



HAL
open science

Ordonnancement et planification sous contraintes temporelles et probabilistes

Bassam Baki

► **To cite this version:**

Bassam Baki. Ordonnancement et planification sous contraintes temporelles et probabilistes. Majec-STIC 2005: Manifestation des Jeunes Chercheurs francophones dans les domaines des STIC, IRISA – IETR – LTSI, Nov 2005, Rennes, France. pp.323-330. inria-00000669

HAL Id: inria-00000669

<https://inria.hal.science/inria-00000669>

Submitted on 14 Nov 2005

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Ordonnancement et planification sous contraintes temporelles et probabilistes

Bassam BAKI

GREYC- Université de Caen
Campus II- Bd. Maréchal Juin
14032 Caen Cedex
bbaki@info.unicaen.fr

Résumé : Nous nous intéressons à la planification et l'ordonnancement temporels sous incertitude qui permettent de résoudre un ensemble des buts.

Nous proposons une approche de planification temporelle permettant aux tâches de respecter des contraintes temporelles et de précédence. Dans ce problème, chaque tâche possède un ensemble de contraintes temporelles, un ensemble de probabilités d'exécution et un ensemble de coûts. Une relation de précédence relie les tâches dans un graphe ET/OU. Grâce à une propagation temporelle à travers le graphe, nous calculons les intervalles d'exécution des tâches. Puis, nous choisissons le meilleur plan pouvant être exécuté en respectant toutes les contraintes. Le problème que nous traitons dans cet article combine deux types de planification : temporelle et probabiliste.

Mots-clés : Planification, Ordonnancement, Raisonnement temporel, Incertitude, Aide à la décision et Intelligence Artificielle.

1 INTRODUCTION

La planification consiste en la recherche d'une séquence d'opérations permettant de passer d'un état initial à un état final souhaité. Un problème d'ordonnancement consiste à organiser dans le temps un ensemble d'activités, de façon à satisfaire un ensemble de contraintes et optimiser une ou plusieurs fonctions objectifs. En d'autres termes, la planification consiste à déterminer les tâches à exécuter et l'ordonnancement consiste à déterminer quand exécuter ces tâches.

Le contrôle d'exécution de plans est un problème particulièrement difficile lorsqu'il doit être effectué à bord de systèmes autonomes tels que des robots. Un tel système doit disposer de processus de délibération pour engendrer des plans qui réalisent les buts de la mission tout en respectant des délais et des contraintes de précédence. Plusieurs formalismes et algorithmes ont été développés pour traiter des problèmes de planification tenant compte de contraintes temporelles. De tels problèmes de planification existent dans de nombreuses applications réelles telles que le transport, la robotique, le domaine médical, les services d'urgences, etc.

Le premier système qui a introduit la représentation des actions par pré-conditions/effets est STRIPS [FN71]. Depuis, plusieurs formalismes et algorithmes ont été

développés pour traiter des problèmes de planification tenant compte de contraintes temporelles.

Le domaine de la planification temporelle est spécifié par l'environnement de la planification (un ensemble d'états, un ensemble d'actions, un état initial, un ensemble de buts et un ensemble de contraintes de précédence) et les caractéristiques temporelles (durées, dates de début, dates de fin, temps limites,...). La plupart des formalismes [DMP91] permettent la représentation de contraintes temporelles portant, en général, sur la durée d'exécution des tâches. Cependant d'autres types de contraintes peuvent être envisagés. De plus, les applications réelles présentent une certaine incertitude sur le temps. Par exemple, dans un problème de transport, la durée d'un trajet n'est pas connue avec exactitude. Il est donc important de pouvoir raisonner sur le temps et sous incertitude.

Différentes approches de planification sous incertitude temporelle ont été développées [MMV01], [DKS02] mais elles ne considèrent pas l'incertitude sur les durées d'exécution. Les études récentes en planification concernent de plus en plus des problèmes réels et traitent plusieurs sortes de contraintes [BDM⁺02], [TVP03], [PG02]. Les plans générés par ces systèmes peuvent être considérés comme une extension des plans classiques. IxTeT et parcPlan [GL94], [LI04] sont des planificateurs qui traitent des données temporelles dans la gestion de temps qualitative (ordre et relation utilisées dans la planification) et dans les contraintes quantitatives (temps et durées utilisés dans l'ordonnancement). La représentation du temps dans IxTeT est celle de l'algèbre d'instant. Ces instants sont des variables symboliques sur lesquelles sont basées les contraintes temporelles. Comme IxTeT, en ce qui concerne la distance entre les instants temporels, notre planificateur travaille sur des contraintes symboliques (précédence) et numériques (intervalles $[F, I^+]$). Mais de plus, il prend en considération les coûts et les probabilités d'exécution des plans dans un graphe ET/OU. Néanmoins il ne traite pas le cas de partage des ressources.

Dans [DHW94] et [KHW94], les auteurs présentent un planificateur appelé "C-Buridan" qui construit un plan avec des actions probabilistes (productrices) d'information et une exécution contingente. Leur algorithme est

sûr (s'il arrive à son terme, la solution g en er ee est suffisante pour atteindre le but) et complet (il trouve la solution si elle existe), mais ce planificateur ne prend pas en consid eration les contraintes temporelles locales et globales, ni les contraintes de pr ec edence, ni le d elai entre les t aches. Ses solutions sont satisfaisantes plut ot qu'optimales.

Dans cet article, nous consid erons un syst eme de planification et d'ordonnancement qui g ere des contraintes temporelles complexes sous incertitude. Notre probl eme est repr esent e par un graphe ET/OU o u nous voulons trouver un plan des t aches qui satisfait toutes les contraintes (de temps, de pr ec edence et de co ut) et qui a une grande probabilit e d' tre ex ecut e avec temps et co uts r eduits. A noter que le probl eme de recherche d'un plan optimal a  et e d emontr e NP-complet. Nous nous int eressons   trouver des solutions satisfaisantes plus qu'optimales. Dans cet article, nous consid erons qu'un agent doit ex ecuter un ensemble de t aches reli ees ensemble dans un graphe acyclique orient e tel que les n oeuds correspondent aux t aches et les ar etes sont des contraintes de pr ec edence.  tant donn e un ensemble de t aches, un ensemble de donn ees pour chaque t ache (une date de d ebut, une date de fin, un ensemble de dur ees, un ensemble de probabilit es et un ensemble de co uts) une relation de pr ec edence et de d elais entre les t aches, notre approche permet   l'agent de d eterminer l'ensemble des t aches   ex ecuter de telle fa on qu'en ex ecutant ces t aches, on atteint le but en respectant toutes les contraintes temporelles et de pr ec edence. Autrement dit,  tant donn es : la description du domaine (le graphe, les donn ees temporelles, les co uts, les probabilit es,...), un ensemble d' tats initiaux, un ensemble de buts   r esoudre, et un ensemble de contraintes correspondant   un ordre partiel que l'on souhaite retrouver dans l'ordre de r ealisation des buts, notre approche permet de d eterminer l'ensemble des plans solutions puis par affinements successifs, elle trouve les plans admissibles et parmi ces plans elle choisit, selon un op erateur de pr ef erence, le meilleur plan   ex ecuter.

Notre approche utilise les techniques de propagation de contraintes temporelles [BW00] qui a pour but de simplifier la r esolution d'un probl eme. Pratiquement, cela peut consister   retirer du domaine des variables de d ecision, les valeurs qui n'appartiennent   aucune solution. Ce filtrage  vite de nombreuses tentatives de r esolution vou ees   l' chec.

Une particularit e importante du probl eme consid er e est que les t aches sont repr esent ees dans un graphe ET/OU et que les dur ees des t aches sont pond er ees par des probabilit es d'ex ecution. Cela exprime notamment des incertitudes sur la connaissance exacte de la dur ee d'ex ecution des t aches qui ne sera r eellement connue qu'  l'ex ecution effective. Dans cet article, nous supposons qu'une t ache s'ex ecute pendant l'une de ses dur ees avec la probabilit e donn ee pour cette dur ee, et si pour une raison quelconque la t ache ne r eussit pas   s'ex ecuter en l'une de ses dur ees, on se trouve dans un cas d' chec total o u il faut arr eter l'ex ecution du plan choisi et demander une replanification.

Ainsi dans le domaine de l'Intelligence Artificielle, notre planificateur est le premier qui traite des contraintes complexes parmi lesquelles les contraintes temporelles et les probabilit es d'ex ecution.

Dans la section suivante, nous d crivons quelques d efinitions g en erales dans la section 2. La section 3 d crit le probl eme   r esoudre. Dans la section 4, nous d crivons la propagation des intervalles d'ex ecution puis la propagation des probabilit es dans la section 5. L'algorithme utilis e pour trouver le meilleur plan est pr esent e dans la section 6. Nous analyserons notre approche dans la section 7. Nous pr esenterons des tests exp erimentaux dans la section 8 et nous concluons dans la section 9.

2 PR ELIMINAIRES

Dans cette section, nous d crivons la notion d'un agent et d'un graphe de contraintes temporelles.

2.1 Agent de planification

Le but de l'agent de planification est de choisir, g erer et ordonner un sous-ensemble de t aches   ex ecuter parmi un ensemble de t aches repr esent ees dans un graphe acyclique.

D efinition 1 On appelle donn ees associ ees   une t ache t la liste $\langle I_t^-, I_t^+, \Delta_t, Pr_t, C_t \rangle$ o u :

- I_t^- est la date la plus pr ecoc e   laquelle la t ache t peut commencer son ex ecution et I_t^+ est la date la plus tardive   laquelle la t ache t doit  tre termin e de sorte que $[I_t^-, I_t^+]$ est la fen tre temporelle d'ex ecution de t ;
- $\Delta_t = \{d_t^1, d_t^2, \dots, d_t^m\}$ est l'ensemble des dur ees d'ex ecution possibles de la t ache t ;
- $Pr_t = \{pr_t^1, pr_t^2, \dots, pr_t^m\}$, o u pr_t^j est la probabilit e que la t ache t soit ex ecut e en la dur ee d_t^j , on a par cons equent $0 \leq pr_t^j \leq 1$ et $\sum pr_t^j = 1$;
- $C_t = \{c_t^1, c_t^2, \dots, c_t^m\}$, o u c_t^j est le co ut (en ressources, temps, argents, etc.) de l'ex ecution de la t ache t en la dur ee d_t^j .

2.2 Contraintes de pr ec edence

Dans le monde r eel, la possibilit e d'ex ecuter une t ache peut d ependre de plusieurs conditions, comme le temps et les ressources disponibles ou l'ex ecution d'autres t aches. On consid ere que les contraintes de pr ec edence auparavant sont de la forme d'une relation d'ordre partiel sur un ensemble de t aches T . Dans ce cas, entre deux t aches quelconques il existe une relation de pr ec edence ou il n'existe pas de relation qui les relie. On  tablit une distinction entre deux sortes de contraintes de pr ec edence :

1. *Contraintes de Pr ec edence Conjonctives* : le cas o u un groupe de t aches ind ependantes doit  tre ex ecut e pour permettre l'ex ecution des certaines autres t aches. Si les t aches t_1, t_2, \dots, t_k doivent toutes  tre ex ecut ees avant l'ex ecution d'une t ache t , on  crit symboliquement : $[t_1, t_2, \dots, t_k] \rightarrow t$.
2. *Contraintes de Pr ec edence Disjonctives* : le cas o u une seule t ache parmi un groupe de t aches doit  tre ex ecut ee avant que la t ache en question ne puisse  tre ex ecut ee. S'il suffit qu'une t ache

parmi les tâches t_1, t_2, \dots, t_k s'exécute pour que la tâche t puisse s'exécuter, on écrit symboliquement : $t_1|t_2|\dots|t_k \rightarrow t$.

À noter que parfois on doit attendre un peu de temps après la fin de l'exécution d'une tâche et avant le début de l'exécution de la tâche suivante.

On désigne ce temps de délai entre les tâches, et t_j par δ_{t_i, t_j} .

Nous pouvons aussi représenter ce délai par une tâche intermédiaire avec une durée égale à ce délai d'attente ($\Delta_{t_k} = \delta_{t_i, t_j}$) tel que $t_i \rightarrow t_k \rightarrow t_j$, $c_{t_k} = 0$, $I_{t_k}^- = I_{t_i}^- + \min\{\Delta_{t_i}\}$, $I_{t_k}^+ = I_{t_i}^+ + \delta_{t_i, t_j}$ et $pr_{t_k} = 1$. Cette représentation simplifie le calcul des intervalles temporels d'écrits dans la section 4 mais complique la recherche des plans dans le graphe puisqu'il y aura plus de tâches à explorer.

2.3 Le graphe de contraintes temporelles

Le graphe de contraintes temporelles associée à un agent décrit les contraintes temporelles sur les tâches et les relations de précédence entre elles.

Définition 2 Un graphe de contraintes temporelles est un graphe simple orienté acyclique $G = (T, E)$ où l'on se donne de plus :

- pour chaque tâche $t \in T$, la liste $\langle I_t^-, I_t^+, \Delta_t, Pr_t, C_t \rangle$ des données associées à t ;
- un ensemble de contraintes de précédence conjonctives $[t_1, t_2, \dots, t_k] \rightarrow t$;
- un ensemble de contraintes de précédence disjonctives $t_1|t_2|\dots|t_k \rightarrow t$;
- un ensemble de contraintes de délai $\{\delta_{t, t'}, t \in T, t' \in T, t \rightarrow t'\}$

Il est à noter aussi que les nœuds qui forment l'ensemble T sont divisés en trois sous-ensembles $T = T_I \cup T_M \cup T_F$ de tâches tels que T_I est l'ensemble de tâches qui n'ont pas de prédécesseurs (les tâches initiales), T_M est l'ensemble de tâches qui ont des prédécesseurs et des successeurs et T_F est l'ensemble des tâches finales qui n'ont pas de successeurs et qui, en général, appartiennent à l'ensemble des buts. La figure 1 représente un graphe de contraintes temporelles tel que $T_I = \{t_1, t_2, t_3, t_4, t_5\}$, $T_F = \{t_{17}, t_{18}\}$ et $T_M = \{t_6, \dots, t_{16}\}$. Dans la section suivante, nous écrivons le problème temporel à résoudre.

3 DESCRIPTION DU PROBLÈME

Classiquement un planificateur dispose en entrée d'un problème et d'un domaine de planification. Un problème de planification consiste en une description de l'état initial et de l'état but. Un domaine de planification est décrit par un ensemble d'actions qui vont permettre des transitions entre états logiques.

Le problème à résoudre est un problème de planification temporelle, dont les contraintes de précédence (ET/OU, délai) et les contraintes temporelles ($\Delta, [I_t^-, I_t^+]$).

Le but est de trouver un plan des tâches exécutables qui amène d'un ensemble de tâches initiales à un ensemble de

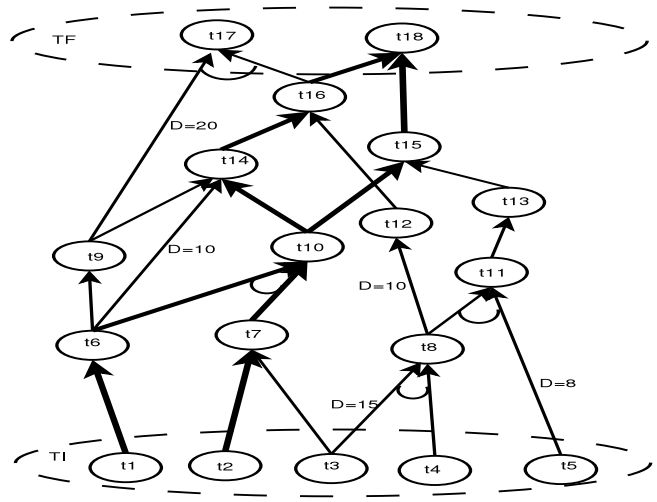


FIG. 1 – Un graphe de contraintes temporelles
Les plans faisables sont marqués en gras. Un arc de cercle entre deux arêtes représente une relation conjonctive entre deux tâches.

but et qui satisfait toutes les contraintes. Un tel plan est nommé *plan admissible*. Nous nous intéressons à trouver un plan qui a une grande probabilité d'être exécutée et un temps et un coût réduits. On définit le problème comme suit :

Définition 3 Un problème de génération de plan est un couple (G, T') où $G = (T, E)$ est un graphe de contraintes temporelles et où $T' \subset T_F \subset T$ est un sous-ensemble de l'ensemble T_F des tâches finales de G .

Une solution du problème d'écrit précédemment est définie par un ensemble de tâches à exécuter de telle façon que tous les buts demandés soient atteints. Un plan est un chemin dans le graphe de l'ensemble des tâches initiales T_I à l'ensemble des tâches appartenant à T_F .

Nous utilisons la méthode de recherche dans l'espace des plans partiels qui part d'un plan incomplet contenant l'état final. Ce plan partiel est étendu jusqu'à obtenir un plan complet, appelé *faisable*, sans défauts, c'est à dire qui respecte toutes les contraintes de précédence. Ce problème de planification est démontré NP-complet.

Le plan doit être aussi *admissible* en respectant les contraintes temporelles. On désigne par \mathcal{P} l'ensemble des plans faisables dans le graphe G et $T(\mathcal{P})$ le sous-ensemble de T de toutes les tâches dans un plan faisable \mathcal{P} . Par exemple, dans la figure 1, les plans faisables (marqués en gras) amenant de $t \in T_I$ à $t_{18} \in T_G$ sont $[[t_2, t_7], [t_1, t_6], t_{10}, t_{14}, t_{16}, t_{18}]$ et $[[t_2, t_7], [t_1, t_6], t_{10}, t_{15}, t_{18}]$.

4 PROPAGATION DES INTERVALLES D'EXÉCUTION

Étant donné un graphe de contraintes temporelles $G = (T, E)$, notre but est de calculer pour chaque tâche des plans faisables l'ensemble des intervalles de temps d'exécution possibles. Nous avons développé un algorithme basé sur une propagation temporelle dans le

graphe des tâches. Cette propagation organise le graphe en plusieurs niveaux : l_0 contient les tâches initiales du graphe (T_I), l_1 contient tous les nœuds (tâches) qui succèdent directement aux tâches du niveau l_0 . Les feuilles sont les nœuds qui n'ont pas de successeurs (T_F). Pour chaque tâche dans un niveau donné l , nous calculons tous les intervalles temporels possibles à partir de ses prédécesseurs en respectant toutes les contraintes temporelles et les contraintes de précedence.

Nous désignons par st_t la date de début d'exécution, par et_t la date de fin d'exécution, par ST_t l'ensemble de dates de début possibles, par ET_t l'ensemble de dates de fin possibles et par I_t l'ensemble des intervalles temporels possibles pour une tâche t . Tous les intervalles d'exécution possibles de chaque tâche peuvent être calculés hors-ligne (avant l'exécution des tâches) en utilisant un algorithme similaire à celui décrit dans [BW00]. Les dates de début et les dates de fin d'une tâche $t \in T(\mathcal{P})$ en supposant que $\Delta_t = \{d_t^1, d_t^2, \dots, d_t^m\}$ et $i = 1..m$ sont calculées comme suit :

- niveau l_0 : $t \in T_I$ (t est une tâche initiale). Supposons que l'agent se réveille à la date *start time*.
 - $st_t = \max\{I_t^-, \text{start time}\}$, $et_t^i = st_t + d_t^i$
 - $I_t^i = [st_t, et_t^i]$
 - $ST_t = \{st_t\}$, $ET_t = \{et_t^i\}$ et $I_t = \{I_t^i\}$
- niveau l_i : les dates de début possibles de chaque nœud (tâche) appartenant à un niveau l_i , sont calculées à partir des dates de fin des prédécesseurs du nœud. Les dates de début et de fin des tâches autres que la racine sont donc calculées de la manière suivante :
 - si $t \in T_M \cup T_F$ (t est une tâche intermédiaire ou finale) et t est un nœud OR tel que $ET_{t_i} = \{et_{t_i}^1, \dots, et_{t_i}^n\}$ (déjà calculée) est l'ensemble des dates de fin possibles de la tâche précédente, t et supposons que $\Delta_t = \{d_t^1, d_t^2, \dots, d_t^m\}$ alors
 - $st_t^j = \max\{I_t^-, et_{t_i}^j + \delta_{t_i,t}\}$, $et_t^{jk} = st_t^j + d_t^k$
 - $I_t^{jk} = [st_t^j, et_t^{jk}]$, $I_t = \{I_t^{jk}\}$
 - $ST_t = \{st_t^j\}$ et $ET_t = \{et_t^{jk}\}$
 - tel que $j = 1..n$ et $k = 1..m$
 - si $t \in T_M \cup T_F$ (t est une tâche intermédiaire ou finale) tel que $\Delta_t = \{d_t^1, d_t^2, \dots, d_t^m\}$ et t a n tâches $\{t_1, \dots, t_n\}$ comme prédécesseurs immédiats (le cas d'un nœud ET) tel que pour chaque tâche t_i ($i = 1..n$), $ET_{t_i} = \{et_{t_i}^1, et_{t_i}^2, \dots, et_{t_i}^{j_i}\}$ alors :
 - $st_t^k = \max\{I_t^-, \max(et_{t_i}^k + \delta_{t_i,t})\}$, $et_t^{kr} = st_t^k + d_t^r$
 - $I_t^{kr} = [st_t^k, et_t^{kr}]$, $I_t = \{I_t^{kr}\}$
 - $ST_t = \{st_t^k\}$ et $ET_t = \{et_t^{kr}\}$
 - tel que $k = 1..j_i$, $r = 1..m$ et $i = 1..n$.

Un plan faisable $\mathcal{P} \in \mathcal{P}_F$, pour qu'il devienne *admissible*, doit vérifier toutes les contraintes temporelles. Plus formellement : Pour toute tâche $t_i \in T(\mathcal{P})$ et pour chaque $d_{t_i}^j \in \Delta_{t_i}$, ces conditions doivent être vraies : $st_{t_i}^j \geq I_{t_i}^-$, et $st_{t_i}^j + d_{t_i}^j \leq I_{t_i}^+$.

Nous désignons par \mathcal{P}_A l'ensemble de tous les plans admissibles \mathcal{P} de \mathcal{P}_F . $\mathcal{P} = \{t_i/I_{t_i}^j, t_{i+1}/I_{t_{i+1}}^k, \dots, t_n/I_{t_n}^l\}$ est un plan formé par un ensemble de tâches t_1, \dots, t_n exécutées dans leurs intervalles temporels $I_{t_1}^j, \dots, I_{t_n}^l$

respectivement. Nous avons $\mathcal{P}_A = \bigcup_{i=1}^m \mathcal{P}_i$, tel que $\mathcal{P}_i \in \mathcal{P}_F$ est un plan admissible.

5 ALGORITHME DE PROPAGATION DES PROBABILITÉS

Nous écrivons dans cette section comment attribuer une probabilité à chacun des intervalles calculés précédemment. En effet, la probabilité d'un intervalle d'exécution I_t^i dépend de la date à laquelle la tâche commence (date de fin de la tâche précédente) et de la probabilité de sa durée d'exécution d_t^i (pour simplifier, nous considérons dans le reste de cet article que les délais entre les tâches sont égaux à zéro). Soit I_t^i un intervalle, st_t^i sa date de début et et_t^i sa date de fin ($I_t^i = [st_t^i, et_t^i]$). La probabilité qu'une tâche t s'exécute dans l'intervalle I_t^i est égale à la probabilité que son exécution commence à st_t^i et se termine à et_t^i , autrement dit que l'exécution de la tâche commence à st_t^i et dure $et_t^i - st_t^i$ unités de temps. Afin de commencer l'exécution d'une tâche t , tous les prédécesseurs de cette dernière doivent avoir été exécutés. La probabilité que l'exécution de t commence à st_t^i est définie par :

- $pr_{start}(st_t^i | et_{t'}^j)$ si t est un nœud OU et il a un seul prédécesseur qui termine son exécution à $et_{t'}^j$:
 - si $st_t^i < et_{t'}^j$, $pr_{start}(st_t^i | et_{t'}^j) = 0$
 - si $st_t^i \geq et_{t'}^j$, $pr_{start}(st_t^i | et_{t'}^j) = 1$
- $pr_{start}(st_t^r | et_{t_i}^k)$ si t est un nœud ET et il a plusieurs prédécesseurs $\{t_1, t_2, \dots, t_n\}$. Soit $t_i \in \{t_1, t_2, \dots, t_n\}$ tel que $ET_{t_i} = \{et_{t_i}^1, et_{t_i}^2, \dots, et_{t_i}^{j_i}\}$ où j_i représente le nombre de dates de fin possibles de t_i et $ST_t = \{st_t^1, st_t^2, \dots, st_t^m\}$ est l'ensemble de dates de début possibles de t alors :
 - si $st_t^r < \max\{et_{t_i}^k\}$ alors $pr_{start}(st_t^r | et_{t_i}^k) = 0$
 - si $st_t^r \geq \max\{et_{t_i}^k\}$ alors $pr_{start}(st_t^r | et_{t_i}^k) = 1$

Un cas spécial doit être fait de la première tâche. En effet, les tâches initiales ($\in T_I$) n'ont pas de prédécesseurs. La probabilité qu'une tâche initiale t commence son exécution à st_t est donnée par $pr_{start}(st_t) = 1$. $pr_t(d_t^i | st_t^i)$ désigne la probabilité que l'exécution de t dure d_t^i unités de temps si elle a commencé à st_t . Les probabilités sur les durées dépendent donc des dates de début d'exécution.

La probabilité qu'une tâche s'exécute dans un intervalle I_t^i sachant que son prédécesseur t' a terminé son exécution à $et_{t'}^j$ est notée $Pr_w(I_t^i | et_{t'}^j)$. Elle est définie de la manière suivante :

$$Pr_w(I_t^i | et_{t'}^j) = pr_{start}(st_t^i | et_{t'}^j) * pr_t(d_t^i | st_t^i) \quad (1)$$

où $et_{t'}^j$ est la date de fin de la dernière tâche exécutée (prédécesseur). En effet, une tâche t ne pourra commencer à être exécutée que lorsque son prédécesseur t' a terminé. La date de début de t dépend donc de la date de fin de t' . Dans le cas où t , tel que $\Delta_t = \{d_t^1, d_t^2, \dots, d_t^m\}$ est un nœud ET et a plusieurs prédécesseurs, t_1, t_2, \dots, t_n tel que $t_i \in \{t_1, t_2, \dots, t_n\}$ termine son exécution à $et_{t_i}^{k_{t_i}}$ ($k_{t_i} = 1..j_i$ tel que j_i représente le nombre de dates de fin possibles de t_i), on a :

$$Pr_w(I_t^r | et_{t_i}^{k_{t_i}}) = pr_{start}(st_t^r | et_{t_i}^{k_{t_i}}) * pr_t(d_t^r | st_t^r) \quad (2)$$

où $r = 1..m$.

Un cas spécial doit être envisagé pour les tâches initiales :

$$Pr_w(I_t^r) = pr_{start}(s_t) * pr_t(d_t^r | s_t) \quad (3)$$

Supposons que la probabilité que l'exécution de la tâche t_i dure 2 unités de temps lorsqu'elle commence à 5, soit égale à 0.7 ($Pr_{t_i}(d_{t_i}^j = 2 | st_{t_i}^j = 5) = 0.7$). Supposons également que la tâche précédente se termine à 3. Considérons la tâche t , la probabilité de l'intervalle $[5, 7]$ est alors 0.7 : $Pr_w(I_{t_i}^j = [5, 7] | et_{t_i}^j = 3) = 1 \times 0.7 = 0.7$

Si par exemple, la tâche $t_{i'}$ se termine à 7, on a : $Pr_w(I_{t_i}^j = [5, 7] | et_{t_{i'}}^j = 7) = 0 \times 0.7 = 0$

6 SÉLECTION ET EXÉCUTION DU MEILLEUR PLAN

Étant donné que l'ordre de recherche des plans est arbitraire, il est possible d'obtenir plusieurs plans admissibles. Dans ce cas, plusieurs critères de préférences (probabilité, temps et coût) sont utilisés pour analyser et comparer les plans et ensuite en choisir un seul pour être exécuté. Dans cette section, nous proposons la méthode utilisée pour choisir "le meilleur plan". Nous nous sommes intéressés à trouver un plan qui a une forte probabilité d'être exécuté avec un temps et un coût réduits.

Pour chaque tâche dans un plan admissible \mathcal{P} , nous calculons les utilités espérées de coût et de temps. Soit la tâche t telle que $\Delta_t = \{d_t^1, d_t^2, \dots, d_t^m\}$ est l'ensemble des durées d'exécution possibles de t , $C = \{c_t^1, c_t^2, \dots, c_t^m\}$ est l'ensemble des coûts d'exécution de t tel que ξ est le coût de l'exécution de la tâche t pendant la durée k ($k = 1..m$).

1. si t a un seul prédécesseur t' dont l'ensemble des dates de fin possibles est $ET_{t'} = \{et_{t'}^1, et_{t'}^2, \dots, et_{t'}^r\}$, et si la probabilité que t s'exécute pendant une durée k sachant que son prédécesseur termine son exécution à $et_{t'}^j$ est notée $Pr_w(I_t^k | et_{t'}^j)$, alors :

- l'utilité espérée de coût de t est calculée de la façon suivante :

$$\mu(cout(t|t')) = \sum_{j=1}^r \sum_{k=1}^m Pr_w(I_t^k | et_{t'}^j) * c_t^k \quad (4)$$

Pour simplifier, on note l'utilité espérée de coût de t par $\mu(cout(t))$.

- l'utilité espérée de temps de t est calculée de la façon suivante :

$$\mu(temps(t|t')) = \sum_{j=1}^r \sum_{k=1}^m Pr_w(I_t^k | et_{t'}^j) * \delta_t^k \quad (5)$$

Pour simplifier, on note l'utilité espérée de temps de t par $\mu(temps(t))$.

2. si t a un ensemble de prédécesseurs directs t_1, t_2, \dots, t_n alors, soit $ET_{t_i} =$

$\{et_{t_i}^1, et_{t_i}^2, \dots, et_{t_i}^{r_{t_i}}\}$ l'ensemble des dates de fin possibles de $t_i \in \{t_1, t_2, \dots, t_n\}$ alors :

- l'utilité espérée de coût de t est calculée de la façon suivante :

$$\mu(cout(t|t_1, \dots, t_n)) = \sum_{i=1}^n \mu(cout(t|t_i)) \quad (6)$$

Pour simplifier, on note l'utilité espérée de coût de t par $\mu(cout(t))$.

- l'utilité espérée de coût de t est calculée de la façon suivante :

$$\mu(temps(t|t_1, \dots, t_n)) = \sum_{i=1}^n \mu(temps(t|t_i)) \quad (7)$$

Pour simplifier, on note l'utilité espérée de temps de t par $\mu(temps(t))$.

Après avoir calculé l'utilité espérée du coût et de temps de chaque tâche, nous calculons l'utilité espérée totale de coût et l'utilité espérée totale de temps de chaque plan comme suit :

- l'utilité espérée totale de coût d'un plan admissible \mathcal{P} est calculée en additionnant toutes les utilités espérées de coût de toutes les tâches de ce plan. Plus formellement :

$$\mu(cout(\mathcal{P})) = \sum_{i=1}^n \mu(cout(t_i)) \quad (8)$$

tel que n est le nombre des tâches dans le plan \mathcal{P} .

- l'utilité espérée totale de temps d'un plan admissible \mathcal{P} ($\mu(temps(\mathcal{P}))$) est initialisée à $\max(\mu(temps(t_i)))$ où $t_i \in T_F$ est calculée comme suit :

- si t est un nœud OU et il a un seul prédécesseur t' avec une utilité espérée $\mu(temps(t'))$, alors on ajoute $\mu(temps(t'))$ à l'utilité espérée totale du plan. Plus formellement :

$$\mu(temps(\mathcal{P})) = \mu(temps(\mathcal{P})) + \mu(temps(t')) \quad (9)$$

- si t est un nœud ET et il a un ensemble de prédécesseurs $\{t_1, t_2, \dots, t_n\}$ alors, on ajoute le maximum de l'utilité espérée des tâches t_1, t_2, \dots, t_n à l'utilité espérée totale du plan. Plus formellement :

$$\mu(temps(\mathcal{P})) = \mu(temps(\mathcal{P})) + \max_{i=1}^n \mu(temps(t_i)) \quad (10)$$

Pour calculer l'utilité espérée totale de chaque plan $\mathcal{P} \in \mathcal{P}_A$, nous assignons une valeur réelle à chaque plan selon les préférences de l'utilisateur. Nous traduisons ces préférences en une fonction $\mu(\mathcal{P})$: μ appelée la fonction utilité. En introduisant des coefficients α et β pour le temps et le coût, nous valorisons l'importance des utilités données par l'utilisateur au temps ou au coût. Nous obtenons un problème d'analyse à multi-critères qui nous permet d'ajuster l'importance relative des diverses utilités selon les préférences de l'utilisateur. Cette utilité est décrite de la façon suivante :

$$\mu(\mathcal{P}) = \alpha * \mu(cout(\mathcal{P})) + \beta * \mu(temps(\mathcal{P})) \quad (11)$$

tels que $\alpha + \beta = 1$ et $\mathcal{P} \in \mathcal{P}_A$.

Parmi tous les plans admissibles, nous choisissons pour l'exécution celui qui a l'utilité minimale :

$$\mathcal{P}^* = \operatorname{argmin}_{\mathcal{P} \in \mathcal{P}_A} \mu(\mathcal{P}) \quad (12)$$

À noter que chaque plan $\mathcal{P} \in \mathcal{P}_A$ est formé de plusieurs ordonnancements en prenant les différents intervalles d'exécution possibles I_i^j pour chaque tâche t_i de $T(\mathcal{P})$. Plus formellement : $\mathcal{P} = \bigcup \mathcal{P}^f$ tel que $\mathcal{P}^f = \{t_i/I_i^j, t_{i+1}/I_{i+1}^k, \dots, t_n/I_n^l\}$ avec $j = 1 \dots m$ où m est le nombre d'intervalles d'exécution possibles de la tâche t_i , $k = 1 \dots s$ où s est le nombre d'intervalles d'exécution possibles de la tâche t_{i+1} et $l = 1 \dots o$ où o est le nombre d'intervalles d'exécution possibles de la tâche t_n .

Le nombre total des ordonnancements d'un plan \mathcal{P} est égal au produit des cardinalités des ensembles d'intervalles d'exécution possibles de toutes les tâches de $T(\mathcal{P})$. Plus formellement ce nombre est égal à $m * s * \dots * o$.

Pour chaque ordonnancement $\mathcal{P}_i^* = \{t_i/I_i^j, t_{i+1}/I_{i+1}^k, \dots, t_n/I_n^l\}$ d'un plan admissible \mathcal{P}^* choisi pour être exécuté, nous calculons sa fonction d'utilité, tel que $\alpha + \beta = 1$ et $t \in T(\mathcal{P}^*)$, défini par :

$$\mu(\mathcal{P}_i^*) = \alpha * \mu(\text{cout}_{t_i}^*) + \beta * \mu(\text{temps}_{t_i}^*) \quad (13)$$

tel que :

$$\mu(\text{cout}_{t_i}^*) =$$

$$Pr_w(I_{t_i}^j) * \text{cout}(I_{t_i}^j) + Pr_w(I_{t_{i+1}}^k | et_{t_i}^j) * \text{cout}(I_{t_{i+1}}^k | et_{t_i}^j) \\ + \dots + Pr_w(I_{t_n}^l | et_{t_{n-1}}^s) * \text{cout}(I_{t_n}^l | et_{t_{n-1}}^s)$$

et

$$\mu(\text{temps}_{t_i}^*) =$$

$$Pr_w(I_{t_i}^j) * \text{temps}(I_{t_i}^j) + Pr_w(I_{t_{i+1}}^k | et_{t_i}^j) * \text{temps}(I_{t_{i+1}}^k | et_{t_i}^j) \\ + \dots + Pr_w(I_{t_n}^l | et_{t_{n-1}}^s) * \text{temps}(I_{t_n}^l | et_{t_{n-1}}^s)$$

De tous les ordonnancements formés par \mathcal{P} nous choisissons celui qui a l'utilité minimal :

$$\mathcal{P}_{exec}^* = \operatorname{argmin}_{\mathcal{P}_i^* \in \mathcal{P}^*} \mu(\mathcal{P}_i^*) \quad (14)$$

Cet ordonnancement détermine les intervalles d'exécution espérés, pour chaque tâche, d'être exécutés par l'agent.

7 ANALYSE ET DISCUSSION

Dans cette section, nous analysons le nombre d'intervalles temporels pour chaque tâche t_i dans le graphe $G = (T, E)$. Dans le pire des cas, le nombre d'intervalles temporels pour une tâche initiale (dans T_I) est égale à la cardinalité de son ensemble de durées. Le nombre d'intervalles temporels pour une tâche intermédiaire ou finale (dans T_M ou T_F) de la forme d'un nœud OU est égale à la cardinalité de son ensemble de durées multipliée par la cardinalité de l'ensemble de dates de fins de la tâche précédente. Le nombre d'intervalles temporels pour

une tâche intermédiaire ou finale (dans T_M ou T_F) de la forme d'un nœud ET est égale à la cardinalité de son ensemble de durées multipliée par toutes les cardinalités de tous les ensembles de date de fin des tâches précédentes. Plus formellement, dans le pire de cas, nous avons :

- si $t \in T_I$ alors $|I_t| = |\Delta_t|$,
- si $t \in T_M \cup T_F$ et t est un nœud OU alors $|I_t| = |\Delta_t| * |ET_{t'}|$ tel que t' est une tâche directement précédente de la tâche t ,
- si $t \in T_M \cup T_F$ est un nœud ET tel que t a un ensemble de prédécesseurs directs $\{t_2, \dots, t_n\}$ alors $|I_t| = |\Delta_t| * \prod_{i=1}^n |E_{t_i}|$

Le nombre d'intervalles temporels croît exponentiellement avec la taille du plan dans le graphe. Mais il est possible qu'on obtienne plusieurs intervalles temporels identiques (qui ont les mêmes dates de début et de fin), mais avec des probabilités d'exécution différentes. Dans ce cas, nous pouvons considérer que ces intervalles temporels forment un seul intervalle avec la date de début et de fin et avec une probabilité égale à la somme de toutes les probabilités de tous les intervalles identiques (démonstration dans la section 7.1). Par exemple, si pour la tâche t_i nous obtenons les cinq intervalles temporels $[6, 7]$, $[6, 8]$, $[7, 9]$, $[6, 7]$, et $[7, 9]$ avec les probabilités 0.10, 0.20, 0.09, 0.21, 0.40 respectivement, on peut dire qu'on a seulement trois intervalles temporels $[6, 7]$, $[6, 8]$ et $[7, 9]$ avec les probabilités 0.31, 0.20 et 0.49 respectivement. A noter que $\sum pr_{t_i}^j \leq 1$.

7.1 Réduction de nombre d'intervalles

Dans cette section, nous analysons les différents cas de calcul des intervalles temporels des tâches afin de le réduire.

- niveau l_0 : $t \in T_I$ (t est une tâche initiale) tel que $\Delta_t = \{d_t^1, d_t^2, \dots, d_t^m\}$:
Le nombre d'intervalles de t est égal au nombre de durées qui satisfont la condition : $\max(\text{start_time}, I_t^-) + d_t^i \leq I_t^+$, où $[I_t^-, I_t^+]$ est la fenêtre temporelle de la tâche t et $d_t^i \in \Delta_t$.
- niveau l_i : $t \in T_M \cup T_F$ (t est une tâche intermédiaire ou finale) :

Commençons par le cas où t est un nœud OU et a une seule tâche directement précédente t' ($t' \rightarrow t$). Supposons que l'ensemble des dates de fins possibles de t' est $ET_{t'} = \{et_{t'}^1, \dots, et_{t'}^p\}$, la fenêtre temporelle de la tâche t est $[I_t^-, I_t^+]$ et $\Delta_t = \{d_t^1, d_t^2, \dots, d_t^m\}$. On considère la tâche t . Nous distinguons deux cas :

1. si le maximum des dates de fin $et_{t'}^j \in ET_{t'}$ est inférieur ou égal à I_t^- , alors $st_t = I_t^-$ et le nombre d'intervalles temporels possibles de la tâche t est égal au nombre de ses durées qui satisfont la condition : $I_t^- + d_t^i \leq I_t^+$.
2. si le minimum des dates de fin $et_{t'}^j \in ET_{t'}$ est supérieur ou égal à I_t^- , alors le nombre d'intervalles temporels possibles de la tâche t est égal à la cardinalité de $ET_{t'}$ multipliée par le nombre de durées de la tâche t qui satisfont la condition : $\max(et_{t'}^j, I_t^-) + d_t^i \leq I_t^+$.
3. sinon, le nombre d'intervalles temporels pos-

sibles de la tâche t est égale à un plus le nombre d'intervalles temporels possibles de la tâche t' qui ont différentes valeurs de $e_{t'}^i$ et pour lesquels $e_{t'}^j > I_t^-$ multiplié par le nombre de durées de la tâche t qui satisfont la condition : $\max(e_{t'}^j, I_t^-) + d_t^i \leq I_t^+$.

Nous considérons maintenant le cas où t est un nœud ET et a plusieurs tâches directement précédentes $\{t_1, t_2, \dots, t_n\}$. Soit $t_i \in \{t_1, t_2, \dots, t_n\}$ tel que $E_{t_i} = \{e_{t_i}^1, e_{t_i}^2, \dots, e_{t_i}^p\}$ est l'ensemble de dates de fin possibles de t_i . Pour calculer l'ensemble de dates de fin possibles de l'ensemble de prédécesseurs de t , noté $ET(t_1, \dots, t_n)$, on fait une combinaison entre tous les ensembles E_{t_i} pour choisir la date de fin maximale de chaque combinaison. Plus formellement :

$$E(t_1, \dots, t_n) = \bigcup \{ \max_{k_{t_i}=1}^p (e_{t_1}^{k_{t_1}}, e_{t_2}^{k_{t_2}}, \dots, e_{t_n}^{k_{t_n}}) \}$$

Par exemple, si une tâche t a trois prédécesseurs directs t_1, t_2 et t_3 qui terminent leurs exécutions à $E_{t_1} = \{e_{t_1}^1, e_{t_1}^2, e_{t_1}^3\}$, $E_{t_2} = \{e_{t_2}^1, e_{t_2}^2, e_{t_2}^3\}$ et $E_{t_3} = \{e_{t_3}^1, e_{t_3}^2\}$ respectivement. L'ensemble $E(t_1, t_2, t_3)$ est composé de l'ensemble de dates de fin possibles suivant : $\{\max(e_{t_1}^1, e_{t_2}^1, e_{t_3}^1), \max(e_{t_1}^1, e_{t_2}^1, e_{t_3}^2), \dots\}$.

Dans le pire de cas, on a :

$$|E(t_1, \dots, t_n)| = |E_{t_1}| * |E_{t_2}| * \dots * |E_{t_n}|$$

Après avoir calculé l'ensemble de dates de fin possibles ($E(t_1, \dots, t_n)$) de l'ensemble de prédécesseurs de t , on peut calculer le nombre d'intervalles temporels possibles de t comme décrit ci-dessus.

Dans la section suivante, nous donnons un exemple explicatif pour bien comprendre ces analyses.

7.2 Exemple sur la réduction de nombre d'intervalles

Dans cette section, nous expliquons les analyses précédentes grâce à un exemple. Supposons qu'on a trois tâches t_i, t_j et t_k avec ces données temporelles : $(t_i, [2, 6], \{2, 3\}, \{0.6, 0.4\}, \{10, 20\})$, $(t_j, [2, 8], \{1, 4, 6\}, \{0.1, 0.4, 0.5\}, \{10, 30, 35\})$ et $(t_k, [5, 9], \{1, 2, 4\}, \{0.1, 0.6, 0.3\}, \{15, 30, 35\})$.

Nous représentons ces tâches dans la figure 2 et nous supposons qu'il n'y a pas de délais entre les tâches.

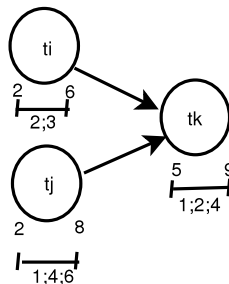


FIG. 2 – Un exemple d'un graphe de contraintes temporelles composé de 3 tâches.

Au début, supposons que t_k est un nœud OU. Considérons le plan (t, t_k) .

Grâce à la propagation temporelle décrite dans le paragraphe 4, nous obtenons : $I_{t_i}^1 = [2, 4]$ avec une probabilité égale à 0.60 et $I_{t_i}^2 = [2, 5]$ avec une probabilité égale à 0.40. Ces deux intervalles temporels satisfont la condition : $e_{t_i}^m = \{4, 5\} \leq I_{t_i}^+ = 6$. Pour calculer les intervalles temporels de la tâche t_k , nous utilisons les dates de fins possibles de la tâche t_i . Nous obtenons six intervalles possibles : $[5, 6], [5, 7], [5, 9], [5, 6], [5, 7]$ et $[5, 9]$ avec les probabilités 0.06, 0.36, 0.18, 0.04, 0.24 et 0.12 respectivement. À noter que trois intervalles temporels sont rejetés. Nous les réduisons en un seul intervalle temporel et nous additionnons leurs probabilités. Nous obtenons trois intervalles temporels $[5, 6], [5, 7]$ et $[5, 9]$ avec les probabilités 0.10, 0.60 et 0.30 respectivement.

Maintenant supposons que t_k est un nœud ET. Considérons le plan $([t, t_j], t_k)$. Grâce à la propagation temporelle décrite dans 4, nous obtenons : $I_{t_i}^1 = [2, 4]$ avec une probabilité égale à 0.60 et $I_{t_i}^2 = [2, 5]$ avec une probabilité égale à 0.40. Ces deux intervalles temporels satisfont la condition $e_{t_i}^m = \{4, 5\} \leq I_{t_i}^+ = 6$. $I_{t_j}^1 = [2, 3]$ avec une probabilité égale à 0.10, $I_{t_j}^2 = [2, 6]$ avec une probabilité égale à 0.40 et $I_{t_j}^3 = [2, 8]$ avec une probabilité égale à 0.50. Ces intervalles temporels satisfont la condition $e_{t_j}^n = \{3, 6, 8\} \leq I_{t_i}^+ = 8$.

Pour calculer les intervalles temporels de la tâche t_k nous utilisons les dates de fins possibles des tâches t_i et t_j .

Nous obtenons huit intervalles possibles :

$[5, 6], [5, 7], [5, 8], [6, 7], [6, 8], [6, 10], [8, 9], [8, 10], [8, 12], [5, 6], [5, 7], [5, 9], [6, 7], [6, 8], [6, 10], [8, 9], [8, 10]$ et $[8, 12]$. Remarquons que les intervalles $[6, 10], [8, 10]$ et $[8, 12]$ ne sont pas valides puisque ils ne satisfont pas la condition $e_{t_k}^p = \{10, 12\} \leq I_{t_k}^+ = 9$.

Nous éliminons les plans qui contiennent ces intervalles. Parmi tous les autres intervalles, six intervalles temporels sont répétés. Nous les réduisons en un seul et nous additionnons leurs probabilités. Nous obtenons seulement sept intervalles temporels $[5, 6], [5, 7], [5, 9], [6, 7], [6, 8]$ et $[8, 9]$.

8 EXPÉRIENCES

Le nombre d'intervalles temporels de chaque tâche t_i dépend du nombre des tâches prédécesseurs, de la largeur de sa fenêtre temporelle $[I_{t_i}^-, I_{t_i}^+]$ et de la cardinalité de son ensemble de durées $|\Delta_{t_i}|$. Quand une de ses contraintes croît, le nombre d'intervalles temporels croît. Le nombre de plans dans le graphe dépend du nombre d'intervalles temporels des tâches, du nombre de tâches et du nombre de contraintes de précédence. Ce nombre croît avec le nombre d'intervalles temporels des tâches et du nombre de tâches et diminue avec le nombre de contraintes de précédence. Nous avons réalisé des tests afin de vérifier l'efficacité de notre approche. Ces tests sont faits sur des graphes binaires. Nous avons calculé le nombre total de plans qui respectent les contraintes de précédence et le nombre total de plans admissibles obtenu après la vérification de la validité des contraintes temporelles. Nous avons fixé le nombre d'intervalles (5 inter-

valles) pour toutes les tâches puis nous avons augmenté le nombre de tâches. Nous avons commencé par un ensemble de 2 tâches (avec 5 intervalles chacune) puis nous avons ajouté un successeur à chaque tâche n'ayant qu'un seul successeur. Lorsque toutes les tâches ont 0 ou 2 successeurs, nous ajoutons un successeur à chaque feuille. La figure 3 montre le résultat obtenu, la ligne continue représente le nombre de plans faisable et la ligne pointillée représente le nombre de plans admissibles.

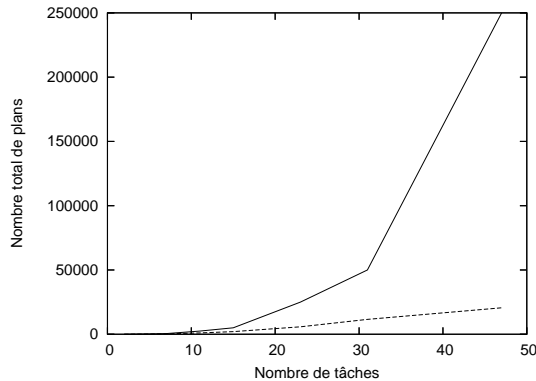


FIG. 3 – Nombre de plans obtenu vs. nombre de plans total (Nb d'intervalles =5)

9 CONCLUSION

Nous avons présenté une nouvelle approche pour la planification temporelle sous incertitude. Les tâches possèdent des contraintes temporelles, des coûts et des probabilités. Nous calculons les plans amenant d'une tâche initiale à une tâche finale en respectant les contraintes de précédence et grâce à une propagation temporelle à travers le graphe nous calculons les intervalles d'exécution respectant les contraintes temporelles. Puis nous choisissons un seul plan qui a une forte probabilité d'être exécuté avec un temps et un coût réduits.

La suite de notre travail va consister à étendre cette approche pour qu'elle soit capable de re-planifier en-ligne dans le cas où un plan exécutable échoue.

BIBLIOGRAPHIE

- [BDM⁺02] J. L. Bresina, R. Dearden, N. Meuleau, S. Ramakrishnan, D. E. Smith, and R. Washington. Planning Under Continuous Time and Resource Uncertainty : A Challenge for AI. In *AIPS Workshop on Planning for Temporal Domains*, pages 91–97, April 2002.
- [BW00] J. Bresina and R. Washington. Expected Utility Distributions for Flexible Contingent Execution. In *Proceedings of the AAAI-2000 Workshop : Representation Issues for Real-World Planning Systems*, 2000.
- [DHW94] D. Draper, S. Hanks, and D. Weld. Probabilistic Planning with Information Gathering and Contingent Execution. In *Proceedings*

of the Second International Conference on AI Planning Systems, AIPS-94, pages 31–63, June 1994.

- [DKS02] J. Dix, S. Kraus, and V. S. Subrahmanian. Agents dealing with Time and Uncertainty. In *The First International Joint Conference on Autonomous Agents & Multiagent Systems, AAMAS*, pages 912–919. ACM, July 2002.
- [DMP91] R. Dechter, I. Meiri, and J. Pearl. Temporal Constraint Network. *Artificial Intelligence*, 49(1-3) :61–95, 1991.
- [FN71] R. Fikes and N. J. Nilsson. STRIPS : A New Approach to the Application of Theorem Proving to Problem Solving. *Artif. Intell.*, 2(3/4) :189–208, 1971.
- [GL94] M. Ghallab and H. Laruelle. Representation and Control in IxTeT, a Temporal Planner. In *Proceedings of the Second International Conference on Artificial Intelligence Planning Systems, AIPS-94*, pages 61–67, June 1994.
- [KHW94] N. Kushmerick, S. Hanks, and D. Weld. An Algorithm for Probabilistic Least-Commitment Planning. In *Proceedings of the Twelfth National Conference on Artificial Intelligence, AAAI-94*, pages 1073–1078, August 1994.
- [LI04] S. Lemai and F. Ingrand. Interleaving Temporal Planning and Execution in Robotics Domains. In *AAAI*, pages 617–622, 2004.
- [MMV01] P. Morris, N. Muscettola, and T. Vidal. Dynamic Control of Plans with Temporal Uncertainty. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence, IJCAI-01*, pages 494–502. Morgan Kaufmann, 2001.
- [PG02] J. C. Pemberton and L. G. Greenwald. On the Need for Dynamic Scheduling of Imaging Satellites. In *Proceedings of the American Society for Photogrammetry and Remote Sensing, ASPRS-02*, 2002.
- [TVP03] I. Tsamardinos, T. Vidal, and M. E. Pollack. CTP : A New Constrained-based Formalism for Conditional Temporal Planning. *Constraints journal, special issue on Planning*, 8(4) :365–388, October 2003.