Petri Nets : A model for timed multimedia information. In Proc of the 15th ICATPN, vol 815 of LNCS, pages 219-238, Zaragosse, Spain, June 1994.

[54] Dill, D.L. Timing assumptions and verification of finite-state concurrent systems. In Workshop Automatic Verification Methods for Finite-State Systems. Volume 407. (1989) 197-212

[55] G. Gardey, O.H. Roux and, O.F.Roux. "A zone based method for computing the state space of time Petri Nets". In formal modelling and analysis of timed systems (FORMATS'03) Vol LNCS 2791, Springer Sept 2003.

[56] GRiNS http ://www.oratrix.com/grins/

[57] T.A Henzinger, X.Nicollin, J. Sifakis, and S.Yovine. " Symbolic model-checking for real time systems". Dans proc. 7eme LICS, pages 394-406. IEEE Computer Society Press 1992.

[58] ISO/ IEC IS 10744. Hypermedia/Time based Document structuring Language Hy Time. 1992

[59] S. Fischer. Implementation of multimedia systems based on real time extension of ESTELLEs. In Formal description Techniques VIII FORTE /PST V'96 Kaiserslautern Germany, Oct 1996

[60] M. Hack. PETRI nets language . Computation Structures Group Memo 127, MIT 1975.

[61] M.Haindl. A new Multimedia synchronisation model. IEEE Journal on selected area in communications, 14(1) :73-88 jan 1996

[62] R. Janicki and M.Koutney On causality semantics of nets with priority. Fundamenta Informaticae 38(3)S 233-255, 1999.

[63] K.Jensen. Coloured Petri Nets : Basic concepts, Analysis Methods And Pratical use. Volume 1 et 2 , EATCS Monographs on Theoritical Computer Science, Springer-Verlag 1982

[64] Muriel Jourdan, Nabil Layaïda, Cécile Roisin, Loay Sabry-Ismaïl, Laurent, and Tardif. Madeus. Authoring Environment for Interactive Multimedia Documents. ACM Multimedia 1998 : 267-272.

[65] K. Khansa, J. Denat, and S. Collart-Dutilleul. "P-Time petri nets for manufacturing systems". In Proc. Of WODES'96, pages 94-102, Edimburgh, UK, 1996.

[66] J.Lilius. "Efficient state space Search for time Petri nets". In MFCS Workshop on concurrency '98 Vol 18 of Electronic Notes in Theoretical Computer Science. Elsevier,1999

[67] D.Lime, and O.H. Roux. State class timed automaton of a time Petri net. In 10th International workshop on Petri Nets and Performance Models. (PNPM'03) IEEE computer society, september 2003.

[68] D.Lime, and O.H.Roux. Expressiveness and analysis of scheduling extended time Petri nets. In 5th IFAC International Conference on Fieldbus Systems and their Applications, (FET'03), Elsevier Science, July, 2003.

[69] Lintian Qiao, and Klara Nahrstedt. Lip Synchronization within an Adaptive VOD System. In Proc of SPIE International Conference on Multimedia Computing and Networking (MMCN'97), pp. 170-181, San Jose, California, February, 1997.

[70] T. Little, and A.Ghafoor. Synchronisation and storage models for multimedia objects. IEEE JSAC, 8(3) :413-427, March 1990.

[71] Nabil Layaïda, Loay Sabry-Ismaïl, and Cécile Roisin. Dealing with Uncertain Durations in Synchronized Multimedia Presentations. Multimedia Tools Appl. 18(3) : 213-231 (2002).

[72] Morgan Magnin, Didier Lime, Olivier H. Roux : An Efficient Method for Computing Exact State Space of Petri Nets With Stopwatches. Electr. Notes Theor. Comput. Sci. 144(3) : 59-77 (2006).

[73] M. Menasche and B. Berthomieu. Une approche par énumération pour l'analyse des réseaux de Petri temporels. In Actes de la confèrence IFIP'83, pages 71-77, 1983.

[74] P. Merlin. A study of the recoverability of computer system. PHD Thesis Dep. Computer. Science, Univ. California, Irvine, 1974.

[75] MHEG : Information technology : coded representation of multimedia and hypermedia Objects, DIS ISO/ IEC13522-1, 1994

[76] R Milner. A calculus of communicating systems. LNCS springer Verlag Vol 92 1980.

[77] ORIS TOOL : http ://www.stlab.dsi.unifi.it/oris/index.html.

[78] C. A. Petri : Fundamentals of a Theory of Asynchronous Information Flow. IFIP Congress 1962 : 386-390

[79] J.L. Peterson. Petri Nets. ACM Computing Surveys. Vol. 9, no. 3,Sept. 1977.

[80] C. Ramchandani. Analysis of Asynchronous Concurrent Systems by Timed Petri Nets. PhD thesis, MIT, Cambridge, Feb 1974.

[81] Real Player Home page Url : http ://real.com.

[82] ROMEO TOOL http ://romeo.rts-software.org/.

[83] O. H. Roux, D. Lime. Time Petri nets with inhibitors Hyperarcs, formal semantics and state Space computation. In proc ICATPN 2004 LNCS Bologna, Italy.

[84] P.N.M. Sampaio. Conception formelle des documents multimédias interactifs : Une approche s'appuyant sur RT-LOTOS. Thèse de Doctorat - laboratoire LAAS- Université P.H.sabatier Toulouse France.

[85] C.A.S.Santos, J.P. Courtiat, P. Saqui-sannes. A disign methodology for the formal specification and verification of hypermedia documents. In Proc of FORTE/ PSTV'98 Paris France, Nov1998. Chapman&Hall.

[86] P.N.M.Sampaio, C.A Sontos, and J. P. Courtiat. Using formal method to verify the temporal semantics of SMIL documents. In Proc. Conference of Software : Theory and Practice, IFIP World Congress, Beijing, Chine, August 21 - 25 2000.

[87] P.Senac. Modeling logical and temporal synchronisations. In IEEE JSAC V14, n°1 Jan 1996 pp.84-103.

[88] P.Sénac, E.Exposito, and M.Diaz. Towards a New Generation of Generic Transport Protocols. Lecture Notes in Computer Science 2170, Springer, Eds. S.Palazzo.

[89] J.Sifakis. Use of Petri nets for performance evaluation. In E.Beilner and E.Gelenbe, Editors, Measuring Modelling and Evaluating Computer Systems, pages 75-93, North-Holland, 1977.

[90] SMIL 2.0 : W3C Recommendations, SMIL2.0 Specification. URL :http ://www.w3.org/TR/SMIL20.

[91] TINA Tool http ://www.laas.fr/tina/.

[92] S. Tripakis, and S.Yovine. Analysis of timed Systems based on time abstracting bisimulations. Formal method in system design Kluwer Academic Publishers 2001.

[93] Toussaint, J., Simonot-Lion, F., Thomesse, J.P. : Time constraint verification methods based on time Petri nets. In : 6th Workshop on Future Trends in Distributed Computing Systems (FTDCS'97), Tunis, Tunisia (1997) 262–267

[94] E.Vicario. Static Analysis and Dynamic Steering of Time-Dependent Systems. IEEE Trans. Software Eng. 27(8) : 728-748 (2001) .

[95] B. Walter. Timed net for modelling and analysing protocols with time. In Proceedings of the IFIP Conference on Protocol Specification Testing and Verification, North Holland, 1983.

[96] Yahiatene Dahbia "Implémentation et validation d'une nouvelle approche de construction de graphe des classes des réseaux de Petri temporels étendus aux chronomètres" Projet de fin d'étude pour l'obtention du diplôme d'ingénieur- N°ordre : 128/2007 soutenu au Département d'Informatique-université USTHB en Juillet 2007.

[97] T. Yoneda, and K. H. Ryuba. "CTL Model checking of time Petri Net Using Geometric Regions". IEEE Trans. On information and systems, Volume E99-D, n°3, 1998 1-10.

[98] S.Yovine. Kronos : A verification tool for real time systems. International Journal of soft tools for tech transfer, 1(1), 1997.

Chapitre 8

Annexe A : Preuves

Nous présentons ci-après les preuves des propositions et théorèmes énoncées dans cette thèse.

8.1 Preuve de la Proposition 1

- Cas de $DR_n[\bullet, r]$: Pour prouver les formules de calcul, il suffit de remplacer $DR_n[\bullet, r]$ par son expression formelle de la $Définition.20$.

$DR_n[\bullet, r] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{Up_n(r)\}$.

Ensuite, nous remplaçons $Up_n(r)$ par son expression formelle de la $Définition.11$, en énumérant tous les cas possibles selon le type de synchronisation associé au rendez-vous r. Par exemple, si $typ(r) \in \{Or, Wand, Wmas\}$ on obtient :

$DR_n[\bullet, r] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \underset{\forall t \in Trans(r)}{MAX} \{y_n(t)\} = \underset{\forall t \in Trans(r)}{MAX} \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{y_n(t)\}$

$DR_n[\bullet, r] := \underset{\forall t \in Trans(r)}{MAX} \left\{ \widetilde{D}_n[\bullet, t] \right\}$

- Cas de $DR_n[r, \bullet]$: Ces formules sont démontrables de la même manière que précédemment.

$DR_n[r, \bullet] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{-Low_n(r)\}$.

De même, nous remplaçons $Low_n(r)$ par son expression formelle en énumérant tous les cas possibles. Par exemple, si $typ(r) \in \{And, Wand, Amas, Async\}$, on obtient :

$DR_n[r, \bullet] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} - \underset{\forall t \in Trans(r)}{MAX} \{x_n(t)\} = \underset{\forall t \in Trans(r)}{MIN} \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{-x_n(t)\}$.

$DR_n[r, \bullet] := \underset{\forall t \in Trans(r)}{MIN} \left\{ \widetilde{D}_n[t, \bullet] \right\}$

- Cas de $DR_n[r, r']$: Nous remplaçons $DR_n[r, r']$ par son expression formelle.

$$DR_n[r, r'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \begin{cases} MIN \begin{cases} Low_n(r') - Low_n(r) \\ Up_n(r') - Low_n(r) \end{cases} & \text{si } r' \in \Phi_n(r) \\ Up_n(r') - Low_n(r) & \text{sinon} \end{cases}$$

$$DR_n[r, r'] := \begin{cases} MIN \begin{cases} \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{Low_n(r') - Low_n(r)\} \\ \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{Up_n(r') - Low_n(r)\} \end{cases} & \text{si } r' \in \Phi_n(r) \\ \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{Up_n(r') - Low_n(r)\} & \text{sinon} \end{cases}$$

Nous procédons ensuite au remplacement de $Low_n(r)$ par son expression formelle en énumérant

tous les cas possibles selon le type de synchronisation du rendez-vous r. Par exemple, si $typ(r) \in \{Async, And, Wand, Amas\}$, nous obtenons :

$$DR_n[r,r'] := \begin{cases} MIN \begin{cases} \underset{\forall e_n \in \widetilde{E}_n}{MAX} & \left\{ Low_n(r') - \underset{\forall t \in Trans(r)}{MAX} \{x_n(t)\} \right\} \\ \underset{\forall e_n \in \widetilde{E}_n}{MAX} & \left\{ Up_n(r') - \underset{\forall t \in Trans(r)}{MAX} \{x_n(t)\} \right\} \end{cases} & \text{si } r' \in \Phi_n(r) \\ \underset{\forall e_n \in \widetilde{E}_n}{MAX} & \left\{ Up_n(r') - \underset{\forall t \in Trans(r)}{MAX} \{x_n(t)\} \right\} & \text{sinon} \end{cases}$$

De là, on peut écrire :

$$DR_n[r,r'] := \underset{\forall t \in Trans(r)}{MIN} \begin{cases} MIN \begin{cases} \underset{\forall e_n \in \widetilde{E}_n}{MAX} & \{Low_n(r') - \{x_n(t)\} \\ \underset{\forall e_n \in \widetilde{E}_n}{MAX} & \{Up_n(r') - \{x_n(t)\} \end{cases} & \text{si } r' \in \Phi_n(r) \\ \underset{\forall e_n \in \widetilde{E}_n}{MAX} & \{Up_n(r') - x_n(t)\} & \text{sinon} \end{cases}$$

Nous posons ensuite :

$$\beta_n[t,r'](r) := \begin{cases} MIN \begin{cases} \underset{\forall e_n \in \widetilde{E}_n}{MAX} & \{Low_n(r') - \{x_n(t)\} \\ \underset{\forall e_n \in \widetilde{E}_n}{MAX} & \{Up_n(r') - \{x_n(t)\} \end{cases} & \text{si } r' \in \Phi_n(r) \\ \underset{\forall e_n \in \widetilde{E}_n}{MAX} & \{Up_n(r') - x_n(t)\} & \text{sinon} \end{cases}$$

Et nous obtenons ainsi les formules de $DR_n[r,r']$.

- Cas de $\beta_n[t,r'](r)$.

Pour déterminer les formules de $\beta_n[t,r'](r)$ nous devons remplacer $Low_n(r')$ et $Up_n(r')$ par leurs expressions formelles respectives.

Par exemple, en supposant que $typ(r') = And$; nous obtenons :

$$\beta_n[t,r'](r) := \begin{cases} MIN \begin{cases} \underset{\forall t' \in Trans(r') \forall e_n \in \widetilde{E}_n}{MAX} & \{x_n(t') - \{x_n(t)\} \\ \underset{\forall t' \in Trans(r') \forall e_n \in \widetilde{E}_n}{MIN} & \{y_n(t') - \{x_n(t)\} \end{cases} & \text{si } r' \in \Phi_n(r) \\ \underset{\forall t' \in Trans(r') \forall e_n \in \widetilde{E}_n}{MIN} & \{y_n(t') - x_n(t)\} & \text{sinon} \end{cases}$$

D'après la *Définition.14,* nous pouvons écrire :

$x_n(t') := MAX\,(0, \quad y_n(t') + \alpha(t'))$ avec, $\alpha(t') = EFT(t') - LFT(t')$

En effectuant le remplacement dans l'expression de $\beta_n[t,r'](r)$, nous obtenons :

$$\beta_n[t,r'](r) := \begin{cases} MIN \begin{cases} \underset{\forall t' \in Trans(r') \forall e_n \in \widetilde{E}_n}{MAX} & MAX MAX \begin{cases} -x_n(t) \\ y_n(t') - x_n(t) + \alpha(t') \end{cases} \\ & \underset{\forall t' \in Trans(r') \forall e_n \in \widetilde{E}_n}{MIN} \; MAX \{y_n(t') - \{x_n(t)\} \end{cases} & \text{si } r' \in \Phi_n(r) \\ \underset{\forall t' \in Trans(r') \forall e_n \in \widetilde{E}_n}{MIN} & MAX \{y_n(t') - x_n(t)\} & \text{sinon} \end{cases}$$

$$\beta_n[t,r'](r) := \begin{cases} MIN \begin{cases} \underset{\forall t' \in Trans(r')}{MAX} MAX \begin{cases} \widetilde{D}_n[t,\bullet] \\ \widetilde{D}_n[t,t'] + \alpha(t') \end{cases} & \text{si } r' \in \Phi_n(r) \\ \underset{\forall t' \in Trans(r')}{MIN} \left\{ \widetilde{D}_n[t,t'] \right\} \end{cases} \\ \underset{\forall t' \in Trans(r')}{MIN} \left\{ \widetilde{D}_n[t,t'] \right\} & \text{sinon} \end{cases}$$

Les autres formules de $\beta_n[t,r'](r)$ se démontrent de la même manière.

8.2 Preuve du Théorème 5

La condition du *Théorème.5* peut être formulée par : $\underset{\forall r \in Senable_n}{MIN} \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{Up_n(r)\} \prec 0$. Cette condition dénote que si l'état e_n accessible dans $\widetilde{E}_n$ qui maximize $\underset{\forall r \in Senable_n}{MIN} \{Up_n(r)\}$ de telle sorte que $\underset{\forall r \in Senable_n}{MIN} \{Up_n(r)\} \prec 0$ soit satisfaite *(e_n est temporellement inconsistant)*, alors tous les états accessibles dans $\widetilde{E}_n$ sont aussi *temporellement inconsistants* puisque ils vérifient tous la *Définition 12*.

8.3 Preuve du Théorème 6

Prouvons la contraposée : r_n est non franchissable ssi, $\exists r \in Senable_{n-1}, DR_{n-1}[r_n, r] \prec 0$, ou $\exists r \in \Phi_{n-1}(r_n), \quad DR_{n-1}[r_n, r] \preceq 0$. Cette condition peut être formulée comme suit : $\exists r \in Senable_{n-1} \underset{\forall e_{n-1} \in \widetilde{E}_{n-1}}{MAX} \{Up_{n-1}(r) - Low_{n-1}(r_n)\} \prec 0 \quad \vee$
$\exists r \in \Phi_{n-1}(r_n), \underset{\forall e_{n-1} \in \widetilde{E}_{n-1}}{MAX} \{Low_{n-1}(r') - Low_{n-1}(r)\} \preceq 0$; en clair si l'état e_{n-1} accessible dans $\widetilde{E}_{n-1}$ qui maximiserait les écarts entre r_n et les autres rendez-vous fortement sensibilisés ne satisfait pas les contraintes de franchissement de la $Définition$ 13. (en d'autres termes r_n est non franchissable à partir de e_{n-1}), alors le rendez-vous r_n est donc non franchissable à partir de n'importe quel état accessible dans $\widetilde{E}_{n-1}$, puisque tous ne satisfont pas la condition de franchissement donnée dans la $Définition$ 13.

8.4 Preuve de la Proposition 2

1. La clause (1) est donnée en $Définition$ 14.
2. Démontrons les formules de calcul de la matrice $\widetilde{D}_n$

   (a) Cas de $\widetilde{D}_n[\bullet, t]$.
   Nous remplaçons $\widetilde{D}_n[\bullet, t]$ par son expression formelle donnée en *Définition 19*. De là, plusieurs cas peuvent être observés selon l'âge de la sensibilisation de la transition t :

      i. Si t est *persistante*, alors nous observons deux cas :

A. Si $t \in Ti_{n-1}$; t était *inhibée* au point $n-1$
$\widetilde{D}_n[\bullet,t] := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{y_n(t)\}$
D'après la *Définition 14* , nous avons $y_n(t) = y_{n-1}(t)$ d'où :
$\widetilde{D}_n[\bullet,t] := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{y_{n-1}(t)\}$
Comme l'espace de $\widetilde{E}_{n-1}$ inclut celui de $\widetilde{E}_n$, nous avons la sur-approximation :
$\widetilde{D}_{n-1}[\bullet,t] \preceq \widetilde{D}_{n-1}[\bullet,t]$

B. Si $t \notin Ti_{n-1}$; t *n'est pas inhibée* au point $n-1$
En conformité avec la *Définition 14*, nous avons : $y_n(t) = y_{n-1}(t) - \underline{r_n}$. De là, nous écrivons :
$\widetilde{D}_n[\bullet,t] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{y_{n-1}(t) - \underline{r_n}\}$.
Par conséquent, pour optimiser la valeur précédente, nous devons minimiser la valeur de $\underline{r_n}$. Pour ce faire, en se basant sur la condition de franchissement donnée par la *Définition 13* ; la variable $\underline{r_n}$ prend ses valeurs dans le respect des contraintes, (2) :

$$0 \preceq Low_{n-1}(r_n) \preceq \underline{r_n} \preceq MIN \begin{cases} \underset{\forall r \in \Phi_{n-1}(r_n)}{MIN} Low_{n-1}(r) \\ \underset{\forall r \in Senable_{n-1}}{MIN} Up_{n-1}(r) \end{cases}$$

De plus, les états appartenant à la classe $\widetilde{E}_n$ sont seulement la projection des états de la classe $\widetilde{E}_{n-1}$ satisfaisant l'inégalité (2). Par conséquent, nous pouvons écrire :
$\widetilde{D}_n[\bullet,t] := \underset{\forall e_{n-1} \in \widetilde{E}_{n-1}}{MAX} \{y_{n-1}(t) - Low_{n-1}(r_n)\}$.

Ensuite, pour obtenir la formule de $\widetilde{D}_n[\bullet,t]$, il suffit de remplacer $Low_{n-1}(r_n)$ par son expression formelle donnée en *Définition 11*, en considérant les valeurs possibles de $typ(r_n)$. Par exemple, si $typ(r_n) \in \{And, Async, Wand, Amas\}$, nous obtenons :

$$\widetilde{D}_n[\bullet,t] := \underset{\forall e_{n-1} \in \widetilde{E}_{n-1}}{MAX} \{y_{n-1}(t) - \underset{\forall t' \in Trans(r_n)}{MAX}\{x_{n-1}(t')\}\}$$
$$\widetilde{D}_n[\bullet,t] := \underset{\forall e_{n-1} \in \widetilde{E}_{n-1}}{MAX} \{\underset{\forall t' \in Trans(r_n)}{MIN}\{y_{n-1}(t) - x_{n-1}(t')\}\}$$
$$\widetilde{D}_n[\bullet,t] := \underset{\forall t' \in Trans(r_n)}{MIN} \underset{\forall e_{n-1} \in \widetilde{E}_{n-1}}{MAX} \{y_{n-1}(t) - x_{n-1}(t')\}$$
$$\widetilde{D}_n[\bullet,t] := \underset{\forall t' \in Trans(r_n)}{MIN} \left\{ \widetilde{D}_{n-1}[t',t] \right\}.$$

ii. Si t est *nouvellement sensibilisée,* alors en se basant sur la *Définition*.14, nous avons $y_n(t) = LFT(t)$:
$\widetilde{D}_n[\bullet,t] := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{y_n(t)\} := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{LFT(t)\} = LFT(t)$

(b) Cas de $\widetilde{D}_n[t,\bullet]$.

i. Si t est *persistante*, alors nous observons deux cas :

A. Si $t \in Ti_{n-1}$; t était *inhibée* au point $n-1$

En se basant sur la $D\acute{e}finition$ 19, nous écrivons :

$\widetilde{D}_n[t,\bullet] := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{-x_n(t)\}$

D'après la $D\acute{e}finition$ 14, nous avons $x_n(t) = x_{n-1}(t)$ d'où :

$\widetilde{D}_n[t,\bullet] := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{-x_{n-1}(t)\}$

Là aussi, comme l'espace de $\widetilde{E}_{n-1}$ inclut l'espace de $\widetilde{E}_n$, nous avons la sur-approximation :

$\widetilde{D}_n[t,\bullet] \preceq \widetilde{D}_{n-1}[t,\bullet]$

B. Si $t \notin Ti_{n-1}$; t *n'est pas inhibée* au point $n-1$

En conformité avec la $D\acute{e}finition$ 14, nous avons : $x_n(t) = MAX(x_{n-1}(t) - \underline{r_n}, 0)$. De là, nous écrivons :

$\widetilde{D}_n[t,\bullet] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \quad \{MIN(\underline{r_n} - x_{n-1}(t), 0)\}$

Pour optimiser l'expression $\{MIN(\underline{r_n} - x_{n-1}(t), 0)$, nous devons maximiser la valeur de $\underline{r_n}$. En se basant sur la relation (2), nous remplaçons $\underline{r_n}$ par $MIN \left\{ \begin{array}{l} \underset{\forall r \in \ \Phi_{n-1}(r_n)}{MIN} Low_{n-1}(r) \\ \underset{\forall r \in Senable_{n-1}}{MIN} Up_{n-1}(r) \end{array} \right.$, et dela nous écrivons :

$$\widetilde{D}_n[t,\bullet] := \underset{\forall e_{n-1} \in \widetilde{E}_{n-1}}{MAX} MIN \left(\begin{array}{l} 0 \\ MIN \left\{ \begin{array}{l} \underset{\forall r \in \ \Phi_{n-1}(r_n)}{MIN} Low_{n-1}(r) - x_{n-1}(t) \\ \underset{\forall r \in Senable_{n-1}}{MIN} Up_{n-1}(r) - x_{n-1}(t) \end{array} \right. \end{array} \right.$$

Comme $\Phi_{n-1}(r_n) \subseteq Senable_{n-1}$ nous pouvons alors écrire :

$\widetilde{D}_n[t,\bullet] :=$

$$MIN \left(\begin{array}{l} 0 \\ \underset{\forall r \in Senable_{n-1}}{MIN} \left\{ \begin{array}{ll} MIN \left\{ \begin{array}{l} \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{Low_n(r') - \{x_n(t)\} \\ \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{Up_n(r') - \{x_n(t)\} \end{array} \right. & \text{si } r' \in \ \Phi_n(r_n) \\ \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{Up_n(r') - x_n(t)\} & \text{sinon} \end{array} \right. \end{array} \right.$$

Ensuite, il suffit de remarquer l'expression de $\beta_{n-1}[t,r](r_n)$, et de là nous obtenons la formule :

$$\widetilde{D}_n[t,\bullet] := MIN\left(0, \underset{\forall r \in Senable_{n-1}}{MIN} \left\{\beta_{n-1}[t,r](r_n)\right\}\right)$$

A ce point, la précedente formule est exacte. Cependant comme les valeurs de $\beta_{n-1}[t,r](r_n)$ peuvent être approximées, par conséquent la valeur de $\widetilde{D}_n[t,\bullet]$ le devient aussi. Pour limiter cette sur-approximation lorsqu'elle se produit nous minorant la valeur de $\widetilde{D}_n[t,\bullet]$ comme suit :

$$\widetilde{D}_n[t,\bullet] := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{-x_n(t)\} := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{MIN(-y_n(t) - \alpha(t), 0)\}$$

$$\widetilde{D}_n[t,\bullet] := MIN\left(0, \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{-y_n(t)\} - \alpha(t)\right) := MIN\left(0, -\underset{\forall e_n \in \widetilde{E}_n}{MIN}\{y_n(t)\} - \alpha(t)\right)$$

$$\widetilde{D}_n[t,\bullet] \succeq MIN\left(0, -\underset{\forall e_n \in \widetilde{E}_n}{MAX}\{y_n(t)\} - \alpha(t)\right)$$

$\widetilde{D}_n[t,\bullet] \succeq MIN\left(0, -\widetilde{D}_n[\bullet,t] - \alpha(t)\right)$

Donc,

$\widetilde{D}_n[t,\bullet] \succeq MIN\left(0, -\widetilde{D}_n[\bullet,t] - \alpha(t)\right)$

Remarque : Cette minoration peut ne pas être de rigueur, elle permet ici de contrôler l'exactitude de l'algorithme. Dans le sens où la valeur de $MIN(0, \beta_{n-1}[t,r](r_n))$ doit être toujours supérieur ou égale à $MIN\left(0, -\widetilde{D}_n[\bullet,t] - \alpha(t)\right)$.

ii. Si t est *nouvellement sensibilisée,* alors en se basant sur la $D\acute{e}finition$.14, nous avons $x_n(t) = EFT(t)$:

$\widetilde{D}_n[t,\bullet] := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{-x_n(t)\} := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{-EFT(t)\} := -EFT(t)$

(c) Cas de $\widetilde{D}_n[t,t']$.

Nous remplaçons $\widetilde{D}_n[t,t']$ par son expression formelle donnée en $D\acute{e}finition\ 19$. De là, plusieurs cas peuvent être observés selon l'âge de la sensibilisation des transitions t' et t :

i. Si t ou t' est *nouvellement sensibilisée*, nous observons les cas suivants :

A. Si t est *nouvellement sensibilisée* :

$\widetilde{D}_n[t,t'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{y_n(t') - x_n(t)\} = \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{y_n(t') - EFT(t)\}$
$\widetilde{D}_n[t,t'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{y_n(t')\} - EFT(t) = \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet]$

B. Si t' est *nouvellement sensibilisée* :

$\widetilde{D}_n[t,t'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{y_n(t') - x_n(t)\} = \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{LFT(t') - x_n(t)\}$
$\widetilde{D}_n[t,t'] := LFT(t') + \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{-x_n(t)\} = \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet]$

C. Si t et t' sont *nouvellement sensibilisées* :
$\widetilde{D}_n[t,t'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{y_n(t') - x_n(t)\} = LFT(t') - EFT(t) = \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet]$

ii. Si t et t' sont *persistantes*, plusieurs sous cas sont observés selon le statut des transitions t et t' au point $n-1$.

A. Si $t \in Ti_{n-1}$ et $t' \in Ti_{n-1}$; les deux transitions *sont inhibées* au point $n-1$.

$\widetilde{D}_n[t,t'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\ \{y_n(t') - x_n(t)\}$
En se basant sur la $D\acute{e}finition\ 14$: $y_n(t) = y_{n-1}(t)$ et $x_n(t) = x_{n-1}(t)$.
$\widetilde{D}_n[t,t'] = \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{y_{n-1}(t') - x_{n-1}(t)\}$
D'un coté nous avons : $\widetilde{D}_n[t,t'] \preceq \widetilde{D}_{n-1}[t,t']$
et de l'autre : $\widetilde{D}_n[t,t'] \preceq \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet]$.

Nous pouvons prouver par induction sur *(n)* que :

$\widetilde{D}_n[t,t'] := \widetilde{D}_{n-1}[t,t']$ si $\widetilde{D}_{n-1}[t,t'] \preceq \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet]$

$\widetilde{D}_n[t,t'] := \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet]$ sinon

D'où la formule : $\widetilde{D}_n[t,t'] := MIN(\widetilde{D}_{n-1}[t,t'], \quad \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet])$

B. Si $t \notin Ti_{n-1}$ et $t' \notin Ti_{n-1}$; les deux transitions ne sont *pas inhibées* au point $n-1$.
$\widetilde{D}_n[t,t'] = \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{y_n(t') - x_n(t)\}$
Toujours sur la base de la $D\acute{e}finition$ 14, nous écrivons :
$\widetilde{D}_n[t,t'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{y_n(t') - MAX(0, x_{n-1}(t) - \underline{r_n})\}$
$\widetilde{D}_n[t,t'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{y_n(t') + MIN(0, \underline{r_n} - x_{n-1}(t))\}$
$\widetilde{D}_n[t,t'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{MIN(y_n(t'), \quad y_n(t') + \underline{r_n} - x_{n-1}(t))\}$

$$\widetilde{D}_n[t,t'] := MIN \begin{cases} \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{y_n(t')\} \\ \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{(y_{n-1}(t') - \underline{r_n}) + \underline{r_n} - x_{n-1}(t)\} \end{cases}$$

Après élimination de $\underline{r_n}$

$$\widetilde{D}_n[t,t'] := MIN \begin{cases} MAX_{\forall e_n \in \widetilde{E}_n} \{y_{n-1}(t') - x_{n-1}(t)\} \\ \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{y_n(t')\} \end{cases}$$

Comme pour la démonstration précédente, nous prouvons par induction que :

$$\widetilde{D}_n[t,t'] := MIN \begin{cases} MIN(\widetilde{D}_{n-1}[t,t'], \quad \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet]) \\ \\ \widetilde{D}_n[\bullet,t'] \end{cases}$$

De là, nous déduisons la formule : $\widetilde{D}_n[t,t'] := MIN(\widetilde{D}_{n-1}[t,t'], \quad \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet])$

C. Si $t \in Ti_{n-1}$ et $t' \notin Ti_{n-1}$; Seule la transition t *est inhibée* au point $n-1$.
$\widetilde{D}_n[t,t'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{y_n(t') - x_n(t)\}$
D'après la $D\acute{e}finition$ 14, on a $y_n(t') = y_{n-1}(t') - \underline{r_n}$ et $x_n(t) = x_{n-1}(t)$.
$\widetilde{D}_n[t,t'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{y_{n-1}(t') - \underline{r_n} - x_{n-1}(t)\}$
$\widetilde{D}_n[t,t'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX} \{y_{n-1}(t') - \underline{r_n} - x_{n-1}(t)\}$
En remplaçant $\underline{r_n}$ par sa valeur minimum donnée par $Low_{n-1}(r_n)$, et en se projetant sur l'espace de $\widetilde{E}_{n-1}$, nous obtenons :
$\widetilde{D}_n[t,t'] := \underset{\forall e_{n-1} \in \widetilde{E}_{n-1}}{MAX} \{y_{n-1}(t') - x_{n-1}(t) - Low_{n-1}(r_n)\}$
En procédant ensuite à la sur-approximation de la valeur de $\widetilde{D}_n[t,t']$:
$\widetilde{D}_n[t,t'] \preceq \underset{\forall e_{n-1} \in \widetilde{E}_{n-1}}{MAX} \{y_{n-1}(t') - x_{n-1}(t)\} + \underset{\forall e_{n-1} \in \widetilde{E}_{n-1}}{MAX} \{-Low_{n-1}(r_n)\}$

$$\widetilde{D}_n[t,t'] \preceq\!\!\!\!\!\!\sim \quad \widetilde{D}_{n-1}[t,t'] + \widetilde{D}_{n-1}[r_n,\bullet]$$

Comme $\widetilde{D}_n[t,t'] \preceq \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet]$ nous déduisons la formule :

$$\widetilde{D}_n[t,t'] \preceq\!\!\!\!\!\!\sim \quad MIN \begin{pmatrix} \widetilde{D}_{n-1}[t,t'] + \widetilde{D}_{n-1}[r_n,\bullet] \\ \\ \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet] \end{pmatrix}$$

D. Si $t \notin Ti_{n-1}$ et $t' \in Ti_{n-1}$; Seule la transition t' était *inhibée* au point $n-1$.
$\widetilde{D}_n[t,t'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{y_n(t') - x_n(t)\}$

D'après la *Définition* 14, on a :
$y_n(t') := y_{n-1}(t')$ et $x_n(t) := MAX(0, x_{n-1}(t) - \underline{r_n})$, d'où :
$\widetilde{D}_n[t,t'] := \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{y_n(t') - MAX(0, x_{n-1}(t) - \underline{r_n})\}$
$\widetilde{D}_n[t,t'] := MIN\left(\underset{\forall e_n \in \widetilde{E}_n}{MAX} y_n(t'), \quad \underset{\forall e_n \in \widetilde{E}_n}{MAX}\left\{y_n(t') + \underline{r_n} - x_{n-1}(t)\right\}\right)$
$\widetilde{D}_n[t,t'] := MIN\left(\underset{\forall e_n \in \widetilde{E}_n}{MAX} y_n(t'), \quad \underset{\forall e_n \in \widetilde{E}_n}{MAX}\left\{y_n(t') + \underline{r_n} - x_{n-1}(t)\right\}\right)$
En procédant ensuite à la sur-approximation de la valeur de $\widetilde{D}_n[t,t']$:
$\widetilde{D}_n[t,t'] \preceq\!\!\!\!\!\!\sim \quad MIN\left(\underset{\forall e_n \in \widetilde{E}_n}{MAX} y_n(t'), \quad \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{y_n(t') - x_{n-1}(t)\} + \underset{\forall e_n \in \widetilde{E}_n}{MAX} \underline{r_n}\right)$
Ensuite, nous posons :
$DS_n[n-1,n] \;=\; \underset{\forall e_{n-1} \in \widetilde{E}_{n-1}}{MAX}\left\{\underline{r_n}\right\} \succeq 0$
Par conséquent, nous écrivons :

$$\widetilde{D}_n[t,t'] \preceq\!\!\!\!\!\!\sim \quad MIN\left(\begin{array}{l} \widetilde{D}_n[\bullet,t'], \quad MIN\left(\begin{array}{l} \widetilde{D}_{n-1}[t,t'] + DS_n[n-1,n] \\ \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet] + DS_n[n-1,n] \end{array}\right) \end{array}\right)$$

Comme $\widetilde{D}_n[t,t'] \preceq\!\!\!\!\!\!\sim \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet]$ nous écrivons :

$$\widetilde{D}_n[t,t'] \preceq\!\!\!\!\!\!\sim \quad MIN\left(\widetilde{D}_{n-1}[t,t'] + DS_n[n-1,n], \quad \widetilde{D}_n[\bullet,t'] + \widetilde{D}_n[t,\bullet]\right)$$

La démonstration des formules de calcul de $DS_n[n-1,n]$ est donnée ci-après.

8.5 Preuve de la Proposition 3

1. cas où $i = n$

   (a) $DS_n[n,n] := 0$; évident.

   (b) $DS_n[n, \overrightarrow{r}] := \underset{space(S_i^n)}{MAX}\{Up_n(r)\}$

   En se basant sur les *Définitions 21* et *22*, il apparait clairement que $space(S_i^n)$ détermine l'espace de $\widetilde{E}_n$.

   $DS_n[n, \overrightarrow{r}] := \underset{space(S_i^n)}{MAX}\{Up_n(r)\} = \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{Up_n(r)\} = DR_n[\bullet, r]$.

(c) $DS_n[n, \overleftarrow{r}] := \underset{space(S_i^n)}{MAX}\{Low_n(r)\}$

Remplaçons ensuite $Low_n(r)$ par son expression formelle selon le type de synchronisation associé à r. Par exemple, si $typ(r) = And$

$DS_n[n, \overleftarrow{r}] := \underset{space(S_i^n)}{MAX}\underset{\forall t \in Trans(r)}{MAX}\{x_n(t)\} := \underset{\forall t \in Trans(r)}{MAX}\underset{space(S_i^n)}{MAX}\{MAX(0, \quad y_n(t) + \alpha(t))\}$

$DS_n[n, \overleftarrow{r}] := MAX\left(0, \underset{\forall t \in Trans(r)}{MAX}\underset{space(S_i^n)}{MAX}\{y_n(t) + \alpha(t)\}\right)$

$DS_n[n, \overleftarrow{r}] := MAX\left(0, \underset{\forall t \in Trans(r)}{MAX}\left\{\widetilde{D}_n[\bullet, t] + \alpha(t)\right\}\right).$

(d) $DS_n[r, n] := -\underset{space(S_i^n)}{MIN}\{Low_n(r)\} = -\underset{\forall e_n \in \widetilde{E}_n}{MIN}\{Low_n(r)\} = DR_n[r, \bullet].$

(e) $DS_n[n, t] := \underset{space(S_i^n)}{MAX}\{y_n(t)\} = \underset{\forall e_n \in \widetilde{E}_n}{MAX}\{y_n(t)\} = \widetilde{D}_n[\bullet, t].$

(f) $DS_n[t, n] := -\underset{space(S_i^n)}{MIN}\{x_n(t)\} = -\underset{\forall e_n \in \widetilde{E}_n}{MIN}\{x_n(t)\} = \widetilde{D}_n[t, \bullet].$

2. Cas de $DS_n[i, n]$:

Remplaçons $DS_n[i, n]$ par son expression formelle de la *Définition 23* :
$DS_n[i, n] = \underset{space(S_i^n)}{MAX}\{\underline{r_{i+1}} + .. + \underline{r_n}\}$.

Ensuite, en utilisant la *Définition 22*, nous nous projetons sur l'espace $space(S_i^{n-1})$ en maximisant la valeur de $\underline{r_n}$ en lui substituant $MIN\left(\begin{array}{c}\underset{\forall r \in \ \Phi_{n-1}(r_n)}{MIN} Low_{n-1}(r) \\ \underset{\forall r \in Senable_{n-1}}{MIN} Up_{n-1}(r)\end{array}\right)$

Nous obtenons ensuite la valeur de $DS_n[i, n]$ comme suit :

$$:= \underset{space(S_i^{n-1})}{MAX}\left\{\underline{r_{i+1}} + .. + \underline{r_{n-1}} + MIN\left(\begin{array}{c}\underset{\forall r \in \ \Phi_{n-1}(r_n)}{MIN} Low_{n-1}(r) \\ \underset{\forall r \in Senable_{n-1}}{MIN} Up_{n-1}(r)\end{array}\right)\right\}$$

$$:= MIN\left(\begin{array}{l}\underset{\forall r \in \ \Phi_{n-1}(r_n)}{MIN}\underset{space(S_i^{n-1})}{MAX}\underline{r_{i+1}} + .. + \underline{r_{n-1}} + Low_{n-1}(r) \\ \underset{\forall r \in Senable_{n-1}}{MIN}\underset{space(S_i^{n-1})}{MAX}\underline{r_{i+1}} + .. + \underline{r_{n-1}} + Up_{n-1}(r)\end{array}\right)$$

$$:= MIN\left(\begin{array}{l}\underset{\forall r \in \ \Phi_{n-1}(r_n)}{MIN}\{DS_{n-1}[i, \overleftarrow{r}]\} \\ \underset{space(S_i^{n-1})}{MAX}\{DS_{n-1}[i, \overrightarrow{r}]\}\end{array}\right)$$

3. Cas de $DS_n[n, i]$:

Nous pouvons écrire : $DS_n[n, i] := -\underset{space(S_i^n)}{MIN}\{\underline{r_{i+1}} + .. + \underline{r_n}\}$.

En utilisant la *Définition 22*, nous nous projetons sur l'espace $space(S_i^{n-1})$ en minimisant la valeur de $\underline{r_n}$, ce qui revient à lui substituer $Low_{n-1}(r_n)$:

$DS_n[n, i] := -\underset{space(S_i^{n-1})}{MIN}\{\underline{r_{i+1}} + .. + \underline{r_{n-1}} + Low_{n-1}(r_n)\}$

$DS_n[n, i] := DS_{n-1}[r_n, i].$

4. Cas de $DS_n[i, \overrightarrow{r}]$:

Nous écrivons : $DS_n[i, \overrightarrow{r}] := \underset{space(S_i^n)}{MAX} \{\underline{r_{i+1}} + .. + \underline{r_n} + Up_n(r)\}$

Remplaçons $Up_n(r)$ par son expression formelle selon le type de synchronisation associé au rendez-vous. En supposant par exemple que *typ(r)* $\in \{Or, Wand, Wmas\}$, on obtient :

$DS_n[i, \overrightarrow{r}] := \underset{space(S_i^n)}{MAX} \{\underline{r_{i+1}} + .. + \underline{r_n} + \underset{\forall t \in Trans(r)}{MAX} \{y_n(t)\}\}$

$DS_n[i, \overrightarrow{r}] := \underset{\forall t \in Trans(r)}{MAX} \underset{space(S_i^n)}{MAX} \{\underline{r_{i+1}} + .. + \underline{r_n} + y_n(t)\}$

$DS_n[i, \overrightarrow{r}] := \underset{\forall t \in Trans(r)}{MAX} \{DS_n[i, t]\}.$

5. Cas de $DS_n[i, \overleftarrow{r}]$:

   $DS_n[i, \overleftarrow{r}] := \underset{space(S_i^n)}{MAX} \{\underline{r_{i+1}} + .. + \underline{r_n} + Low_n(r)\}$

   En supposant que *typ(r)* $= And$, on obtient :

   $DS_n[i, \overleftarrow{r}] := \underset{space(S_i^n)}{MAX} \{\underline{r_{i+1}} + .. + \underline{r_n} + \underset{\forall t \in Trans(r)}{MAX} \{x_n(t)\}\}$

   $DS_n[i, \overleftarrow{r}] := \underset{space(S_i^n)}{MAX} \left\{\underline{r_{i+1}} + .. + \underline{r_n} + \underset{\forall t \in Trans(r)}{MAX} \{MAX(0, y_n(t) + \alpha(t))\}\right\}$

   $$DS_n[i, \overleftarrow{r}] := MAX \left(\begin{array}{l} \underset{space(S_i^n)}{MAX} \left\{\underline{r_{i+1}} + .. + \underline{r_n}\right\} \\ \underset{\forall t \in Trans(r)}{MAX} \underset{space(S_i^n)}{MAX} \left\{\underline{r_{i+1}} + .. + \underline{r_n} + y_n(t) + \alpha(t)\right\} \end{array} \right)$$

   $$DS_n[i, \overleftarrow{r}] := MAX \left(\begin{array}{l} DS_n[i, n] \\ \underset{\forall t \in Trans(r)}{MAX} \{DS_n[i, t] + \alpha(t)\} \end{array} \right)$$

6. Cas de $DS_n[r, i]$:

   $DS_n[r, i] := \underset{space(S_i^n)}{-MIN} \{\underline{r_{i+1}} + .. + \underline{r_n} + Low_n(r)\}.$

   En supposant que *typ(r)* $= And$, on obtient :

   $DS_n[r, i] := \underset{space(S_i^n)}{-MIN} \{\underline{r_{i+1}} + .. + \underline{r_n} + \underset{\forall t \in Trans(r)}{MAX} \{x_n(t)\}\}$

   $DS_n[r, i] := \underset{\forall t \in Trans(r)}{MIN} \underset{space(S_i^n)}{-MIN} \{\underline{r_{i+1}} + .. + \underline{r_n} + x_n(t)\}.$

   $DS_n[r, i] := \underset{\forall t \in Trans(r)}{MIN} \{DS_n[t, i]\}.$

7. Cas de $DS_n[i, t]$

   (a) Si t est persistante, deux cas sont observés :

      i. Si t est *non inhibée* au point $(n-1)$.

      $DS_n[i, t] := \underset{space(S_i^n)}{MAX} \{\underline{r_{i+1}} + .. + \underline{r_n} + y_n(t)\}$

      Nous remplaçons $y_n(t)$ par $y_{n-1}(t) - \underline{r_n}$; nous éliminons ensuite la variable $\underline{r_n}$.

      $DS_n[i, t] := \underset{space(S_i^n)}{MAX} \{\underline{r_{i+1}} + .. + \underline{r_n} + (y_{n-1}(t) - \underline{r_n})\}$

      $DS_n[i, t] := \underset{space(S_i^n)}{MAX} \{\underline{r_{i+1}} + .. + \underline{r_{n-1}} + y_{n-1}(t)\}$

      D'un coté nous avons, $DS_n[i, t] \preceq \underset{space(S_i^{n-1})}{MAX} \{\underline{r_{i+1}} + .. + \underline{r_{n-1}} + y_{n-1}(t)\} = DS_{n-1}[i, t]$

      De l'autre coté : $DS_n[i, t] \preceq \underset{space(S_i^n)}{MAX} \{\underline{r_{i+1}} + .. + \underline{r_n}\} + \underset{space(S_i^n)}{MAX} \{y_n(t)\}.$

Et nous prouvons par induction sur la valeur de *(i)* que :

$DS_n[i,t] := DS_{n-1}[i,t]$ $\quad DS_{n-1}[i,t] \preceq DS_n[i,n] + D_n[\bullet,t]$

$DS_n[i,t] := DS_n[i,n] + D_n[\bullet,t]$ $\quad$ sinon

d'où la formule

$DS_n[i,t] := MIN(DS_{n-1}[i,t], \quad DS_n[i,n] + D_n[\bullet,t])$

ii. Si t est *inhibée* au point $(n-1)$.

$DS_n[i,t] := \underset{space(S_i^n)}{MAX}\{\underline{r_{i+1}} + .. + \underline{r_n} + y_n(t)\}.$

D'après la *Définition .14,* on a : $y_n(t) = y_{n-1}(t)$

$DS_n[i,t] := \underset{space(S_i^n)}{MAX}\{\underline{r_{i+1}} + .. + \underline{r_n} + y_{n-1}(t)\}$

En procédant à la sur-approximation de la valeur de $DS_n[i,t]$, nous obtenons :

$$DS_n[i,t] \preceq \underset{space(S_i^{n-1})}{MAX}\{\underline{r_{i+1}} + .. + \underline{r_{n-1}} + y_{n-1}(t)\} + \underset{space(S_i^n)}{MAX}\{\underline{r_n}\}$$

$$DS_n[i,t] \preceq \underset{space(S_i^n)}{MAX}\{\underline{r_{i+1}} + .. + \underline{r_n}\} + \underset{space(S_i^n)}{MAX}\{y_{n-1}(t)\}$$

$$DS_n[i,t] \preceq MIN\begin{pmatrix} DS_{n-1}[i,t] + DS_n[n-1,n] \\ DS_n[i,n] + \widetilde{D}_n[\bullet,t] \end{pmatrix}$$

(b) Si t est *nouvellement sensibilisée.*

$DS_n[i,t] := \underset{space(S_i^n)}{MAX}\{\underline{r_{i+1}} + .. + \underline{r_n} + y_n(t)\}.$

Tenant en compte que $y_n(t) = LFT(t)$, nous écrivons :

$DS_n[i,t] := \underset{space(S_i^n)}{MAX}\{\underline{r_{i+1}} + .. + \underline{r_n}\} + LFT(t) = DS_n[i,n] + LFT(t).$

8. Cas de $DS_n[t,i]$

(a) Si t est *persistante*, deux cas sont observés :

i. Si t est *non inhibée* au point $n-1$.

$DS_n[t,i] := -\underset{space(S_i^n)}{MIN}\{\underline{r_{i+1}} + .. + \underline{r_n} + x_n(t)\}.$

Nous pouvons constater que :

$DS_n[t,i] \preceq -\underset{space(S_i^n)}{MIN}\{\underline{r_{i+1}} + .. + \underline{r_n}\} - \underset{space(S_i^n)}{MIN}\{x_n(t)\}$

$DS_n[t,i] \preceq DS_n[n,i] + \widetilde{D}_n[t,\bullet].$

Par ailleurs, nous remplaçons $x_n(t)$ par $MAX(0, \quad x_{n-1}(t) - \underline{r_n})$ et nous obtenons :

$DS_n[t,i] := -\underset{space(S_i^n)}{MIN}\{\underline{r_{i+1}} + .. + \underline{r_n} + MAX(0, \quad x_{n-1}(t) - \underline{r_n})\}$

$$DS_n[t,i] := MIN\begin{Bmatrix} -\underset{space(S_i^n)}{MIN}\left\{\underline{r_{i+1}} + .. + \underline{r_{n-1}} + x_{n-1}(t)\right\} \\ -\underset{space(S_i^n)}{MIN}\left\{\underline{r_{i+1}} + .. + \underline{r_n}\right\} \end{Bmatrix}$$

Comme l'espace $space(S_i^{n-1})$ inclut celui de $space(S_i^n)$.

$$DS_n[t,i] \preceq MIN\begin{Bmatrix} -\underset{space(S_i^{n-1})}{MIN}\left\{\underline{r_{i+1}} + .. + \underline{r_{n-1}} + x_{n-1}(t)\right\} \\ DS_n[n,i] \end{Bmatrix}$$

$$DS_n[t,i] \preceq MIN\begin{Bmatrix} DS_{n-1}[t,i] \\ DS_n[n,i] \end{Bmatrix}$$

Ensuite, nous prouvons par induction sur la valeur de (i) que :

$$DS_n[t,i] := \begin{cases} DS_{n-1}[t,i] & \text{si } DS_{n-1}[t,i] \preceq \widetilde{D}_n[t,\bullet] + DS_n[n,i] \\ \widetilde{D}_n[t,\bullet] + DS_n[n,i] & \text{sinon} \end{cases}$$

De là on déduit la formule :

$$DS_n[t,i] := MIN \begin{cases} DS_{n-1}[t,i] \\ \widetilde{D}_n[t,\bullet] + DS_n[n,i] \end{cases}$$

ii. Si t est *inhibée* au point $n-1$.

$$DS_n[t,i] := - \underset{space(S_i^n)}{MIN} \{\underline{r_{i+1}} + .. + \underline{r_n} + x_n(t)\}.$$

$$DS_n[t,i] := - \underset{space(S_i^n)}{MIN} \{\underline{r_{i+1}} + .. + \underline{r_n} + x_{n-1}(t)\}$$

En procédant à la sur-approximation de la valeur de $DS_n[t,i]$, nous obtenons :

$$DS_n[t,i] \preceq - \underset{space(S_i^n)}{MIN} \{\underline{r_{i+1}} + .. + \underline{r_{n-1}} + x_{n-1}(t)\} - \underset{space(S_i^n)}{MIN} \{\underline{r_n}\}$$

$$DS_n[t,i] \preceq - \underset{space(S_i^n)}{MIN} \{\underline{r_{i+1}} + .. + \underline{r_n}\} - \underset{space(S_i^n)}{MIN} \{x_n(t)\}$$

Nous obtenous la formule

$$DS_n[t,i] \preceq MIN \begin{cases} DS_{n-1}[t,i] + DS_n[n,n-1] \\ DS_n[n,i] + \widetilde{D}_n[t,\bullet] \end{cases}$$

$$DS_n[t,i] \preceq MIN \begin{cases} DS_{n-1}[t,i] + DR_{n-1}[\bullet,r_n] \\ DS_n[n,i] + \widetilde{D}_n[t,\bullet] \end{cases}$$

(b) Si t est *nouvellement sensibilisée.*

$$DS_n[t,i] = -MIN_{space(S_i^n)}\{\underline{r_{i+1}} + .. + \underline{r_n} + x_n(t)\}.$$

Nous avons, $x_n(t) := EFT(t)$; ce qui revient à écrire :

$$DS_n[t,i] = - \underset{space(S_i^n)}{MIN} \{\underline{r_{i+1}} + .. + \underline{r_n}\} - EFT(t) = DS_n[n,i] - EFT(t).$$

Chapitre 9

Annexe B : Implémentations et études de cas

Les tests de simulation ont été réalisés sur un **Pentium V** -Vitesse Processeur 2,7 GH et doté d'une capacité mémoire vive de 512 Mo.

Selon les besoins, les tests ont été effectués en utilisant plusieurs outils : l'outil *TINA* [91], l'outil *ROMEO* [82], l'outil *ORIS* [77], et enfin l'outil que nous avons développé *ITPN TOOL[96].* Cet outil permet de construire le graphe sur-approximé *DBM* d'un réseau de Petri temporel étendu aux arcs inhibiteurs. Ceci revient à considérer l'approche théorique présentée dans cette thèse en considérant des rendez-vous ayant comme règles de synchronisation *Async,* i.e chaque rendez-vous est asynchrone constitué d'une seule transition.

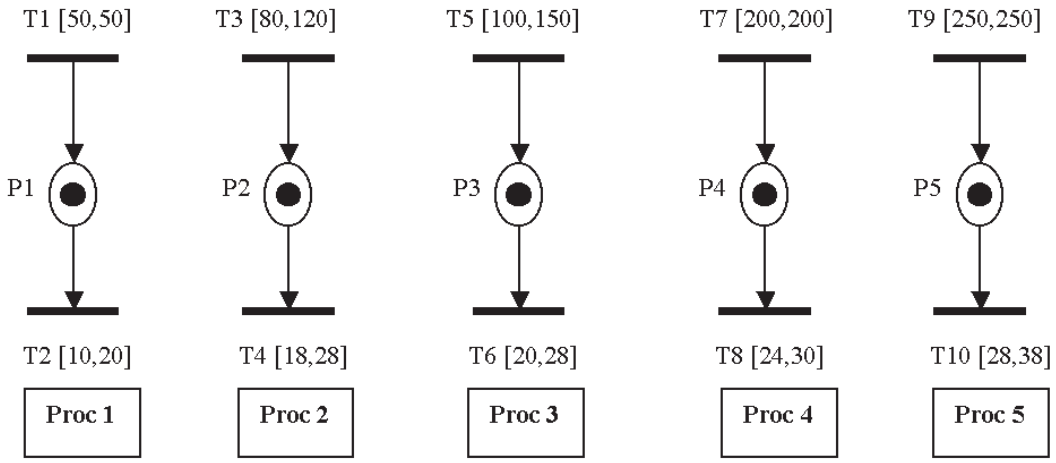

FIGURE 9.1 – Modèles *RdPT* utilisés dans les tests.

Par ailleurs, certains outils ne fournissent pas tous les paramètres, tels le temps de calcul ou le nombre d'arcs. Ces données non renseignées seront indiquées ultérieurement par l'abréviation *ND* (non disponible). Nous dénotons par l'abréviation *NA* (non abouti) les tests de simulations qui ont mené à un crash mémoire.

Les premiers tests que nous avons réalisés tendent à valider notre outil, C.à.d vérifier est ce que le calcul des graphes exacts pour des *RdPT* est conforme avec les autres outils. Pour ce faire, nous avons considéré les combinaisons des *RdPT* donnés en *Figure 9.1.* D'abord en expérimentant

le réseau *Proc1* ensuite en le combinant avec *Proc2*, puis avec *Proc3* et *Proc4* et ainsi de suite.

TABLE 9.1 – Résultats d'expérimentations réalisées sur les *RdPTs*.

Exemples	*OUTILS*	*TINA*	*ROMEO*	*ITPN*
Proc 1	*Classes*	2	2	2
	Transitions	2	2	2
	Temps (ms)	0	*ND*	0
Proc 1 2	*Classes*	186	186	186
	Transitions	262	262	262
	Temps (ms)	0	*ND*	0
Proc 1 2 3	*Classes*	958	958	958
	Transitions	1506	1506	1506
	Temps (ms)	1	*ND*	3
Proc 1 2 3 4	*Classes*	5219	5219	5219
	Transitions	8580	8580	8580
	Temps (ms)	31	*ND*	44
Proc 1 2 3 4 5	*Classes*	42909	42909	42909
	Transitions	73842	73842	73842
	Temps (ms)	734	*ND*	830

Les résultats de ces expérimentations sont illustrés en *Table 9.1.* Les résultats montrent que les résultats donnés pour tous les outils sont identiques. Un avantage certain pour l'outil *TINA* en ce qui concerne les temps de calcul. Cependant, les temps de calculs obtenus avec notre outil, pourraient être grandement améliorées par l'emploi de structures mieux adaptées et d'algorithmes de recherches plus efficaces.

La seconde batterie de tests tente de situer l'approche d'approximation *DBM* que nous proposons ici, par rapport aux autres techniques existantes. Pour cet effet, nous avons utilisé les réseaux *ITPN (Inhibitor Time Petri Net)* de la *Figure 9.2* en faisant varier les intervalles des transitions t_3 et t_5. Nous avons comparé les graphes en considérant trois paramètres, le nombre de classes, le nombres d'arcs et enfin en terme de temps de calcul. Par rapport à ce dernier critère seul l'outil *TINA* permet de fournir les temps d'exécutions, les temps des autres outils sont perçues comme étant moins avantageux.

Ces temps permettent de donner une appréciation des gains en complexité de calcul de l'approche *DBM* que nous développons par rapport aux approches exactes *Polyèdre[68]* et *K-grille[33]* développés respectivement dans les outils *ROMEO* et *TINA*. Les résultats de ces expérimentations sont reportés dans la *Table 9.2*. De là, il est apparu clairement que la méthode des *K-grilles* développée dans *TINA* est meilleure en terme de complexité que celle des polyèdres implémentée dans *ROMEO*. Pour preuve les temps que nous percevons de *ROMEO* mêmes s'ils sont non renseignés sont assez lents, et les crashages mémoire constatés pour les tests (2, 4 et 6) en sont une illustration.

Par ailleurs, nous avions été surpris de constater que l'approximation *DBM* développée dans [42] et implémentée dans l'outil *ORIS* [77] donnait les meilleurs résultats car théoriquement nous prouvons le contraire. Toutefois, les graphes générés pour le premier et cinquième tests nous ont vite rassuré. Car nous constatons que cette méthode qui est censé être une sur-approximation produit des graphes moins volumineux que les graphe exacts générés par *TINA* et *ROMEO*. La fiabilité de l'outil est mise en cause car la procédure de calcul du graphe n'est implémentée que dans

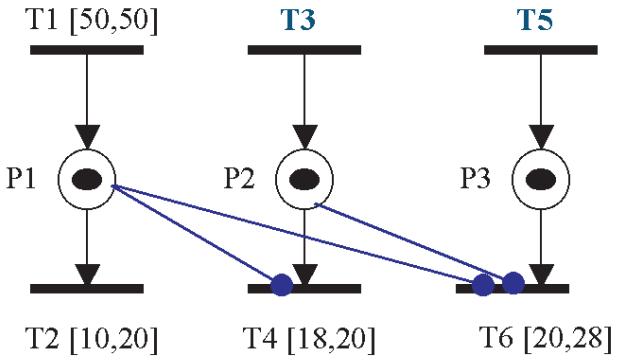

FIGURE 9.2 – *ITPNs* utilisés dans les expérimentations.

une ancienne version de l'outil et fut supprimée dans la version la plus récente. Par conséquent, seul l'outil *ROMEO* permet de fournir une approximation *DBM* fiable avec qui nous pouvons comparer les résultats. D'ailleurs, les écarts entre les tailles des graphes que nous calculons et ceux de *ROMEO* sont en notre faveur. Si *ROMEO* calcule des graphes plus compacts pour les tests (1,2,4,5) néanmoins ces écarts ne dépassent pas une centaine de classes, alors que lorsque les graphes sont volumineux nous constatons que notre approche est plus performante et que les écarts peuvent atteindre jusqu'à plusieurs milliers de classes (voir tests 6, 7, 13, 14, 16).

Plus encore, les temps que nous mesurons avec notre outil sont plus avantageux que les autres outils et plus spéciquement par rapport à l'approche *DBM* de *ROMEO* qui induit des temps de l'ordre d'une dizaine de secondes pour les graphes dépassant les 10000 classes.

TABLE 9.2 – Résultats d'expérimentations réalisées sur les *ITPN*.

Exemples	*OUTILS*	*ORIS*	*TINA*	*ROMEO*		*ITPN*
	Méthode	*DBM*	*K-grilles*	*Exact*	*DBM*	*DBM*
t_3 [100,150] t_5 [160,160]	*Classes*	4.066	4.489	4.489	5.431	5.462
	Arcs	ND	6.360	6.360	7.608	7.632
	Temps(ms)		2188	*ND*	*ND*	45
t_3 [100,150] t_5 [155,155]	*Classes*	18.801	17.612	*NA*	21.857	22.008
	Arcs	ND	24.522	*NA*	30.065	30.203
	Temps(ms)		9422	*NA*	*ND*	365
t_3 [100,150] t_5 [150,150]	*Classes*	366	320	320	403	402
	Arcs		460	460	575	574
	Temps(ms)		156	*ND*	*ND*	1
t_3 [100,150] t_5 [145,145]	*Classes*	18034	16.913	*NA*	21.033	21.184
	Arcs	ND	23.583	*NA*	28.989	29.127
	Temps(ms)		9016	*NA*	*ND*	336
t_3 [100,150] t_5 [140,140]	*Classes*	3728	4.142	4.142	5.034	5.065
	Arcs	ND	5.889	5.889	7.095	7.119
	Temps (ms)		2000	*ND*	*ND*	37
t_3 [100,150] t_5 [135,135]	*Classes*	12896	11.351	*NA*	16.354	15.358
	Arcs		15.649	*NA*	22.230	20.835
	Temps (ms)		6016	*NA*	*ND*	196
t_3 [100,150] t_5 [130,130]	*Classes*	4021	4.281	4.281	5.903	5.428
	Arcs	ND	6.082	6.082	8.289	7.593
	Temps (ms)		2219	2219	*ND*	40

Exemples	*OUTILS*	*ORIS*	*TINA*	*ROMEO*		*ITPN*
	Méthodes	*DBM*	*K-grilles*	*Exact*	*DBM*	*DBM*
t_3 [100, ∞[	*Classes*	1668	1.309	1.309	2.117	2.121
t_5 [140,140]	*Arcs*	*ND*	1.738	1.738	2.791	2.793
	Temps (ms)		594	*ND*	*ND*	34
t_3 [100, ∞ [	*Classes*	8149	5.490	5.490	9.268	9.258
t_5 [145,145]	*Arcs*	*ND*	7.070	7.070	11.885	11.862
	Temps (ms)		2297	*ND*	*ND*	204
t_3 [100, ∞ [	*Classes*	227	166	166	244	244
t_5[150,150]	*Arcs*	*ND*	228	228	333	333
	Temps(ms)		94	*ND*	*ND*	0
t_3 [100, ∞ [	*Classes*	8491	5.716	5.716	9.626	9.616
t_5 [155,155]	*Arcs*	*ND*	7.354	7.354	12.322	12.299
	Temps(ms)		2375	*ND*	*ND*	206
t_3 [100, ∞[	*Classes*	1815	1.418	1.418	2.280	2.284
t_5 [160,160]	*Arcs*	*ND*	1.876	1.876	2.988	2990
	Temps(ms)		609	*ND*	*ND*	19
t_3 [80,120]	*Classes*	7.505	7.175	7.175	11.368	10.983
t_5 [160,160]	*Arcs*	*ND*	10.425	10.425	16.271	15.636
	Temps(ms)		3750	*ND*	*ND*	121
t_3 [80,120]	*Classes*	36981	28 392	*NA*	47.622	46.409
t_5 [155,155]	*Arcs*	*ND*	41.452	*NA*	67.309	65.221
	Temps (ms)		6452	*NA*	*ND*	1080
t_3 [80,120]	*Classes*	35949	1.229	1.229	2.463	2.224
t_5 [150,150]	*Arcs*	*ND*	1.826	1.826	3.653	3.274
	Temps (ms)		703	*ND*	*ND*	8
t_3 [80,120]	*Classes*	8236	7.018	*NA*	12.379	11.505
t_5 [140,140]	*Arcs*	*ND*	10.242	*NA*	17.829	16.474
	Temps (ms)		3672	*NA*	*ND*	285