Modèle de réseaux de Pétri pour un système de partitions interactives

Antoine Allombert

23 janvier 2008

Table des matières

1 Les partitions interactives

Je présente ici rapidement la problématique qui est la nôtre. Notre objectif est de développer un formalisme pour la composition et l'interprétation de pièces musicales interactives. Cette idée de pièces interactives découle du fait que la musique contemporaine utilise massivement l'informatique mais que les outils disponibles peinent à proposer une expressivité satisfaisante pour définir un cadre d'interprétation des pièces contemporaines. L'interprétation d'une pièce musicale passe par la modification pendant l'exécution de certains paramètres de celle-ci par le musicien. Il peut s'agir de l'anticipation ou du retardement de début ou fin de notes par rapport aux dates écrites par le compositeur ou encore la modification de paramètres sonores de certaines notes. Cependant, ces modifications ne peuvent s'effectuer que dans un cadre défini par le compositeur qui limite délibérément les possibilités pour éviter que sa pièce se trouve dénaturée par des modifications trop importantes. Ainsi l'interprétation s'appuie à la fois sur des libertés laissées à l'interprète par le compositeur mais également sur les limites de ces libertés. Notre volonté au travers des partitions interactives est de développer un outils permettant à la fois la composition de

pièces, la définition par le compositeur du cadre de l'interprétation et également l'interprétation de la pièce par le musicien. Il y a donc deux aspects dans les partitions interactives : un aspect compositionnel à destination du compositeur et un aspect exécution des pièces destiné au musicien. Des tentatives de création de pareils outils ont déjà été menées mais les systèmes développés étaient des systèmes "had hoc" créés pour des pièces particulières. Notre objectif est de concevoir un outils général de composition et d'exécution. Il nous faut donc définir à la fois une structure de données permettant la définition et la représentation de l'ensemble des informations constitutives d'une partition interactive et une machine générique interprétant ce type de structures.

Dans cette étude, nous limitons volontairement l'interprétation à la modification des dates de début et de fin de notes, celles-ci pouvant être avancées ou retardées par rapport à ce qui est écrit par le compositeur. A partir de maintenant, nous appellerons "événements", les débuts et de fins de note. Tous les débuts et fins de note ne sont pas décalables et cette possibilité est précisée par l'introduction de points d'interaction dans la partition par le compositeur. Un point d'interaction permet de lier le déclenchement d'un événement à un contrôle extérieur actionné par le musicien au cours de l'exécution, l'événement est alors dit interactif. La définition du cadre de l'interprétation se fait de deux manières. Tout d'abord, pour éviter une désorganisation de sa pièce des suites des décalages, le compositeur aura la possibilité de définir la structure temporelle de la pièce en liant les notes avec les relations temporelles de Allen. Ces relations sont présentées sur la figure 1. L'introduction de ces relations permet

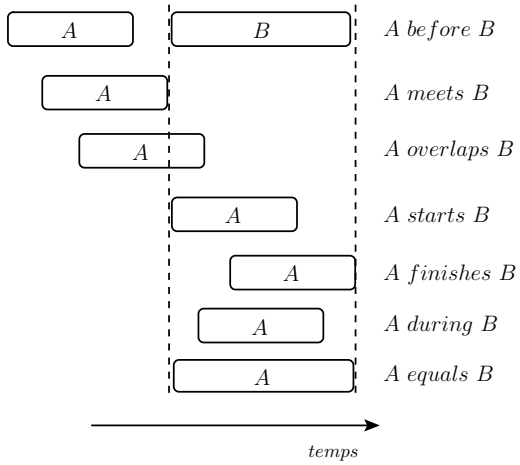


FIG. 1 – Les relations de Allen

de définir un ordre partiel entre les évènements. Cet ordre est partiel car bien sûr l'idée est naturellement de laisser un espace de liberté à l'interprète et à moins de sur-contraindre le système en introduisant des relations entre toutes les notes, des modifications dans l'ordre des événements sont possibles au cours de l'exécution au travers de l'action de l'interprète.

La seconde possibilité offerte au compositeur pour définir le cadre de l'interprétation permet un raffinement de l'organisation temporelle effectuée avec les relations de Allen. Elle consiste en l'introduction d'intervalles. Concrètement

il s'agit de matérialiser l'intervalle de temps qui sépare deux événement et de définir les propriétés de cet intervalle. Soit il est rigide et dans ce cas, la différence de date entre les deux événement doit toujours être maintenue, soit il est souple et dans ce cas sa valeur écrite sur la partition n'a plus d'importance au cours de l'exécution et pourra être modifiée de n'importe quelle manière, enfin un intervalle peut être "semi rigide" ce qui signifie que sa valeur peut-être modifiée pendant l'interprétation mais qu'elle doit rester comprise entre des bornes min et max.

Pour augmenter encore la finesse de composition, nous avons également introduit des événements qui ne sont ni des débuts ni des fins de notes. Ceux-ci sont rattachés à une note mais leur sens musical les relie à un moment particulier de cette note, par exemple, le début d'un crescendo ou tout autre moment particulier du déroulement d'une note. Ils permettent une synchronisation plus fine entre les éléments musicaux de la pièce et également la définition d'intervalles plus fins.

Finalement, une partition interactive est un système contraint dont les variables sont les dates des événements et dont les contraintes sont issues des relations de Allen et des intervalles. Notre objectif est donc trouver des mécanismes permettant de maintenir ces contraintes tant au cours de la composition que de l'exécution. Enfin dans toute cette étude, nous représentons les notes sous forme de boites. Cette convention est issue du formalisme du logiciel de composition Boxes (développé au Scrime) qui accueillera le modèle des relations interactives. Le modèle de partition que nous utilisons comporte également la particularité d'être hiérarchique en ce sens où il existe deux types de notes :

- des notes simples, qui correspondent à des processus musicaux
- des notes complexes qui contiennent d'autres notes

Enfin, comme les paramètres musicaux des notes ne nous intéressent pas pour le moment, ces boites sont considérées comme sans contenu et sont des représentations abstraites. Un exemple de partition interactive est présenté dans la figure 2. Sur cet exemple, on présente les éléments constitutifs d'une partition interactive :

- T_7 est une note complexe
- T_1 et T_2 sont des notes simples
- T_4 est un événement
- T_5 est un point d'interaction
- les intervalles Δ_0, Δ_1 et Δ_6 sont rigides (en gras sur la figure)
- l'intervalle Δ_3 est semi rigide et sa valeur doit être comprise dans l'intervalle $[\Delta_{min},\ \Delta_{max}]$

2 Notre modèle de réseaux de Petri

Comme nous l'avons précisé précédemment, une partition interactive est système contraint dont nous devons assurer le maintien des contraintes. Dans la phase de composition pour laquelle la notion de temps de calcul n'est pas centrale, bien que des temps de calcul trop longs nuiraient lourdement à la fluidité de l'utilisation, il s'agit de la résolution de problèmes de satisfactions de contraintes. Nous avons donc opté pour cette partie pour une solution par propagation (la librairie Gecode développée par Christian Schulte) qui offre des résultats tout à fait satisfaisants.

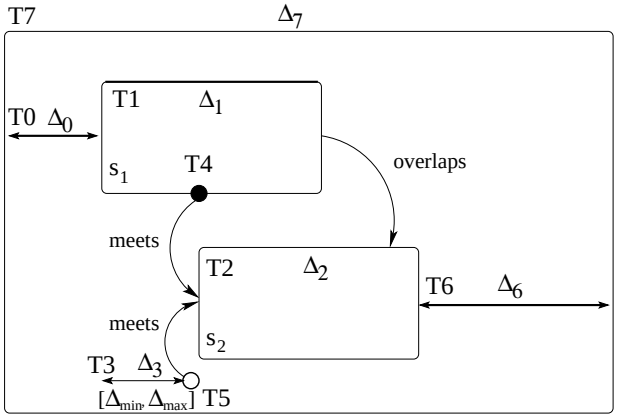


FIG. 2 – Un exemple de partition interactive

En revanche dans le cadre de l'interprétation pour laquelle, il faut que notre système permette au musicien de s'exprimer au travers des libertés offertes par le compositeur tout en maintenant les contraintes formant les limites de ces libertés et pour laquelle la question du temps réel est centrale, un algorithme de satisfaction de contraintes présente des risques de latence trop importante pour être acceptable. Notre choix de structure de données c'est donc porté sur les réseaux de Pétri pour les possibilités qu'ils offrent en ce qui concerne l'ordonnancement et l'exécution de processus parallèles qui doivent se synchroniser à certains moments, ce qui correspond exactement à notre problématique en assimilant les notes à des processus. Tout l'enjeu est maintenant de trouver d'une part le type de réseaux correspondant à notre problème, de trouver un algorithme de transformation des partitions interactives en réseaux qui préserve les propriétés de ces dernières, d'établir des techniques d'analyse des réseaux et enfin de construire un interpréteur de ces réseaux permettant de les exécuter.

Notre approche a été de commencer par chercher des correspondances entre les propriétés des partitions et celles des réseaux que nous cherchons à définir, de trouver des techniques de transformation et par la même d'esquisser un algorithme général de transformation. La caractérisation formelle des réseaux reste encore à déterminer précisément même si des éléments en ce sens commencent à se préciser. Le présent document suivant notre approche, il expose largement les techniques de transformation que nous avons définies avant de tenter une caractérisation des réseaux à partir des propriétés des réseaux produits par les différentes techniques.

La partie statique

Dans un premier temps nous allons considérer les partitions interactives comme étant des partitions statiques, c'est à dire dépourvues de points d'interactions et par la même occasion de contraintes d'intervalles puisque ceux-ci n'ont de sens que dans le cadre où des dates d'événements sont susceptibles d'être modifiées. Ainsi pour l'instant on peut considérer que tous les intervalles des partitions sont rigides. Dans le cadre de partitions statiques, toute la ques-

tion réside dans l'exploitation des informations fournies par les relations de Allen introduites par le compositeur. C'est à dire qu'il nous faut traduire en réseaux de Pétri l'ordonnancement partiel des événement défini au travers de l'ensemble de relations de la partition.

Pour ce faire, nous avons exhibé des transformations dites "élémentaires" qui correspondent à la transformation de configurations simples pour laquelle une seule relation intervient. Ainsi chacune des 7 relation de Allen présentées plus haut possède sa transformation élémentaire. Nous distinguons en outre deux types de transformations :

- celles qui ne nécessitent pas de fusion.
- celles qui nécessitent une fusion

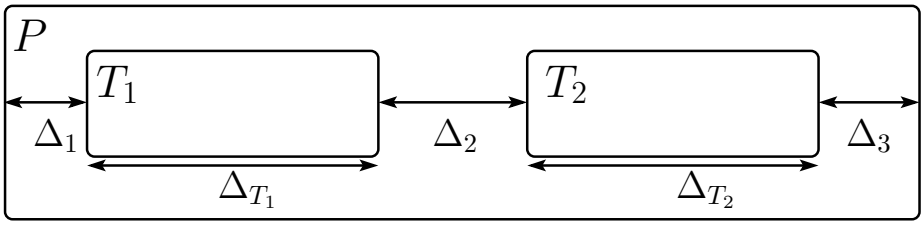
Dans la première catégorie, on retrouve les relations *before*, *during* et *overlaps*, dont les transformations sont présentées dans la figure 3. Il s'agit pour celle-ci d'un simple ordonnancement des événements avec utilisation des intervalles que l'ont peu exhiber dans la partition. Les réseaux correspondant sont des réseaux de Pétri, à savoir qu'une transition qui suit une place d'attente n'est sensibilisée qu'après écoulement du délais correspondant. En outre les événements sont associés aux transitions signifiant ainsi qu'ils sont déclenchés au moment du franchissement de la transition. Il est important de noter qu'entre une note et la note complexe qui la contient il existe une relation *during* implicite. De la même manière, une contrainte intrinsèque à chaque note implique que son début soit toujours avant sa fin.

De manière semblable, on définit des transformations élémentaires pour des relations qui impliquent une "fusion" à savoir une synchronisation entre événements, à savoir *meets*, *starts*, *finishes* et *equals*. Le terme "fusion" caractérise la fusion des transitions associées à chacun des événements à synchroniser qui se retrouvent associés à une même transition.

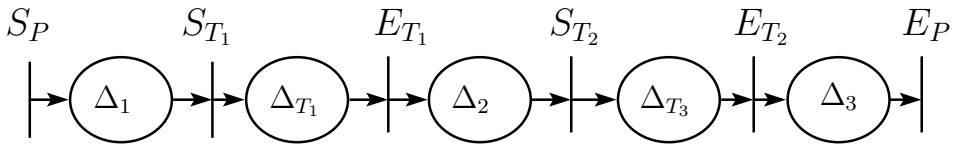
Nous présentons ici l'aglorithme de transformation d'une partition en réseau de Pétri. Son idée est assez simple et est la suivante : créer les transitions associées aux événements de la partition, les relier au début et à la fin de la partition par des places traduisant la relation implicite "during" entre chaque note et la partition et enfin modifier le réseau en fonction de toutes les relations de Allen préciser par le compositeur.

Algorithme 1 (Algorithme de création des réseaux de Pétri)

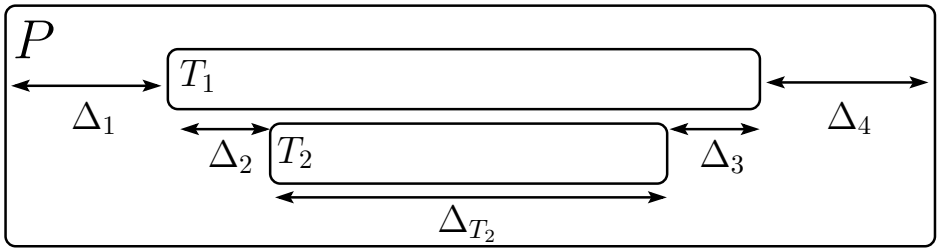
```
petri creer_reseau_partition(partition P)
{
  reseau=nouveau_reseau();

  creer_transition_debut(P,reseau);
  creer_transition_fin(P,reseau);

  for note in Notes(P)
  {
    creer_transition_debut(note,reseau);
    creer_transition_fin(note,reseau);
    ajouter_relation(during,note,P,reseau);
  }
```



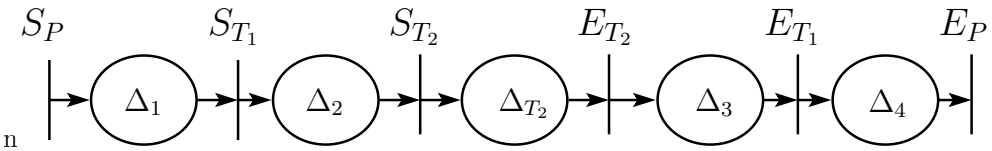
(a) *Before*



(b) Réseau de Petri associé à *Before*

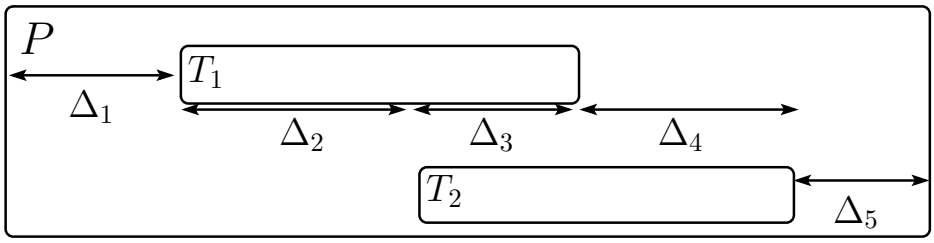


(c) *During*

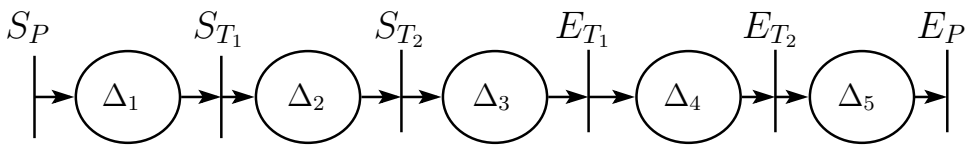


n

(d) Réseau de Petri associé à *During*

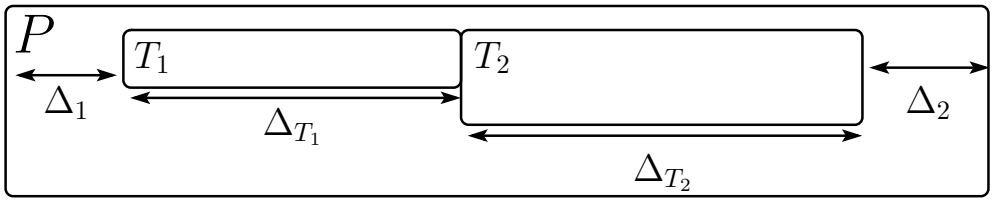


(e) *Overlaps*

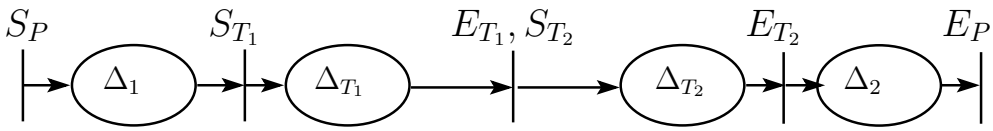


(f) Réseau de Petri associé à *Overlaps*

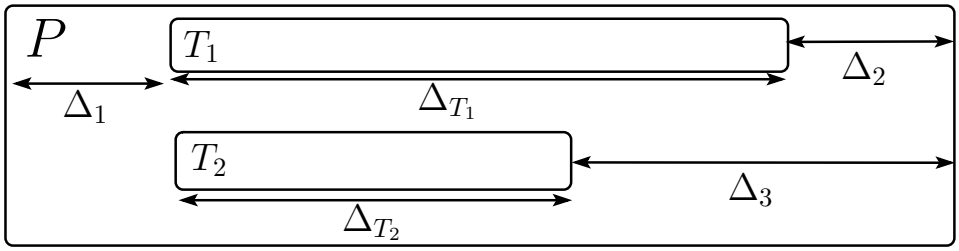
FIG. 3 – L'ensemble des transformations élémentaires ne nécessitant pas de fusion



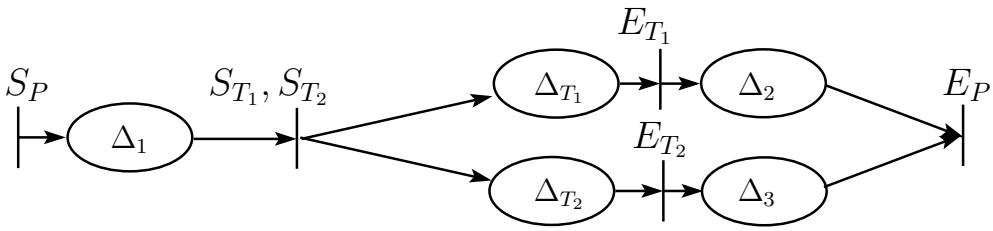
(a) *Meets*



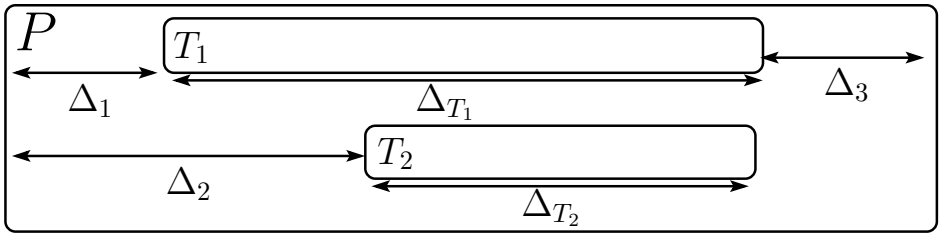
(b) Réseau de Petri associé à *Meets*



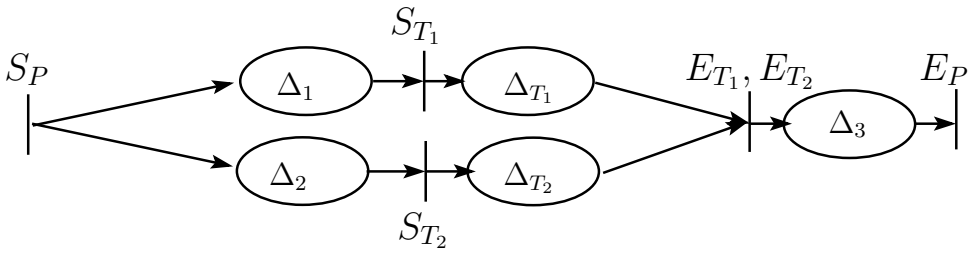
(c) *Starts*



(d) Réseau de Petri associé à *Starts*



(e) *Finishes*



(f) Réseau de Petri associé à *Finishes*

FIG. 4 – L'ensemble des transformations élémentaires nécessitant une fusion

```
  trier(Relations(P));

  for relation in Relations(P)
  {
    note_origine=origine(relation);
    note_destination=destination(relation);
    ajouter_relation(relation,origine,destination,reseau);
  }

  supprimer_arcs_inutiles(reseau);

  return reseau;
}
```

Ainsi, à l'issue de la transformation d'une partition statique, on obtient, un réseau temporisé représentant l'ordre partiel des événements issus de l'introduction des relations de Allen.

3 Les points d'interactions

L'introduction de points d'interaction ne peut être séparée de celle des intervalles. En effet, l'introduction de points d'interaction sans intervalles associés se traite en supprimant le délais précédant la transition correspondant à l'événement interactif et remplacer l'attente de l'écoulement du délais pas l'attente de l'action du contrôle associé au point d'interaction. Un premier problème apparaît : certains délais du réseau ne seront plus vérifié si un contrôle se fait attendre.

Cependant l'absence d'indications du compositeur concernant les propriétés des intervalles, il nous faut faire des choix pour déterminer le comportement adopté par le du système. Ansi, dans le cas d'une introduction d'un ou plusieurs point d'interaction sans introduction d'intervalles associés pour encadrer le déclenchement des événements interactifs nous considérerons d'une part que le déclenchement de l'événement interactif n'est pas borné dans le temps, précisément il peut intervenir dès que les événements explicitement définis comme devant le précéder au moyen des relations de Allen ont été déclenchés et peut être attendu autant que le musicien le désire. On s'aperçoit dans ces conditions que les re-synchronisations de lignes de temps concurrentes contenant des point d'interactions posent des problèmes de respect des délais précédant la transition de re-synchronisation comme le montre l'exemple simple de la figure 5. Pour ces cas, nous adopterons pour les transitons de re-synchronisation la sémantique dite du "et-faible", ce qui signifie que l'on attend que tous les délais soient écoulés avant de sensibiliser la transition quitte à dépasser certains de ces délais. Sur notre exemple, les délais Δ_2 et Δ_3 sont susceptibles d'être allongés, si le contrôle X intervient plus tard que prévu Δ_2 sera allongé, Δ_3 le sera si X intervient plus tôt que prévu.

Finalement, dans ce cas les délais qui ne sont pas susceptibles d'être modifiés gardent leur valeur quoi qu'il arrive et ceux qui sont susceptibles de l'être ne

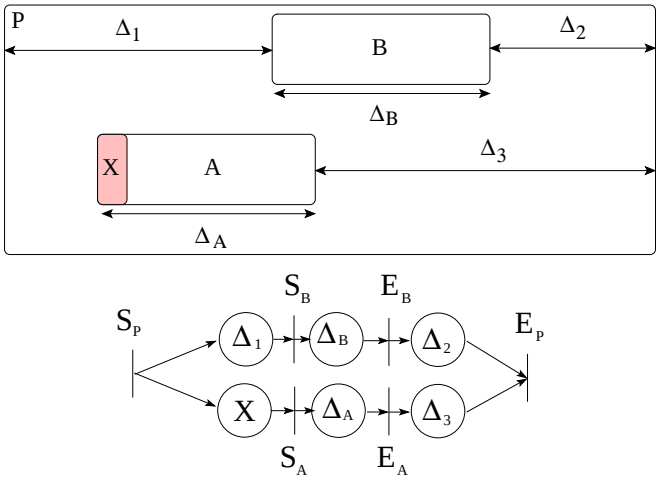


FIG. 5 – Exemple d'introduction d'un point d'interaction sans intervalles

peuvent être qu'allongés. A noter que ce choix d'attitude est arbitraire et qu'on pourrait tout à fait adopter d'autres stratégies. On pourrait par exemple proposer au compositeur le choix d'une stratégie par défaut avant de commencer la composition. On pourrait également détecter automatiquement lors de l'introduction d'un point d'interaction par le compositeur les intervalles susceptibles d'être modifiés pour demander au compositeur de définir les propriétés de ces intervalles.

Réseaux de Pétri colorés

L'introduction conjointe des points d'interactions et des intervalles avec propriété spécifique (rigide, souple et semi-rigide) pose des problèmes sensiblement plus ardus. Notre idée est de traduire au travers des réseaux le système de contraintes issu des partitions pour pouvoir calculer dynamiquement et incrémentalement de nouvelles solutions des systèmes de contraintes et plus particulièrement pour pouvoir calculer de nouvelles valeurs des dates des événements.

Nous illustrons la traduction des systèmes de contraintes dans les réseaux par des exemples qui présentent les différentes difficultés à surmonter. L'exemple présenté dans la figure 6 est celui d'une note complexe dont la durée doit être préservée (la note est rigide) mais qui contient une note (A) dont le début est interactif. La variation de date de déclenchement de cet événement avec ce qui est prévu va nécessairement impliquer :

- un allongement d'intervalle si il intervient plus tôt que prévu
- une compression d'intervalle si il intervient plus tard que prévu

Les intervalles intérieurs sont tous semi-rigides et donc sont tous susceptibles d'être allongés ou compressés dans les limites de valeurs extrêmes.

En outre nous laissons au compositeur le choix de l'ordre de modification des intervalles. Pour cet exemple, le compositeur à choisi un "écrasement par la gauche", c'est à dire que l'ordre de modification des intervalles en cas de décalage du début de A est : Δ_A, Δ_2, Δ_B, Δ_3.

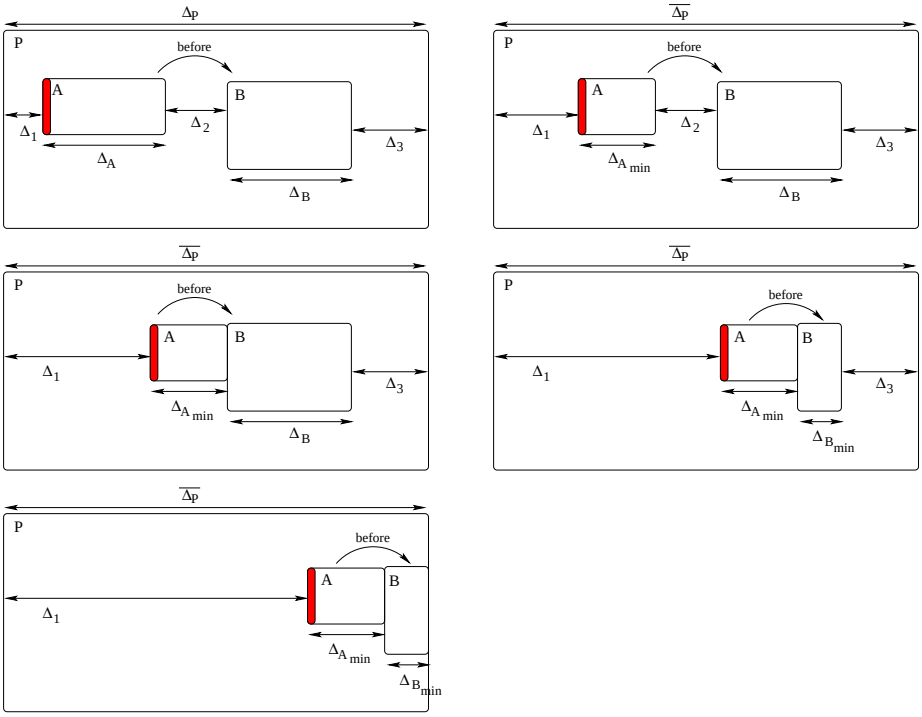


FIG. 6 – Exemple d'écrasement par la gauche

Tous ces choix du compositeur se traduisent par des contraintes qu'il nous faut traduire dans le réseau de Pétri qui devient alors coloré et contient des horloges ou "timewatch".

Nous illustrons l'introduction de ces nouveaux réseaux de Pétri par quelques exemples de partitions accompagnées de leur réseau associé.

Dans l'exemple de la figure 7, il s'agit d'une boite complexe P dont la durée est rigide et qui contient des boites simples A et B, le début de A étant interactif. Les intervalles internes à P, Δ_A, Δ_2, Δ_B, Δ_3 sont tous souples, en outre le compositeur a choisi un écrasement par la droite, ce qui signifie que les modifications des intervalles en cas de décalage du point d'interaction se feront dans cet ordre : $\Delta_3 \Delta_B$, Δ_2, Δ_A.

Comme dans le cas statique, les contraintes d'organisation temporelle issues des relations de Allen sont traduites par un enchaînement des événements reliés par des arcs noires. Les contraintes issues de l'introduction des intervalles avec leur propriétés respectives donnent lieu à la mise en place d'un système permettant de modifier des valeurs de délais dans le futur. Ce système est traduit par un enchaînement d'arcs verts. Pour ces modifications, il faut suivre l'ordre choisi par le compositeur et c'est pourquoi l'enchaînement des arcs verts suit l'ordre de modification. En outre il nous faut également traduire la possibilité d'élargissement de Δ_3. Ce dispositif est complété par des arcs rouges dits arcs "STOP" qui permettent l'arrêt de la décrémentation des valeurs des délais.

Le calcul incrémental des nouvelles valeurs des délais s'effectue comme suit :

– quand la note P commence, d'une part on attend l'arrivée de X et d'autre part on décrémente la valeur de Δ_1 (arc vert). Cette valeur du délais Δ_1

nous est utile pour effectuer l'élargissement de Δ_3. En effet si X intervient plus tôt que prévu, la décrémentation de Δ_1 sera arrêtávant d'atteindre la valeur 0. Or dans le chemin noir d'écoulement du temps musical, la place contenant le délais Δ_1 se trouve avant celle de Δ_3, ainsi le délais Δ_3 sera augmenté de ce qu'il restera de Δ_1 après l'arrivée X.

- si X intervient plus tard que prévu, Δ_1 sera réduit à 0 et la décrémentation de Δ_3 commencera, traduisant l'écrasement de l'intervalle Δ_3.
- en suivant le chemin vert, on décrémente les délais dans l'ordre choisi par le compositeur jusqu'à l'arrivée de X.
- à l'arrivée de X, la pièce se déroule avec les délais issus des décrémentations, ainsi la durée Δ sera respectée.
- si X n'est par intervenu après l'écrasement de tous les délais, ce qui traduit une attente pendant une durée Δ, il nous faut déclencher le début de A pour pouvoir respecter la durée Δ pour P. Pour ce faire, un jeton noir est créé dans une place située avant la transition de début de A. Le déroulement de la pièce se fait donc avec des valeur de délais nul et la fin de P est atteinte immédiatement.

Il est à noter que la transition de début de A est un peu particulière puisque qu'elle n'attend pas la présence de jetons dans les deux places qui la précède mais elle est sensibilisés dès qu'un jeton est produit dans l'une des places qui la précède.

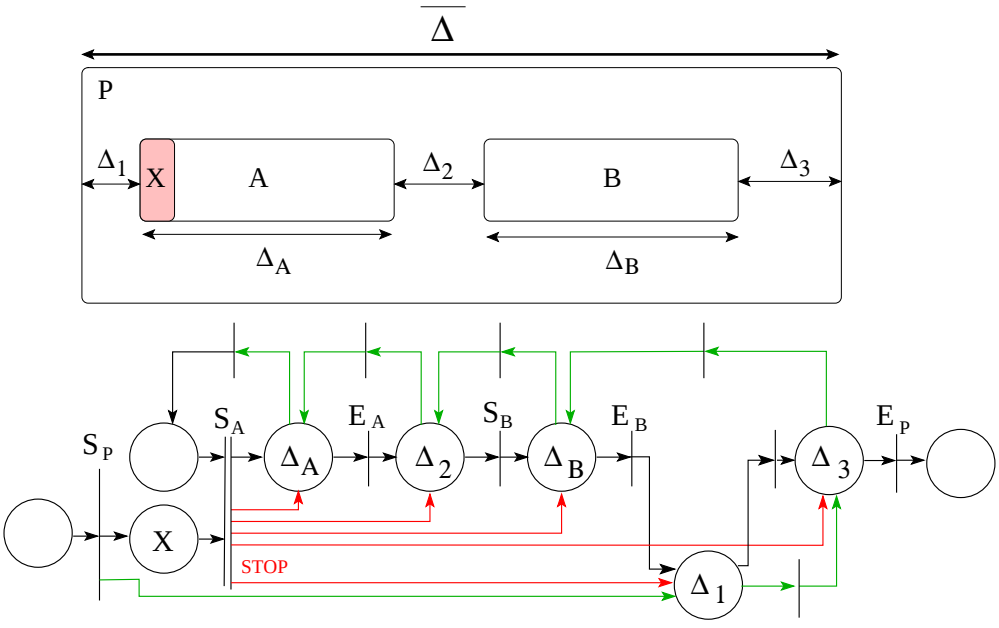


FIG. 7 – Exemple d'une note complexe de durée rigide contenant un point d'interaction et uniquement des intervalles souples

Sur le second exemple présenté par la figure 8 nous montrons le cas d'une note complexe P de durée rigide Δ contenant un point d'interaction mais également des intervalles rigides Δ_A et Δ_B et des intervalles semi-rigides Δ_1, Δ_2 et Δ_3. Ainsi on a :

- $\Delta_{1_{min}} \leq \Delta_1$
- $\Delta_{2_{min}} \leq \Delta_2 \leq \Delta_{2_{max}}$
- $\Delta_{3min} \leq \Delta_3 \leq \Delta_{3_{max}}$

Le compositeur a choisi pour cet exemple un écrasement par la gauche, ce qui signifie que l'ordre de modification des intervalles est : Δ_2, Δ_3.

La difficulté supplémentaire par rapport à l'exemple précédent est que l'on ne peut pas modifier les intervalles comme on le désire Pour se prémunir contre des réduction trop importante, on introduit des places contenant les délais minimum et exclues du chemin vert de décrémentation des valeurs de délais. A l'inverse on ne peut pas élargir les intervalles autant qu'on le souhaite pour absorber une anticipation de l'arrivée de X. En particulier, Δ_2, premier intervalle concerné par un élargissement ne peut dépasser la valeur $\Delta_{2_{max}}$. Pour éviter un élargissement trop important, on divise Δ_1 en :

$$\Delta_1 = \Delta_{1_{min}} + (\Delta_{2_{max}} - \Delta_2) + \epsilon$$

On suppose en outre que $\epsilon \leq \Delta_{3_{max}} - \Delta_3$ sinon le problème est inconsistant. Ou plus exactement, si cette inéquation n'est pas vérifiée, le musicien pourra donner au cours de l'exécution une valeur à Δ_1 pour laquelle il n'existe pas de solution au problème de contraintes Ces valeurs de Δ_1 sont atteintes en retardant tellement l'arrivée de X de telle manière, que l'élargissement imposé á Δ_2 et Δ_3 dépasse leur capacités. Ce vérifications d'inégalités sont à faire au moment de la composition. L'éclatement de Δ_1 se traduit par la présence de 3 places correspondant au 3 valeurs exposées plus haut. Le calcul incrémental est le suivant :

- on attend l'écoulement du délais $\Delta_{1_{min}}$ pour s'assurer que $\Delta_{1_{min}} \leq \Delta_1$
- on commence par décrémenter ϵ ansi, si X intervient tellement tôt que des élargissements de Δ_2 et Δ_3 sont nécessaires, alors Δ_2 prendra la valeur $\Delta_{2_{max}}$ et c'est Δ_3 qui s'élargira pour absorber le reste de l'augmentation de Δ_1.
- quand *espilon* est réduit à 0, cela signifie qu'a partir de ce moment, Δ_2 est capable d'absorber seul une augmentation de Δ_1. Ainsi on décrémente la valeur $\Delta_{2_{max}} - \Delta_2$.
- quand $\Delta_{2_{max}} - \Delta_2$ est réduite à 0. Cela signifie qu'on a atteint une durée Δ_1 depuis la début de P et qu'á partir de maintenant, X arrivera plus tard que prévu et que par conséquent, on doit décrémenté les valeurs des délais comme dans l'exemple précédent.

Le troisième exemple de la figure 9 présente le cas d'une note complexe de durée fixe contenant 2 points d'interactions , des intervalles rigides et des intervalles semi-rigides. La difficulté vient ici du fait que les deux points d'interaction peuvent modifier les valeurs de mêmes intervalles en l'occurrence Δ_3.

Au final, les réseaux que l'on obtient sont temporisés, colorés et contiennent des "timewatchs" (horloge que l'on peut arrêter). A noter que tous les cas possibles n'ont pas été traités ici notamment le cas d'une note complexe semi-rigide qui nécessite quelques modifications avec notamment l'entrée de l'intervalle concernant la durée de la note complexe dans l'ordre de modification déterminé par le compositeur. On peut remarquer en outre qu'un travail de vérification devra être effectué au moment de la composition. Au delà de systèmes inconsistants obtenus par des configurations de relations de Allen impossibles, les propriétés des intervalles sont étroitement liées entre elles et comme l'a vu certains choix

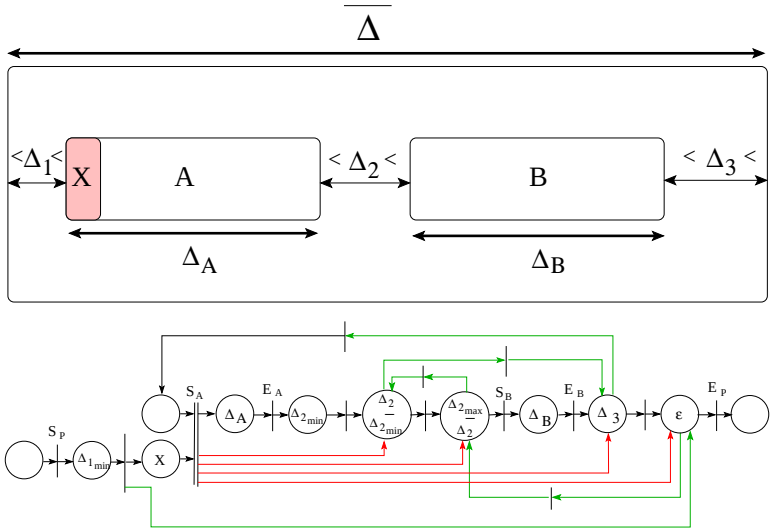


FIG. 8 – Exemple d'une note complexe de durée rigide contenant un point d'interaction et des intervalles rigides et semi-rigides

sans rendre le système de contraintes directment inconsistant peuvent permettre, le choix de valeurs pour des variables pour lesquels il n'existe pas de solutions au système. On peut en revanche arguer qu'un système malin pourrait ne proposer au compositeur que des valeurs d'intervalles compatibles au moment des choix des bornes des intervalles. Ceci éviterai de faire des vérifications en autirisant que de "bonnes" valeurs.

4 Méthode par propagation

La solution précedente basée sur des réseaux de Pétri colorés et temporisés nous est apparue comme relativement ardue à généraliser pour des partitions quelconques, la construction de ces réseaux et leur mise en oeuvre est suffisamment complexes pour que nous cherchions à développer une autre stratégie pour prendre en compte les contraintes sur les intervalles. En particulier nous avons chercher à différer le calcul des valeurs des intervalles. Dans la solution avec réseaux colorés nous calculons les modifications des valeurs des Δ à chaque instant au travers du système de "timewatchs". A l'inverse notre objectif est de calculer les valeurs de delais au moment où nous en avons besoin. Nous nous sommes donc tournés vers un système de propagation des décallages introduits par les points d'interaction. Pour cette solution nous utilisons deux structures :

- un réseau de Pétri
- un graphe de contraintes

Le réseau de Pétri nous permet de représenter les relations de Allen et est donc construit comme dans pour la partie statique de la solution précédente. Ses places contiennent donc les delais écrits par le compositeur et les attentes de déclenchement de contrôles discrets pour les point d'interaction.

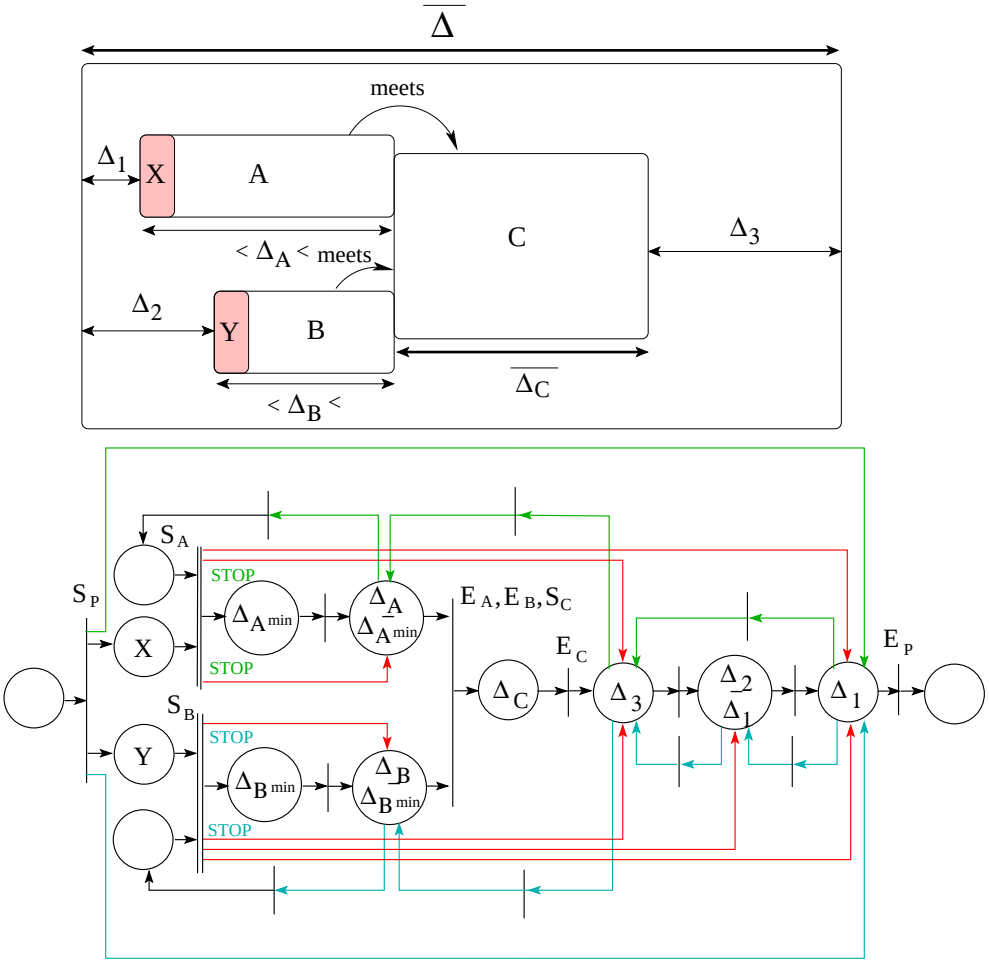


FIG. 9 – Exemple de boite complexe contenant 2 points d'interaction

Le graphe de contraintes représente l'ensemble des contraintes sur les intervalles et les ordres d'écrasement. Il s'agit donc d'un graphe orienté dans lequel chaque nœud représente une variable (valeur d'intervalle).

Nous présentons l'idée de cet algorithme sur l'exemple de la figure 10. Dans cet exemple, une boite complexe P de durée rigide contient des boites simples A et B, le début de A étant interactif. Les intervalles internes à P sont semi-rigides et le compositeur a choisi un écrasement par la droite.

Le graphe contient donc un nœud pour chaque intervalle, ceux-ci étant reliés par des arcs créant ainsi des chemins représentant les ordres d'écrasement. A noter que sur l'exemple le nœud de Δ_P est lui même inclus dans un chemin représentant un ordre d'écrasement impliquant des variables de durées au niveau de la boite englobante M.

La suite de figures présente le fonctionnement de l'algorithme lors d'un décalage du point d'interaction. Le système consiste à propager ce décalage au travers du chemin de priorité en faisant en sorte que chaque variable soit modifiée

pendant cette propagation. Concrêtement, pendant la propagation lorsqu'une variable est accédée elle reçoit un résidu du décalage d'origine et est modifiée au maximum des capacités de son domaine de valeur. Si après cette modification, il reste encore un peu de résidu du décalage, celui-ci continue d'être propagé sur le chemin.

Ainsi sur un chemin $[\Delta_1 ... \Delta_i ... \Delta_n]$, en notant Δ, le décalage reçu par le point d'interaction et δ_i le décalage reçu par le variable Δ_i, on a :

$$\Delta = \sum_{i=1}^{n} \delta_i$$

Si un des Δ_i correspond à la durée d'une boite complexe, on propage le décalage qu'elle reçoit à ses durées internes en utilisant la strcture hiérarchique du graphe.

En outre, lorsqu'un décalage se produit, il nous faut réduire le domaine des autres variables en fonction de ce décalage. Malheureusement pour cette opération de réduction de domaine, notre graphe n'est pas très approprié et nous devons nous appuyer sur un graphe de contrainte "classique", la multiplication des structures nous poussent à abandonner notre type de graphe et à n'utiliser qu'un graphe de contraintes représentant les contraintes en incluant l'ordre des écrasements au travers d'un ensemble de contraintes pondérées.

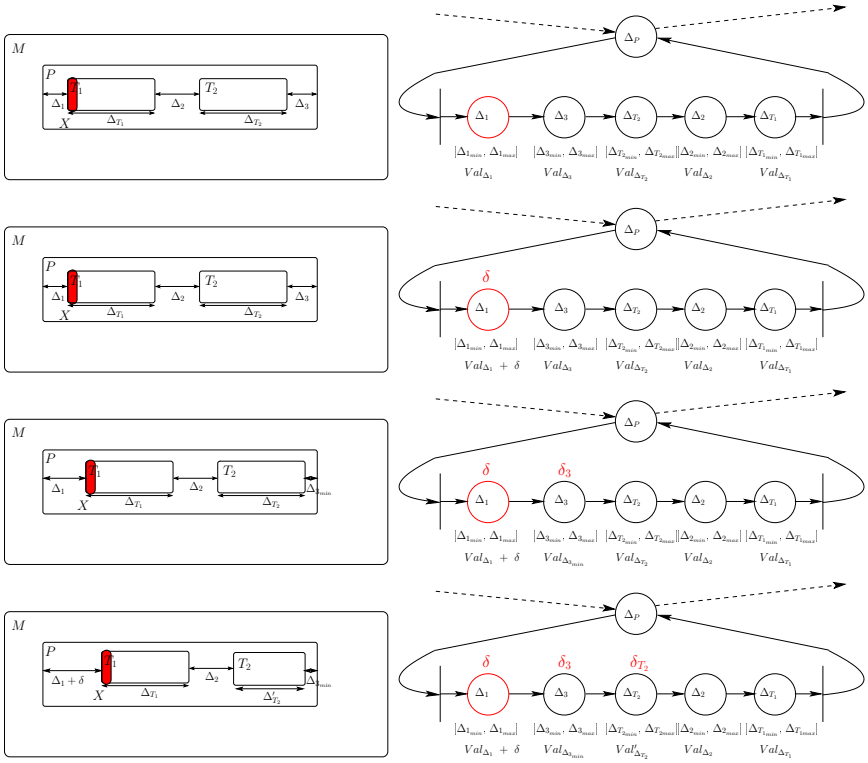


FIG. 10 – Exemple d'utilisation de graphe de contraintes

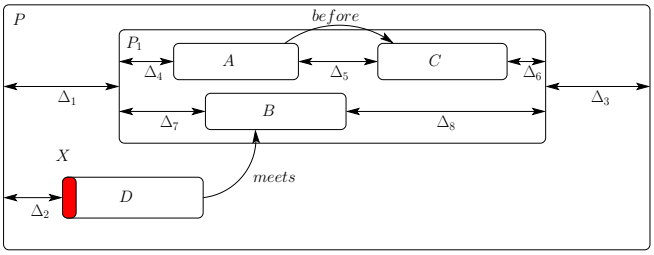


FIG. 11 – Un exemple qui se passe mal

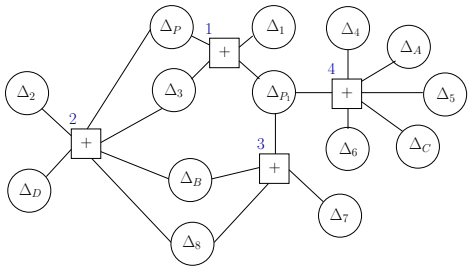


FIG. 12 – Le graphe de contraintes associé aux contraintes de l'exemple

5 Réduction de domaine par propagation

Dans cette section, nous nous intéressons à l'utilisation d'un graphe de contraintes pour réduire le domaine des variables par propagation avec un algorithme de type *Indigo*. Ce type de techniques est assez bien connu, et ne fonctionne pas dans tous les cas. En particulier, la présence de cycles dans le graphe pose des problèmes de consistence puisque la consistence locale assurée par cet algorithme n'est pas équivalente à la consistence globale. Nous illustrons ces situations par l'exemple qui se passe mal présenté par la figure 5

Supposons que le compositeur donne les contraintes d'intervalles suivantes :

- $\Delta_P = 100$
- $\Delta_1 = 30$
- $\Delta_D = 10$

De part la hiérarchie du système, nous avons les équations suivantes :

$$\Delta_P = \Delta_1 + \Delta_{P_1} + \Delta_3$$

$$\Delta_P = \Delta_2 + \Delta_D + \Delta_B + \Delta_8 + \Delta_3$$

$$\Delta_{P_1} = \Delta_7 + \Delta_B + \Delta_8 + \Delta_3$$

$$\Delta_{P_1} = \Delta_4 + \Delta_A + \Delta_5 + \Delta_C + \Delta_6$$

Le graphe de contraintes associé à cet ensemble de contraintes est présenté sur la figure 12.

Dans ses conditions, en applicant un algorithme de réduction de domaines par propagation sur le système de contraintes formé par les 3 équations préce-

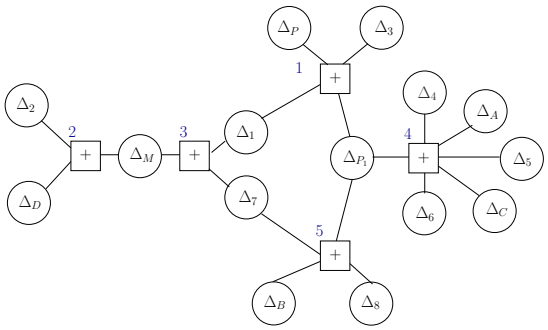


FIG. 13 – Le graphe de contraintes de l'exemple après changement de contraintes

dentes on a :

$$\text{Au départ : } \Delta_1 = [10,10], \Delta_2 = [0,\infty], \Delta_3 = [0,\infty], \Delta_4 = [0,\infty]$$

$$\Delta_5 = [0,\infty], \Delta_6 = [0,\infty], \Delta_7 = [0,\infty], \Delta_8 = [0,\infty]$$

$$\Delta_P = [100,100], \Delta_{P_1} = [0,\infty]$$

$$\Delta_A = [0,\infty], \Delta_B = [0,\infty], \Delta_C = [0,\infty], \Delta_D = [10,10]$$

$$\text{Avec 1 : } \Delta_{P_1} + \Delta_3 = 70 \; d'o \; \Delta_{P_1} = [0,70], \Delta_3 = [0,70]$$

$$\text{Avec 2 : } \Delta_2 + \Delta_7 + \Delta_B + \Delta_8 + \Delta_3 = 90$$

$$\text{d'où : } \Delta_2 = [0,90], \Delta_7 = [0,90], \Delta_B = [0,90], \Delta_8 = [0,90]$$

Avec 3 : $\Delta_7 + \Delta_B + \Delta 8 = \Delta_{P_1} = [0,70]$ d'où $\Delta_7 = [0,70], \Delta_B = [0,70], \Delta_8 = [0,70]$

Comme les domaines de Δ_7, Δ_B et Δ_8 ont changé, on réapplique 2 mais les domaines restent inchangés. On a donc à la fin de l'algo de réduction de domaine :

$$\Delta_1 = [30,30], \Delta_2 = [0,90], \Delta_3 = [0,70], \Delta_4 = [0,70]$$

$$\Delta_5 = [0,70], \Delta_6 = [0,70], \Delta_7 = [0,70], \Delta_8 = [0,70]$$

$$\Delta_P = [100,100], \Delta_{P_1} = [0,70]$$

$$\Delta_A = [0,70], \Delta_B = [0,70], \Delta_C = [0,70], \Delta_D = [10,10]$$

Or il apparaît rapidement que ce système n'est pas consistent car le domaine de Δ_2 n'est clairement pas assez réduit. En effet avec une valeur de Δ_2 strictement inférieure à 20, le système n'a pas de solution, par exemple avec $\Delta_2 = 10$ on a $\Delta_7 = -10$. Le domaine de Δ_2 aurait du être [20,90].

Pourquoi cela ne marche pas ?

Ce problème de réduction de domaine est du à la présence de cycles dans le graphe de contraintes associé au problème consistué des 4 contraintes utilisées précédemment. En effet comme on peut le voir sur la figure 12 On peut chercher à éviter ces cycles en changeant notre système de contraintes. Précisément en utilisant l'équation $\Delta_1 + \Delta_7 = \Delta_2 + \Delta_D$ impliquée par la relation *meets* et en introduisant une variable Δ_M telle que $\Delta_M = \Delta_1 + \Delta_7$, on obtient le graphe de la figure 13. En applicant l'algorithme de propagation, cela fonctionne mais il reste encore un cycle dans ce graphe et par conséquent, il se peut qu'un changement de préférences du compositeur nous conduisent à une autre situation ou l'algo ne fonctionne pas.

Certaines pistes sont envisageables pour chercher à avoir une équivalence entre consistence locale et consistence globale :

- Utiliser l'évolution du temps qui va contraindre de plus en plus le système puisque de plus en plus de variables verront leur domaine réduit à un singleton ce qui pourrait faire apparaître des ensembles de variables fixées cassant les cycles.
- Utiliser la consistence locale d'ordre "k" pour avoir l'équivalence avec la consistence globale
- Utiliser d'autres caractéristiques du système pour assurer cette équivalence

6 Partitions ouvertes

Nous avons également envisager la possibilité pour le compositeur de créér des partitions "ouvertes" dans lesquelles le musicien doit effectuer des choix de parties qu'il désire jouer tout en abandonnant d'autres. Il ne n'existe pas de difficultés particuliéres dans l'introduction de ces possibilités comme en témoigne la figure 14.

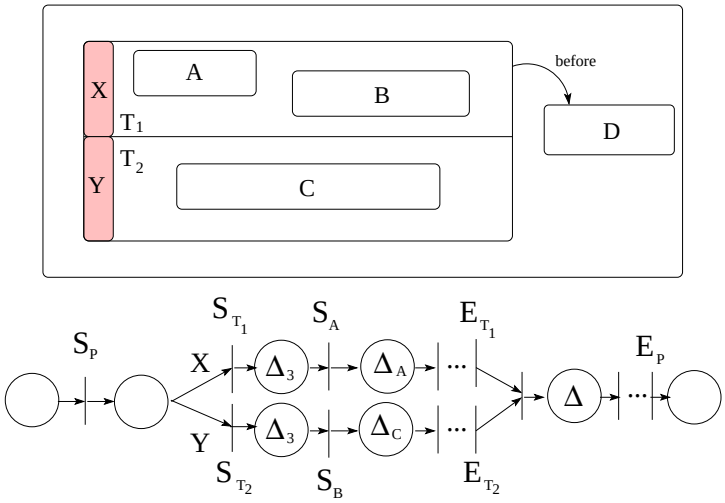


FIG. 14 – Exemple de partition avec choix

7 Contraintes globales

Enfin, nous souhaiterions pouvoir introduire dans les partitions des contraintes "globales" impliquant toutes les notes. L'idée seraient de stocker ces contraintes dans un magasin qui serait interrogé avant le déclenchement des événements. Plusieurs stratégies sont alors envisageables dans le cas d'une violation de contrainte :

- on attend que la contrainte ne soit plus violée par le déclenchement de l'événement et on déclenche l'événement
- on ne joue pas la note
- dans le cas de contraintes sur le contenu, on peut modifier le contenu de la note á l'origine du viol de la contrainte pour que son déclenchement ne viole plus de contraintes du magasin.

Cependant, l'introduction d'un tel système soulève d'épineux problèmes concernant des situations de blocages totales du déroulement de la partition en cours d'exécution et donc implique un travail de vérification préalable.

En outre le premier choix de stratégie (attendre que la contrainte ne soit plus violée avant de déclencher l'événement) semble incompatible avec les contraintes sur les intervalles dans la mesure où les situations d'attente qu'elle implique risquent de briser des contraintes sur les intervalles.

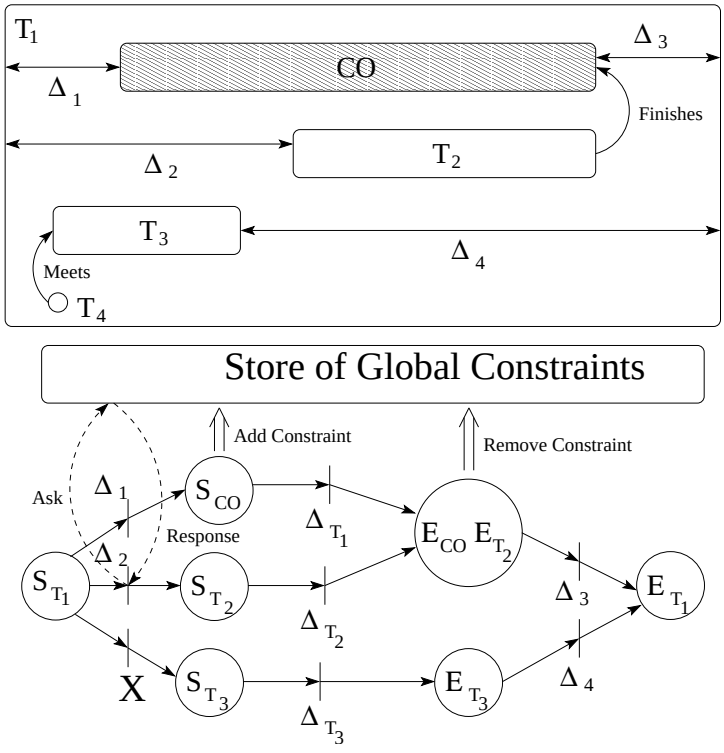


FIG. 15 – Illustration du système avec contraintes globales