Chapitre V

Plan Space Planning Graph

Nous abordons dans ce chapitre le développement de méthodes d'analyse d'accessibilité dans l'espace des plans afin de proposer de nouvelles solutions pour guider la recherche de planificateurs non linéaires.

Nous avons montré dans le chapitre précédent que le succès des planificateurs heuristiques qui explorent l'espace d'états repose sur l'utilisation d'une méthode d'approximation de la portion accessible de l'espace de recherche à travers une structure appelée planning graph. Celle-ci permet notamment d'estimer les conséquences d'un choix au cours de la recherche. Si l'on confronte ces idées aux stratégies discutées dans la section III.2, on est frappé par la simplicité excessive, presque naïve, de ces dernières.

Certes il a déjà été proposé d'utiliser le planning graph comme base d'un estimateur heuristique pour un planificateur non linéaire (voir la présentation de RePOP, section III.4.2). Mais nous avons souligné les limites de cette approche et son inadéquation au cadre temporel traité par des planificateurs tels que RAX-PS ou IxTeT.

Plutôt que de transposer le planning graph et les méthodes propres aux recherches dans l'espace d'états, nous proposons ici une méthode originale pour effectuer une analyse d'accessibilité dans l'espace des plans. Notre approche est fondée sur l'exploitation d'une structure disjonctive permettant de circonscrire une portion pertinente de l'espace de recherche. Cette structure est qualifiée de *Plan Space Planning Graph* ou *PS-PG*. Elle peut être appréhendée de façon similaire à un plan partiel : il s'agit d'un ensemble d'actions et de contraintes qui spécifie un ensemble de plans candidats. Sa spécificité est son caractère disjonctif : cette structure représente en fait une approximation de tous les plans partiels alternatifs pouvant être développés par une recherche de type NONLIN.

Ce chapitre débute par une présentation intuitive des principes de notre approche. Puis nous procédons à une formalisation de ces principes et nous exposons les algorithmes permettant de construire pour un problème donné le PS-PG. Nous montrons ensuite comment le PS-PG peut se faire l'écho de la progression d'une recherche dans l'espace des plans en supportant la propagation des choix de résolution des défauts. Nous terminons en proposant une mesure des effets de telles propagations et nous ouvrons ainsi la discussion sur le chapitre suivant, qui traite des utilisations du Plan Space Planning Graph pour contrôler un processus de synthèse de plans non linéaire.

V.1 Explication intuitive

Bien que le but des travaux décrits ici soit de développer un processus d'analyse d'accessibilité spécifique à l'espace des plans, notre première approche a été de nous inspirer des méthodes propres à l'espace d'états, en se fondant sur le rapprochement entre ces deux espaces de recherche en termes de graphes.

Nous avons vu qu'un espace d'états pouvait être interprété comme un graphe dont les noeuds sont des états du monde et les arcs des transitions entre états par application d'actions. Un espace des plans pour un problème donné peut quant à lui être interprété comme un graphe dont les noeuds sont des plans partiels et les arcs des transformations de plans partiels visant à résoudre des défauts.

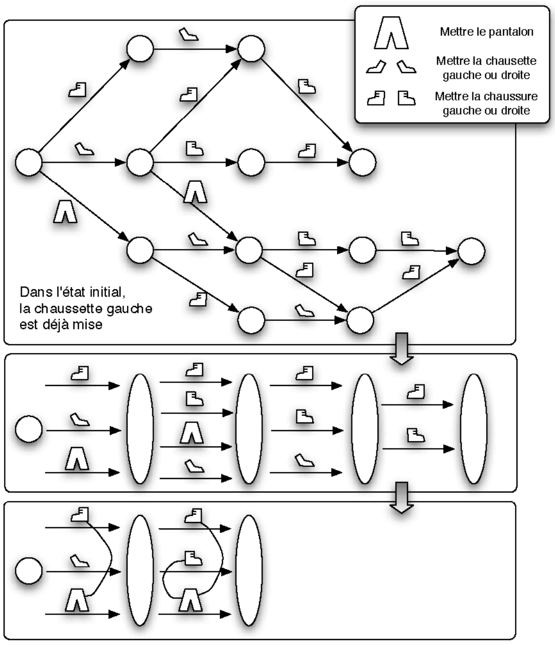

Fig. V.1 – Espace accessible pour un problème simple et Planning Graph associé

Dans le chapitre précédent, nous avons expliqué en détail les méthodes liées au Planning Graph. Pour un problème donné, le Planning Graph est une approximation de la partie de l'espace d'états accessible depuis l'état initial, c'est-à-dire une approximation du sous-graphe composé des noeuds accessibles depuis le noeud correspondant à l'état initial. Cette approximation est obtenue en développant une structure dans laquelle sont entrelacées deux types de couches. Les premières représentent des *méta-états*, en fait des fusions d'états. Les secondes définissent la relation entre deux méta-états successifs : une telle *méta-transition* réunit toutes les actions dont les préconditions sont contenues dans le méta-état qui la précède, les effets de ces actions étant présents dans le méta-état suivant. La première couche est un méta-état correspondant au seul état initial.

La figure V.1 illustre ainsi l'approximation d'une portion d'un espace d'état par un planning graph. Elle met en évidence que, au-delà de l'interprétation sémantique rattachée aux notions d'état et d'action, on procède à une approximation d'un graphe fondée sur un regroupement des noeuds situés à une même distance du noeud initial.

En retenant cette méthode d'approximation de graphes, on est alors tenté de procéder à une simple substitution des éléments propres à la planification non linéaire pour développer une forme de planning graph spécifique à l'espace des plans. Il s'agirait alors de développer à partir d'un plan partiel initial, une succession de couches contenant alternativement des fusions de plans partiels et toutes les façons possibles de résoudre les défauts de ces plans partiels.

De plus, dans la construction du planning graph, une relaxation fréquente consiste à négliger les interactions négatives entre actions. Pour un planificateur non linéaire, ceci revient à ignorer les menaces pouvant surgir dans un plan partiel. On ne calculerait donc pas les résolvantes de tous les défauts mais seulement celles des buts non expliqués.

Ce point de vue nous a guidé dans le développement de notre solution et nous verrons que l'on retrouve dans les algorithmes que nous proposons une phase d'expansion qui construit de façon itérative des plans partiels disjonctifs en considérant toutes les résolvantes des défauts. Néanmoins l'interprétation d'un espace des plans comme un graphe, au même titre qu'un espace d'états, est une abstraction qui masque une spécificité importante de la planification non linéaire : alors que la solution d'une recherche dans un espace d'états est un *chemin* dans le graphe sous-jacent, la solution d'une recherche dans l'espace des plans est un *noeud* du graphe sous-jacent.

Cette nuance doit nous amener à reconsidérer la pertinence de la transposition d'une structure multi-couches "façon planning graph" à un espace des plans partiels : alors que toutes les couches du Planning Graph sont porteuses d'informations, dans le cadre d'un espace des plans partiels la dernière couche devrait contenir toutes les informations.

Pour éclairer ces considérations sibyllines, il nous paraît judicieux de revenir sur l'interprétation initiale de la notion de plan partiel : un plan partiel est un ensemble de contraintes qui spécifie des plans candidats. Parmi ces plans candidats, certains sont des plans solutions. Une recherche non linéaire enrichit progressivement un plan partiel en ajoutant des contraintes, de façon à réduire l'ensemble des plans candidats exclusivement à des plans solutions (voir fig. V.2, où les étoiles correspondent à des solutions).

Nous avons vu que ces contraintes sont ajoutées pour éliminer des défauts dans les plans partiels et qu'un défaut peut souvent être résolu de diverses façons. Ainsi des points de choix apparaissent dans la recherche et il est nécessaire de se focaliser sur une partie des plans candidats, quitte à écarter certaines solutions qui font pourtant partie des candidats du plan partiel initial.

Le but d'une analyse d'accessibilité est de pouvoir cerner aussi précisément que possible *toutes* les solutions pouvant être élaborées à partir du plan partiel initial, pour ensuite pouvoir mesurer à quel point l'ajout de contraintes rapproche d'une solution ou écarte des solutions. Il apparaît que pour calculer toutes les solutions, il faut considérer toutes les résolvantes possible de chacun des défauts et non s'engager par des choix. Réunir ces résolvantes dans une même structure, c'est considérer une disjonction de contraintes.

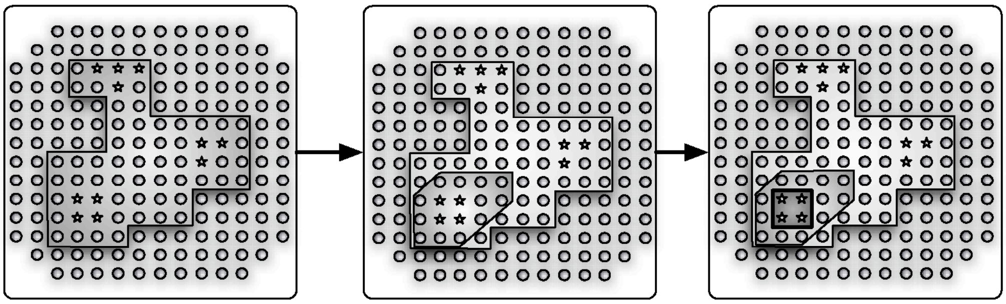

FIG. V.2 – Ajouts successifs de contraintes pour réduire l'ensemble des plans candidats à des solutions

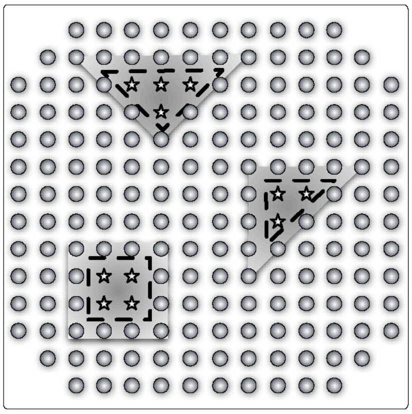

FIG. V.3 – Comment définir l'enveloppe disjonctive des plans solutions ?

Dès lors, étudier une forme d'accessibilité dans l'espace des plans, c'est expliciter une disjonction de contraintes qui cernent aussi précisément que possible les plans candidats. On pourra ensuite mesurer les conséquences d'un choix au cours de la recherche en réduisant la disjonction calculée par ajout des contraintes correspondant à la résolvante désirée. On pourra ainsi détecter par propagation les cas où l'ajout de certaines contraintes conduit à écarter des solutions de l'ensemble des plans candidats.

Bien sûr, la complexité du processus évoqué ci-dessus apparaît comme un frein rédhibitoire à une telle réalisation. C'est pourquoi il est nécessaire de procéder à une relaxation du problème, qui mènera certes à une approximation. Comme suggéré plus haut, une relaxation immédiate est de négliger les interactions négatives entre actions, à l'instar de ce qui est fait pour le planning graph. Ceci revient à ignorer les menaces dans un plan partiel et à se focaliser sur les alternatives pour résoudre les buts non expliqués.

Nous définissons donc dans la suite de ce chapitre une structure disjonctive, nommée *Plan Space Planning Graph* (ou *PS-PG*), fruit des réflexions résumées ci-dessus. Pour un problème donné, le PS-PG capture une approximation de l'ensemble des plans solutions accessibles depuis le plan partiel initial. Nous proposons des algorithmes de construction de cette structure, puis des algorithmes de propagation permettant de répercuter les choix de la recherche non linéaire afin d'en mesurer les conséquences sur les solutions accessibles.

V.2 Présentation formelle

Nous présentons ici une méthode pour calculer pour un problème donné une approximation de la région de l'espace des plans qui peut être explorée par un planificateur non linéaire. À l'instar d'un plan partiel, on construit un ensemble d'actions et de contraintes qui définit une famille de plans. Mais plutôt que de choisir un lien causal pour chaque but non expliqué, on introduit de façon disjonctive les différents établisseurs possibles.

Nous définissons d'abord formellement les éléments constitutifs de la structure développée pour réaliser cette approximation, puis nous en présentons la méthode de construction. Cette dernière suggère une succession de couches de buts non expliqués et d'établisseurs. Mais pour bien comprendre la nature de la structure ainsi développée, il nous paraît plus juste de la rapprocher de la notion des graphes d'actions présentés dans [Smith 93].

Ces graphes d'actions ont initialement été proposés dans le but d'élaborer une stratégie de recherche permettant de repousser le traitement de certaines menaces. Ils permettent d'expliciter de façon sommaire les opérateurs pertinents pour les sous buts pouvant apparaître dans un problème donné, ainsi que les conflits pouvant exister entre ces opérateurs. On y trouve déjà l'idée d'expliciter les liens entre un sous-but et tous les opérateurs pouvant l'établir, mais la mise en oeuvre est limitée par plusieurs facteurs. Notamment les contraintes d'unification et de séparation sont négligées : une action apparaît au plus une fois dans le graphe, sous une forme abstraite qui est reliée à tous les sous-buts pouvant être unifiée avec un de ses effets. Nous proposons au contraire de développer une structure reflétant des relations de causalité en tenant compte de contraintes tant d'unification que d'ordonnancement.

Définition - Le *Plan Space Planning Graph (PS-PG)* est un graphe orienté avec deux types de noeuds : des noeuds d'actions et de propositions.

Il peut être représenté sous la forme d'un 7-tuple ($\mathcal{P}$, $\mathcal{A}$, Pre, $\mathcal{EL}$, $\mathcal{IL}$, $\mathcal{O}$,$\mathcal{B}$).

- $\mathcal{P}$ et $\mathcal{A}$ sont respectivement des ensembles de noeuds proposition et des noeuds action.
- Pre est un ensemble d'arcs entre $\mathcal{P}$ et $\mathcal{A}$: si $a \in \mathcal{A}$ alors $\mathcal{P}$ contient des noeuds correspondants aux préconditions de a et Pre contient des arcs liant ces noeuds à a.
- $\mathcal{EL}$ et $\mathcal{IL}$ définissent les autres arcs du graphe : ce sont des ensembles de liens causaux entre actions et propositions ($\mathcal{EL}$ représentent les liens externes et $\mathcal{IL}$ les liens internes).
- On note $\mathcal{L} = \mathcal{IL} \cup \mathcal{EL}$.
- Enfin $\mathcal{O}$ et $\mathcal{B}$ sont des ensembles de contraintes d'ordonnancement et de contraintes sur les variables ; ces contraintes sont induites par les liens causaux de $\mathcal{L}$.

Nous détaillons ci-après la procédure d'expansion du PS-PG pour un problème donné, ce qui nous permet de préciser la nature de ces diverses composantes. Mais déjà l'exemple de la figure V.4 nous permet de commenter la définition précédente. Le problème considéré est emprunté au domaine Zeno-Travel, il s'agit d'un problème de transport de personnes par avions entre divers aéroports. Les actions possibles sont l'embarquement et le débarquement de passagers, le vol d'avion, qui consomme du carburant, et l'approvisionnement en carburant d'avion. On considère ici le cas très simple où il faut déplacer un passager d'une ville à une autre.

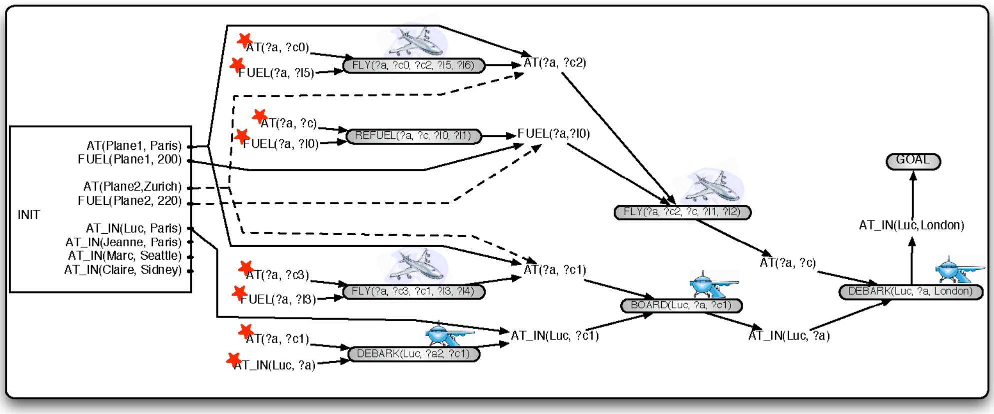

Fig. V.4 – Le Plan Space Planning Graph pour un problème du domaine Zeno

Les deux premiers éléments de $\mathcal{A}$ sont les noeuds correspondants aux actions A_0 et A_∞ (notées *Init* et *Goal* sur la figure) qui introduisent respectivement l'état initial et les buts du problème. Les buts sont des préconditions et les noeuds leur correspondant sont donc ajoutés à l'ensemble $\mathcal{P}$.

Les établisseurs des buts initiaux sont ensuite ajoutés au PS-PG. Dans cet exemple, le but initial n'admet qu'une méthode de résolution : l'ajout d'une action de débarquement. Celle-ci est donc ajoutée dans $\mathcal{A}$, accompagnée de ces préconditions (dans $\mathcal{P}$) et d'un lien causal (dans $\mathcal{EL}$). Ce processus est itéré pour chacun des sous-buts.

On fait la différence entre des liens causaux qui accompagnent l'ajout d'une nouvelle action, qualifiés de liens causaux *externes*, et ceux qui correspondent à la réutilisation d'actions figurant déjà dans $\mathcal{A}$, qualifiés de liens *internes*. Cette différenciation permet par la suite de mieux distinguer le coût des différentes méthodes de résolution d'un sous-buts, selon qu'elles nécessitent ou non l'ajout d'une action.

La structure du PS-PG est développée jusqu'à ce qu'on y distingue une solution valide au problème initial, ici explicité par les flèches en traits pleins. À ce stade certains sous-buts restent encore à résoudre. Ils sont indiqués par des étoiles sur la figure. Dans la suite de la discussion de la procédure d'extension, nous préciserons le critère d'arrêt de la procédure d'expansion qui reflète l'idée de "distinguer une solution", laquelle est liée à une propagation de coût qui permet de mesurer la prépondérance des sous-buts non expliqués.

Construction du PS-PG pour un problème donné - Le PS-PG est initialisé par insertion des deux actions A_0 et A_∞ qui définissent respectivement les conditions initiales et les buts du problème. Puis il est développé via la procédure *"Expansion"* ci-dessous. Cette procédure est par ailleurs illustrée par la figure V.5, qui présente le développement du PSPG correspondant à l'exemple de ZenoTravel examiné précédemment.

Expansion :
1. $OC \leftarrow Pre(A_\infty),\ NewOC \leftarrow \emptyset$
2. tant que $(OC \neq \emptyset)$
3. Pour chaque oc dans OC
4. $ExternEst \leftarrow$ actions_support(oc)
5. Pour chaque E dans $ExternEst$
6. inserer(E)
7. $NewOC \leftarrow NewOC \cup Pre(E)$
8. endfor
9. endfor
10. Pour chaque oc dans OC
11. $InternEst \leftarrow$ etablisseurs_internes(oc)
12. Pour chaque E dans $InternEst$
13. inserer(E)
14. finPour
15. finPour
16. Ajouter les contraintes des nouveaux liens causaux
17. $OC \leftarrow NewOC,\ NewOC \leftarrow \emptyset$
18. finTantque

Remarques sur la procédure "Expansion" -

La fonction *actions_support* calcule la liste des établisseurs externes possibles du but non expliqué oc, i.e. les nouvelles actions dont un effet peut être unifié avec oc. L'insertion d'un établisseur externe comprend les étapes suivantes : unification des variables concernées dans l'action et le but (ajouts de contraints dans $\mathcal{B}$), ajout de l'action dans $\mathcal{A}$, de ses préconditions dans $\mathcal{P}$ et des arcs correspondants dans Pre, puis ajout dans $\mathcal{EL}$ d'un lien causal avec le but non expliqué.

intern_establisher(oc) calcule la liste de tous les établisseurs internes de oc, i.e. toutes les actions déjà dans $\mathcal{A}$ dont un effet peut être unifié avec oc. Un élément de $\mathcal{A}$ peut établir oc si les contraintes résultant du lien causal correspondant sont cohérentes avec $\mathcal{O}$ et $\mathcal{B}$. L'insertion d'un établisseur interne dans le PS-PG est concretisée par l'ajout d'un lien causal dans $\mathcal{IL}$.

Les contraintes considérées en ligne 16 sont les contraintes d'ordonnancement et d'unification dues aux nouveaux établisseurs : une action doit être exécutée après les

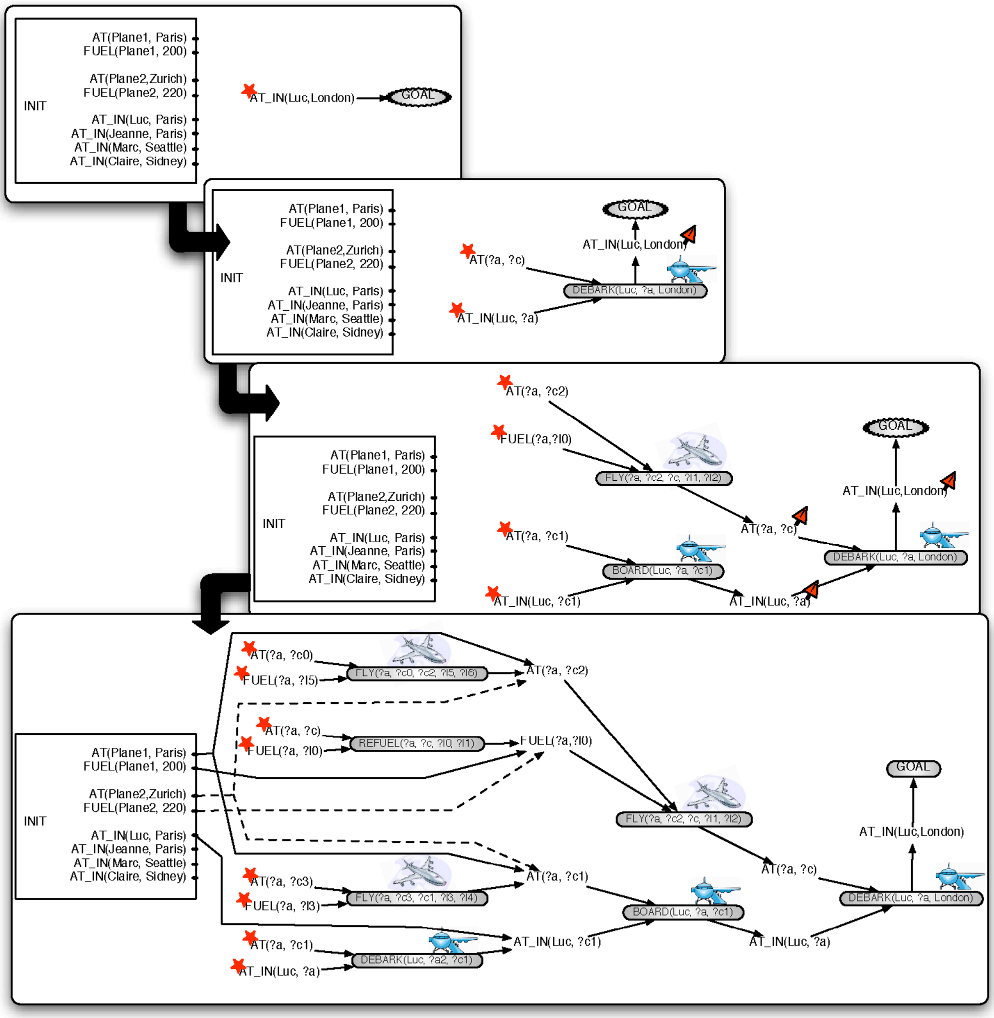

Fig. V.5 – Itérations successives de la procédure d'expansion du PS-PG établisseurs de ses préconditions et les domaines de ses paramètres dépendent de ces derniers. Puisque le PS-PG est développé sans engagement pour aucun établisseur particulier, ces contraintes sont disjonctives. Pour une action $a \in \mathcal{A}$ et une de ses préconditions $p \in \mathcal{P}$, les contraintes suivantes sont vérifiées :

– $(\bigvee_i (a > b_i), b_i \in Est(p)) \in \mathcal{O}$ where $Est(p) = \{b \in \mathcal{A}, b \rightarrow p \in \mathcal{L}\}$

– $\forall x \in Var(p) \quad Dom(x) \subset \cup_i(Dom(y_i), y_i \in Est(x))$ où $Est(x) = \{y, \exists b \in \mathcal{A}/y = \sigma(b \rightarrow p)(x)\}$

Arrêtons nous enfin sur la condition d'arrêt utilisée en ligne 2. Celle-ci n'est pas utilisable dans la plupart des cas. En effet un plan partiel peut presque toujours être étendu par ajout d'actions (boucles) qui vont amener de nouveaux sous buts non expliqués. Une stratégie réaliste est d'arrêter l'expansion du PS-PG en fonction d'un critère indiquant la possibilité que celui-ci contienne un plan solution. Nous proposerons un tel critère dans les sections suivantes.

La Figure V.5 présente un exemple de PS-PG développé pour un problème du domaine Zeno. Le domaine comprend 4 actions ((Fly, Refuel, Board, Debark). Les traits pleins (resp. en délié) sont des liens causaux externes (resp. internes). Les traits épais représentes les liens qui définissent une solution du problème.

Avant d'aborder l'utilisation du Plan Space Planning Graph comme la base d'un estimateur heuristique, nous énonçons une propriété importante.

Propriété - *Tous les plans partiels qu'un planificateur non linéaire pourrait explorer sont contenus dans le PS-PG.*

Plus précisément, si N itérations de la boucle principale de la procédure Expansion sont effectuées, alors tout plan partiel pouvant être élaboré pour résoudre N buts (et un nombre quelconque de menaces) est contenu dans le PS-PG.

Intuitivement, cette propriété est due aux deux étapes au sein de la procédure "Expansion" : d'abord sont déterminées et insérées les nouvelles actions, et c'est seulement après que sont considérés les liens causaux internes. Ainsi les actions qui viennent d'être ajoutées au plan partiel peuvent être réutilisées pour établir d'autres buts comme effets de bord. Nous assurons de la sorte que toutes les combinaisons d'établisseurs sont considérées.

V.2.1 Estimation de distance dans l'espace des plans partiels

Comme expliqué précédemment, le PS-PG peut être considéré comme un ensemble de contraintes qui définissent un ensemble de plans candidats. Il est possible d'évaluer un minorant de la longueur d'un plan solution dans le PS-PG. L'ajout de contraintes, telles que le choix d'un établisseur particulier ou l'imposition d'un ordre sur deux actions, réduit l'ensemble des plans candidats et ainsi peut changer (accroître) le minorant de la longueur d'un plan solution. On peut ainsi classer les résolvantes des défauts d'un plan partiel en fontion de leur vraisemblance à rapprocher d'un plan solution.

Dans cette section, nous présentons d'abord les fonctions utilisées pour estimer un minorant de la longueur d'un plan solution dans le PS-PG. Puis nous détaillons une méthode pour contraindre le PS-PG par ajout d'un engagement pour un établisseur ou de contraintes d'ordonnancements ou encore de séparations de variables. Ces deux éléments nous permettent alors d'élaborer une nouvelle méthode de contrôle heuristique d'une recherche non linéaire de plans.

Minorant de la longueur d'un plan solution

Les fonctions présentées ci-dessous calculent un minorant de la longueur d'un plan solution. La longueur d'un plan partiel P est assimilée au nombre d'opérateurs à appliquer au plan partiel initial pour obtenir P. On calcule ici un minorant strict puisque les résolvantes des menaces n'apparaissent pas dans le PS-PG. Les formules ci-dessous sont liées à l'interprétation du PS-PG comme une structure ET/OU : toutes les préconditions d'une action doivent être établies pour qu'elle puisse être exécutée, tandis qu'à un but non expliqué sont associés plusieurs alternatives d'établissements.

Dans les équations suivantes, P est un plan partiel, $OC(P)$ l'ensemble des buts non expliqués de P, p une proposition, $Est(p)$ l'ensemble des établisseurs potentiels de p qui apparaissent dans le PS-PG (i.e. $Est(p) = \{a \in \mathcal{A} \ / \ a \rightarrow p \in \mathcal{L}\}$), E un établisseur et enfin $OC(E)$ l'ensemble des buts non expliqués que E introduirait dans le plan.

$$\begin{aligned}
&(1)\ cout(P) = \sum_{p \in OC(P)} cout(p) \\
&(2)\ \ cout(p) = \min_{E \in Est(p)} cout(E) \text{ si } Est(p) \neq \emptyset, \\
&\qquad\qquad\quad \infty \text{ sinon} \\
&(3)\ cout(E) = 1 + \sum_{p \in OC(E)} cout(p)
\end{aligned}$$

Soulignons que ces fonctions prennent en compte les interactions positives. En effet une action n'est utilisée qu'une seule fois comme lien causal externe (voir la procédure "Expansion"). Si elle est ensuite utilisée pour réaliser des effets de bord, c'est alors à travers un lien interne. Puisqu'un lien interne n'apporte aucun but non expliqué, le coût de chaque précondition ne sera compté qu'une fois, même si l'action établit plusieurs propositions.

Par ailleurs ces fonctions fournissent un critère pour la condition d'arrêt de la procédure "Expansion" : tant que le coût calculé est infini, le planificateur devra complèter le plan partiel avec des établisseurs qui ne sont pas dans le PS-PG. Une stratégie acceptable est donc de développer le PS-PG jusqu"a atteindre un coût fini et de considérer a posteriori une expansion plus poussée si nécessaire.

Commitment dans le PSPG

La procédure "Expansion" proposée ci-dessus développe le PS-PG en fonction des contraintes définies par la situation initiale et les buts. Nous nous intéressons ici à des méthodes de modification du PS-PG par ajout de contraintes : le but est de répercuter les conséquences du choix d'une résolvante particulière pour un défaut sur la région accessible de l'espace de recherche.

Le choix d'un établisseur pour un but non expliqué écarte les autres établisseurs potentiels : ceci entraîne la disparition de toutes les préconditions attachées à ces alternatives et empêche la réutilisation de celles-ci pour des liens causaux internes. Lechoix d'une contrainte d'ordonnancement ou de séparation pour résoudre une menace va compromettre la validité de certains établisseurs par rapport à $\mathcal{O}$ ou $\mathcal{B}$. Nous présentons ci-dessous comment les effets du choix d'un établisseur sont propagés dans le PS-PG (procédure "Enforce"). La procédure correspondant au choix d'une contrainte résolvant une menace est semblable.

```
Imposer(Establisher E)
a → p = E
ajouter la contrainte (a > s) pour s ∈ A tq (p → s) ∈ Pre
ajouter les contraintes d'unification entre a et p
EtablisseursEcartes ← etablisseurs(p) − a
OCEcartees ← sous_buts(DiscardedEst)
supprimer EtablisseursEcartes du PS-PG.
OC ← Pre(A∞)
tantque(OC ≠ ∅)
   Pour chaque oc dans OC
      Etablisseurs ← etablisseurs(oc)
      Si oc ∈ OCEcartees alors
         OCEcartees ← sous_buts(Etablisseurs)
         Supprimer Etablisseurs du PSPG
      sinon
         Pour chaque E dans Etablisseurs
            Si desormais_impossible(E) alors
               OCEcartees ← OCEcartees
                                 ∪ sous_buts(E)
               Supprimer E du PS-PG
            sinon
               NouvelleOC ← NouvelleOC ∪ sous_but(E)
            finSi
         finPour
      finSi
   finPour
   OC ← NouvelleOC, NouvelleOC ← ∅
finTantque
```

Supprimer un lien causal externe entre une action a et une proposition p, c'est supprimer $a \rightarrow p$ de $\mathcal{EL}$, enlever a de $\mathcal{A}$ et, pour toute précondition p de a, supprimer p de $\mathcal{P}$ et $p \rightarrow a$ de Pre. La suppression d'un lien causal interne n'entraîne que son retrait de $\mathcal{IL}$.

En ligne 16, pour décider si un lien causal est encore valide, on vérifie que l'action impliquée n'a pas été supprimée du PS-PG et si il est encore cohérent avec les contraintes mises à jour en lignes 2 et 3.

La condition d'arrêt en ligne 8 est ici légitime : contrairement à la procédure "Expansion" qui étendait le PS-PG, nous nous contentons ici d'examiner des éléments déjà présents. Puisque le développement du PS-PG a déjà pris fin, nous sommes sûrs d'atteindre un point d'arrêt.

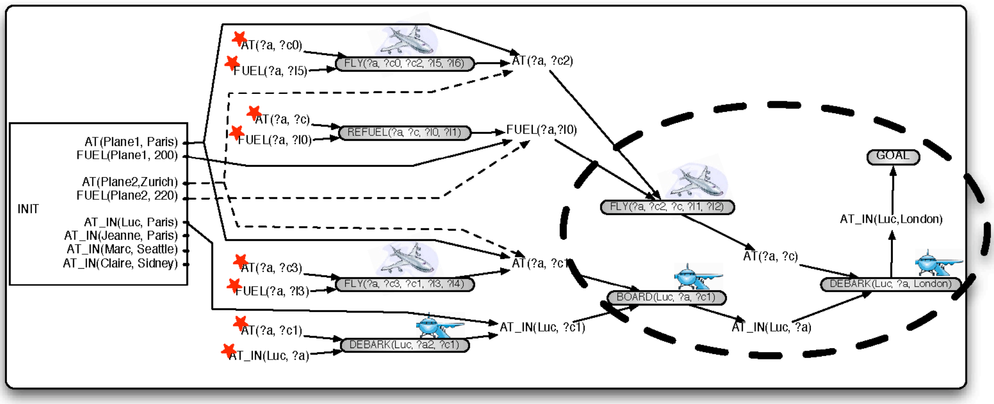

Fig. V.6 – Le PS-PG permet de considérer prioritairement les branches déterministes d'un plan partiel

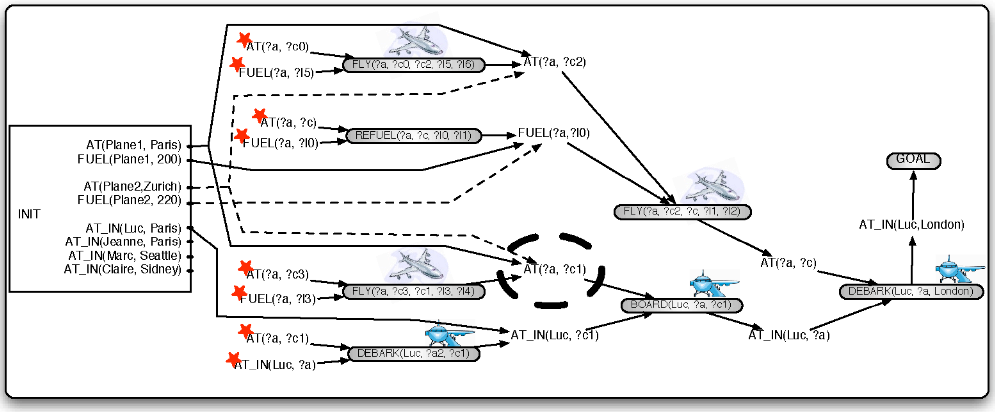

FIG. V.7 – Ici on se focalise sur un défaut présentant trois alternatives de résolution

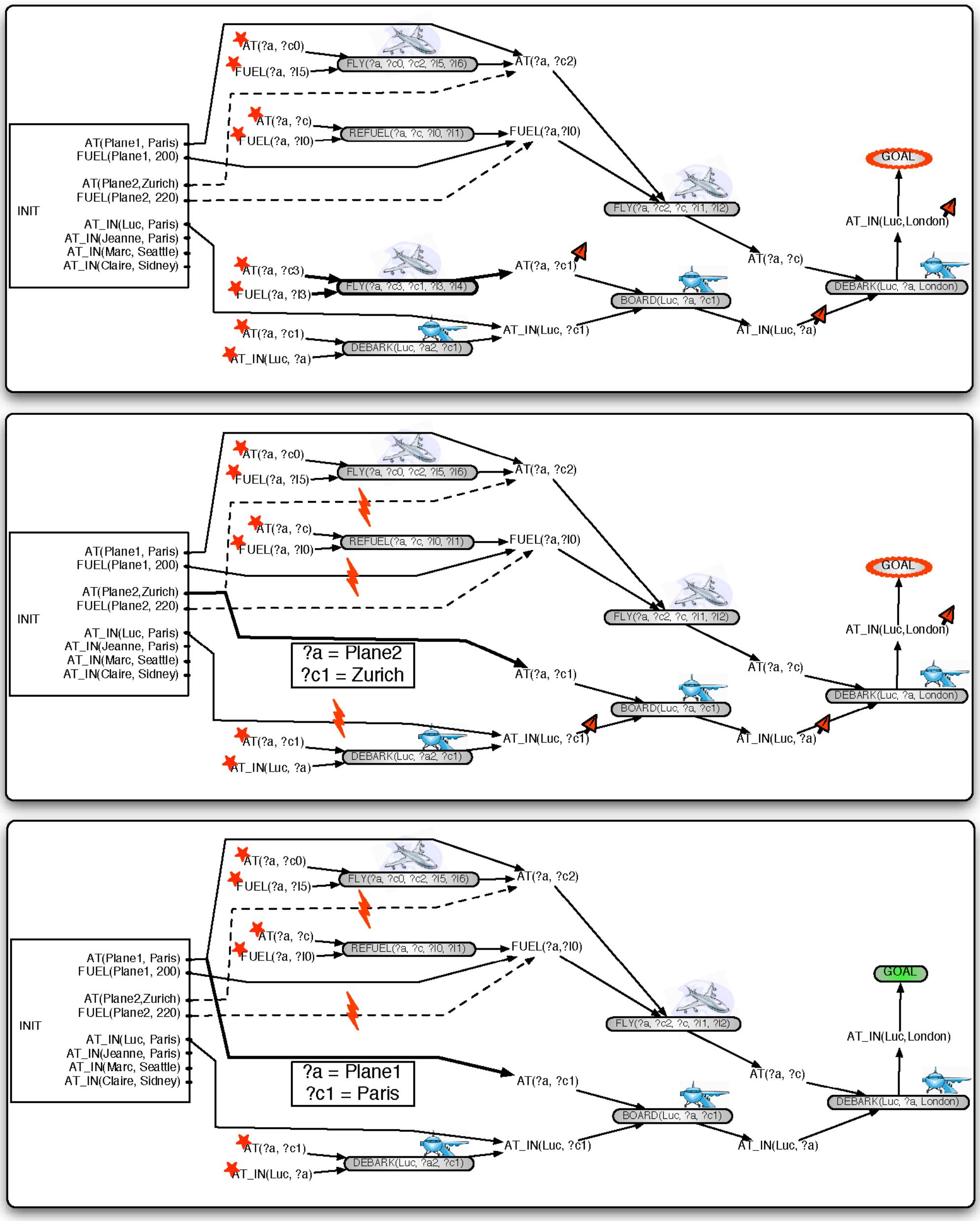

Fig. V.8 – 3 méthodes alternatives pour résoudre un défaut

Chapitre VI

Utilisation du PSPG

Là, petit doute : faut-il un seul chapitre avec deux sections, ou bien éclater ça en deux chapitres ?
Risque de déséquilibre, faut quand même beaucoup discuter la transformation en CSP.

VI.1 Heuristique

Discussion de mise en oeuvre + résultats
Discuter abstractions ?

Munis des procédures "Expansion" et "Imposer" ainsi que des fonctions de coût présentées précédemment, nous sommes maintenant en mesure d'exposer comment utiliser le PS-PG pour guider la recherche d'un planificateur non linéaire. L'idée clef de notre approche est qu'à chaque étape de la recherche les résolvantes sont classées en fonction du minorant de la longueur d'une solution calculé dans le PS-PG après leur insertion. Le PS-PG sera mis à jour incrémentalement au cours de la recherche en forçant effectivement les opérateurs appliqués au plan partiel. La procédure NONLIN ci-dessous est une ébauche naïve d'un algorithme pour illustrer un tel contrôle de la recherche.

NONLIN(I,G)
$P \leftarrow$ plan partiel initial
$(\{A_0, A_\infty\}, \{A_0 < A_\infty\}, \emptyset, Pre(A_\infty), \emptyset)$
$Defauts \leftarrow Pre(A_\infty)$
Expansion du PS-PG
tant que $(Defauts \neq \emptyset)$
  $Resolvantes \leftarrow resolvantes(Defauts)$
  Si $(Resolvantes = \emptyset)$ alors Backtrack.
  Pour chaque R dans $Resolvantes$
    Evaluer le coût de R dans le PS-PG
    /* i.e. imposer temporairement R et calculer *coût* */
  finPour
  choisir R avec le coût minimal
  ajouter R dans P et mise-à-jour de $Defauts$
  Si P est inconsistant alors Backtrack.
  imposer R dans le PS-PG
finTantque

Cet algorithme, quoique simple, nous permet de souligner un des avantages du contrôle heuristique proposé : toutes les résolvantes y sont évaluées de façon uniforme. On peut donc considérer sa transposition à un contexte plus expressif, par exemple des problèmes temporels ou incluant des ressources numériques tels que ceux traités par IxTeT ([Ghallab 94]) ou HSTS ([Muscettola 94a]). Des modules spécialisés seraient nécessaire pour la détection des défauts et le calcul de leurs résolvantes. Par exemple des méthodes telles que celle décrite dans [Laborie 01] pourraient être mises en oeuvre dans un module efficace de détection de surconsommation de ressource, lequel proposerait des conjonctions de contraintes pour résoudre ces défauts. Ces contraintes pourraient ensuite être évaluées via le PS-PG au même titre que les résolvantes de menaces traitées précédemment.

Plusieurs modifications peuvent bien sûr être apportées à l'algorithme ci-dessu afin de gagner en efficacité. Tout d'abord, cette évaluation heuristique pourrait être incorporée à un schéma plus classique, par exemple en réintroduisant le choix en deux temps défaut / résolvante, afin notamment de retrouver la propriété de systématicité de la recherche. Mais le PS-PG pourrait aussi être utilisé plus activement. Par exemple, il pourrait fournir les résolvantes possibles des buts non expliqués à chaque étape de la recherche. De plus, plutôt que d'insérer les établisseurs un par un, les branches déterministes du PS-PG (i.e les portions du graphe sans alternatives) pourrait être insérées conjointement.

VI.2 CSP

Fin papier IJCAI ou RJCIA
+ rÈsultats + abstractions ?

On peut distinguer deux façons de contrôler la recherche d'une solution d'un problème de planification. La première consiste à guider la recherche à travers des fonctions heuristiques. La seconde est d'exhiber des contraintes qui vont structurer l'espace de recherche. Le planning graph a été utilisé dans ces deux approches. Il a tout d'abord été présenté comme une base pour définir des CSPs ou des problèmes SAT dans lesquels étaient encoder des propriétés de la région accessible de l'espace d'états. Puis de nombreux travaux en ont fait un usage heuristique, avec succès. Nous montrons dans cette section que, outre son utilisation comme support heuristique, le *Plan Space Planning Graph* peut aussi être utilisé pour structurer l'espace de recherche : on peut en dériver un problème de satisfaction de contraintes au sein duquel les décisions différées vont jouer un rôle actif à travers la propagation de contraintes.

Parmi les sytèmes qui transforment les problèmes de planification en problèmes de satisfiablisté propositionelle ou de satisfaction de contraintes, on relève un grand nombre d'encodages déduis d'une planification dans l'espace d'états, alors que ceux dérivés de l'espace des plans sont moins populaires. Plusieurs planificateurs non linéaires utilisent des techniques de satisfaction de contraintes, mais c'est alors pour gérer des contraintes spécifiques et la recherche est toujours effectuée dans l'espace des plans. Les travaux auxquels peut être comparée notre proposition et dont nous nous sommes inspirés sont des approches plus anciennes telle que [Kambhampati 94] et plus particulièrement le système *Descartes* ([Joslin 95]).

L'algorithme de recherche de Descartes entrelace l'expansion d'une structure causale avec la construction d'un CSP dynamique. Le but est de maintenir un moindre engagement dans la construction de plans tout en exploitant autant que possible les contraintes liées aux menaces, plutôt que de les mettre dans la file des défauts à résoudre. Ainsi les impasses peuvent être détectées plus tôt et des transformations inutiles peuvent être évitées pour des plans partiels nécessairement incohérents.

¿ partir du PS-PG, considérée comme une structure causale à contributeurs multiples, nous proposons la construction d'un CSP menant à une solution présentant les mêmes avantages que Descartes. Ce CSP contient $\mathcal{B}$ and $\mathcal{O}$. Il contient également une variable, dite variable causale, pour chaque proposition oc de $\mathcal{P}$ dont le domaine est l'ensemble des établisseurs d'oc dans le PS-PG. Des contraintes sont ajoutées qui lient les variables causales aux variables de $\mathcal{B}$ et $\mathcal{O}$: elles représentent les unifications et ordonnancements liées aux liens causaux. Le CSP contient aussi des *variables de décision* qui permettent de représenter les choix de résolvantes pour les menaces. Une variable de décision est associée à chaque menace. Le domaine initial de ces variables est l'ensemble : $\{Promotion, Demotion, Separation\}$. Des contraintes sont ajoutées pour relier ces variables aux contraintes correspondantes dans $\mathcal{B}$ et $\mathcal{O}$.

Un tel CSP permet aux choix différés de jouer néanmoins un rôle actif dans la mesure où la propagation des contraintes peut forcer certains choix sans que le processus de recherche ait eu à se focaliser dessus. Par exemple un établisseur sera automatiquement supprimé du domaine d'une variable causale si les unifications ou ordonnancements correspondants ne sont plus possibles. Des propagations similaires peuvent affecter les variables de décision.

Resolution(CSP) -

1. $Focus \leftarrow VariablesCausales(Pre(A_\infty))$
2. Tant que ($Focus \neq \emptyset$)
3. choisir un element v dans $Focus$
4. choisir une valeur e pour v
5. Propager les contraintes
6. Si ($Incohérent(CSP)$) alors Backtrack
7. $Focus \leftarrow Focus$
   $\cup VariablesCausales(Pre(est))$
   $\cup VariablesDecision\ impliquant\ e$
8. FinTantque

La procédure *"Resolution"* présente une technique de recherche pour calculer un plan partiel grâce au CSP. En ligne 1 et 7, nous notons *VariablesCausales(Pre(a))* les variables causales associées aux préconditions de l'action a. En ligne 7, les variables de décision impliquant l'action a sont celles associées aux menaces impliquant a. Le choix de la prochaine variable à instantier en ligne 3 est bien sûr un élément clef pour l'efficacité de la résolution ; on peut utiliser des stratégies classiques propres aux CSPs telle que la préférence donnée aux variables ayant les plus petits domaines de façon à réduire le facteur de branchement.