Planification avec STRIPS

Exercice 1 Le monde des blocs

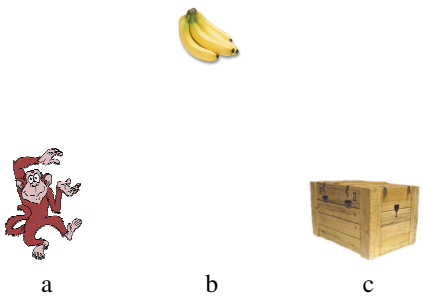
Nous nous trouvons dans la situation initiale donnée ci dessous à gauche. L'objectif que nous nous fixons est de planifier automatiquement les opérations nécessaires pour obtenir au final un empilement des blocs tel que A se trouve sur B, B se trouve sur C, C se trouve sur D et D est posé sur la table. Cela nous donne la situation (ou état) finale suivante (à droite).

Vous répondrez aux questions suivantes :

1. Décrivez la situation initiale en utilisant notamment les opérateurs ON et CLEAR. Décrivez, de la même façon, la situation finale.
2. Rappellez les quatre opérateurs du monde des blocs vus en cours. A quels autres types d'objets ces opérateurs vous font-ils penser ?
3. Résolvez le problème posé à la main en utilisant une stratégie de chaînage arrière. Pour cela, vous utiliserez une pile de buts conjointement à une description de l'état courant du problème et au plan en cours de construction.
4. Que pensez-vous du choix du chaînage arrière pour résoudre ce problème ?

Exercice 2 Problème du singe et des bananes

Le problème du singe et des bananes est un problème très classique en planification. Trois entités (un singe, des bananes et une caisse) sont initialement situées à trois emplacements distincts (respectivement a, b et c) dans une pièce. Le singe et la caisse se trouvent au niveau dit "bas", alors que les bananes sont suspendues au niveau "haut". Le singe est très attiré par les bananes, mais il n'est (malheureusement pour lui) ni situé au bon endroit, ni au bon niveau. La situation initiale est illustrée par la figure suivante :



Le problème consiste alors à trouver pour le singe un plan qui lui permettra de récupérer les bananes. Il sera du type : se déplacer de a en c, prendre la caisse et la déposer en b, monter sur la caisse, prendre les bananes. Pour cela, vous répondrez aux questions suivantes :

1. A partir des prédicats *situé(X,Y)*, *niveau(X,Y)* et *possède(X,Y)*, décrivez l'état initial du problème ainsi que le but à atteindre.
2. Ecrivez en utilisant la syntaxe STRIPS les opérateurs suivants dont vous aurez besoin pour résoudre le problème :

(a) *seDeplace(X,Y) :* le singe se déplace de l'emplacement X à l'emplacement Y.

(b) *prend(X) :* le singe prend l'objet X. Attention, le singe ne peut prendre qu'un objet à la fois.

(c) *depose(X) :* le singe dépose l'objet X.

(d) *monteCaisse :* le singe monte sur la caisse.

3. Résolvez le problème à l'aide d'une stratégie en chaînage arrière.

Annexe

Représentation des actions en STRIPS.

Les schémas d'action sont vus comme des opérateurs permettant de passer d'un état à un autre. Le langage utilisé pour décrire les actions est un fragment très restreint de la logique du premier ordre.

`action(paramètres)`	
`precond`	conjonction de littéraux positifs sans symbole de fonction qui exprime ce qui doit être vrai pour que l'action puisse être exécutée
`delete`	conjonction de littéraux positifs sans symbole de fonction qui exprime ce qui n'est plus vrai après la réalisation de l'action
`precond`	conjonction de littéraux positifs sans symbole de fonction qui exprime ce qui est rendu vrai par l'exécution de l'action

De plus, l'hypothèse cruciale suivante est faite : *toute formule non mentionnée dans la description de l'action reste inchangée.*

Chaînage arrière à sous buts indépendants (algorithme STRIPS).

L'algorithme de STRIPS utilise une planification régressive : partant de l'ensemble des buts, on remonte en choisissant des actions qui satisfont certains buts, et on regarde dans quel cas on atteint la situation initiale. Afin de restreindre le facteur de branchement de ces choix, cet algorithme fait l'hypothèse que les sous-buts sont indépendants. Il adopte une stratégie de type "divide and conquer", en se focalisant sur chacun des sous-buts sans se préoccuper des autres sous-buts.

STRIPS($Init$, $Buts$)

Entrée : $Buts$ - un ensemble de buts
$Init$ - une situation initiale
(implicite : un ensemble de règles d'actions que l'on suppose connu pour simplifier)
Sortie : envoie un plan (une séquence d'actions).

$Plan \leftarrow []$, $E \leftarrow Init$
Tant qu'il y a des buts dans $Buts$ non satisfaits dans E
  Choisir un but b sz $Buts$ non satisfait dans E
  Choisir une action a pertinente pour b (contient b dans `add`)
  $Plan \leftarrow Plan \otimes STRIPS(E, precond(a)) \otimes a$ (où $\otimes$ est la concaténation des actions)
  $E \leftarrow Plan(Init)$ (application des actions à E)
Fin Tant que
RENVOIE $Plan$

STRIPS applique deux heuristiques générales :

1. Si l'on rencontre lors de la recherche un noeud qui contient un ancêtre (qui est donc au moins aussi difficile à résoudre), on peut couper la branche (éviter les boucles).
2. Si une action est applicable dans l'état initial, on l'applique sans permettre le backtracking.

Le planificateur a deux choix importants à effectuer : choisir le sous-but à explorer et choisir l'action permettant de réaliser ce sous-buts. Ces deux choix peuvent être guidés par des heuristiques plus fine liées au domaine étudiés. Si les buts ne sont pas indépendants, cet algorithme peut rencontrer des difficulté et ne pas trouver de plan optimal (anomalie de Sussman).