# Master Informatique 1

# Module BIA

# TD 2

# Représentation et résolution de problème (2)

Nadia Kabachi, Alain Mille, Amjad Rattrout

## 1 Introduction

Le TD est prévu sur 3 tiers-temps avec des passages d'étudiants au tableau.
Comme le TD1 n'a pas pu être vraiment fini et que le TD2 est une application de la fin du TD1, nous reprenons la dernière question du TD1 pour ce TD2.
Ce document ne donne donc que le sujet.

## 2 Résolution par décomposition de problèmes (30 min)

Rappel : la décomposition d'un problème en sous-problèmes plus simples est un principe applicable à des problèmes modélisables de manière récursive, mais pas seulement !

Un algorithme de recherche aveugle permettant de faire une recherche dans un graphe ET/OU (hypergraphe particulier) **sans circuit** issu de la décomposition d'un problème est le suivant : (en absence de coût et pour borner l'espace exploré, on utilise un majorant sur le rang des états, noté rg(u) ; BSH retourne « Echec » si le rang est supérieur à un Seuil)

```
BSH(u) Backtrack Search dans un Hypergraphe
1.      Si u terminal Alors Retourner « Succès »
2.      Si aucune règle de décomposition n'est applicable en u ou si rg(u) > Seuil
        Alors Retourner « Echec »
3.      Itérer sur les règles de décomposition i applicables en u
3.1.        Flag ← vrai
3.2.        Tant que Flag, itérer sur les nœuds v, successeurs de u en lesquels la règle i décompose u
                Si v ∉ RESOLUS Alors faire :
                    Si v ∈ INSOLUBLES Alors Flag ← faux
                    Sinon faire :
                        rg(v) ← rg(u) + 1
                        Si BSH(v) = « Echec » Alors faire :
                            Mettre v dans INSOLUBLES
                            Flag ← faux
                        Sinon mettre v dans RESOLUS
                Fin Itération 3.2
3.3         Si Flag Alors faire :
                Mettre u dans RESOLUS
                règle(u) ← i
                Retourner « Succès »
            Fin Itération 3
4.      Mettre u dans INSOLUBLES
5.      Retourner « Echec »
```

Dans cet algorithme, RESOLUS est l'ensemble 1) des états terminaux, et 2) des états u tels qu'il existe un connecteur $S_i(u)$ dont tous les successeurs v sont dans RESOLUS ; INSOLUBLES est l'ensemble 1) des états non terminaux sans successeur, et 2) des états u tels que pour tout connecteur $S_i(u)$ il existe un successeur v qui est dans INSOLUBLE.

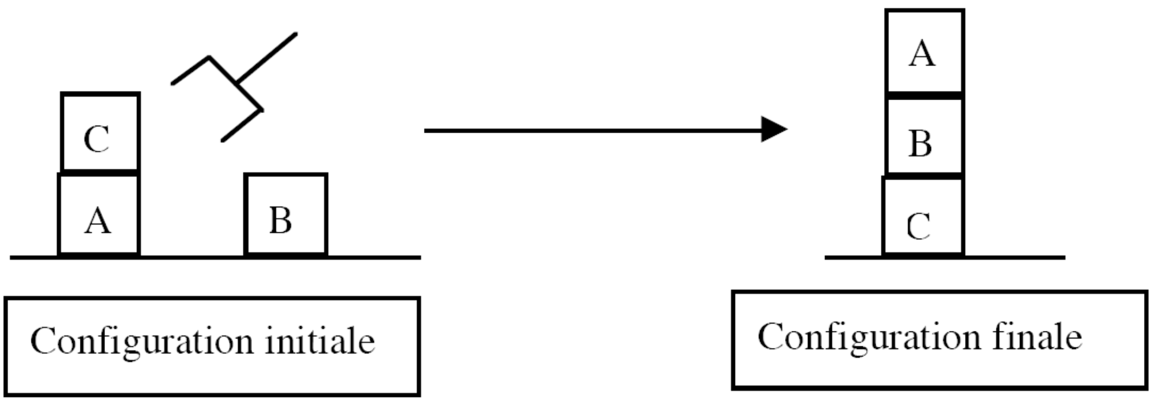
Comprendre et appliquer cet algorithme à un problème qui serait décomposé selon les règles de décomposition suivantes :

| | |
|---|---|
| R1 : d → g, h<br>R2 : d → a, e, f<br>R3 : d → a, k<br>R4 : f → i<br>R5 : f → c, j<br>R6 : a → b, c<br>R7 : k → e, l | Les problèmes terminaux sont : b,c,e,l<br><br>Le problème à résoudre est : d<br><br>Considérer les règles dans l'ordre croissant et les sous-problèmes de « gauche à droite ».<br><br>Dessiner le graphe/arbre de décomposition. |

Faire « tourner à la main » l'algorithme en traçant les structures et variables importantes.

# 3 Problème de planification : illustration avec le monde des cubes

Un bras articulé muni d'une pince permet de saisir un cube, de le déplacer, de le poser sur une table ou de l'empiler sur un autre cube. La pince ne peut prendre qu'un cube à la fois et seul un cube sur lequel rien n'est empilé peut être saisi.




## *3.1 Comment décrire un état (penser « variable » d'état) ?*

Proposer des opérateurs pour passer d'un état à l'autre (forcément liés à la pince qui est le seul objet capable d'agir) ? Pour répondre à cette question, tenter informellement de faire les opérations nécessaires et mettre au point le test de satisfaction de but atteint.

## *3.2 Sur la base de cette modélisation, construire le graphe ET/OU correspondant (partiel)*