Énoncés et représentations

Plan

- Qu'est-ce qu'un problème ?
- Différents types d'énoncés
- Recherche heuristique

- IA : ordinateur manipulant des informations symboliques (connaissances), raisonnant sur ces connaissances (résoudre, construire la solution)

IA : machine

énoncé du pb → solution

+ connaissances, raisonnement, stratégies résout

Formalisation

- Un problème est posé en termes informels, en langage naturel

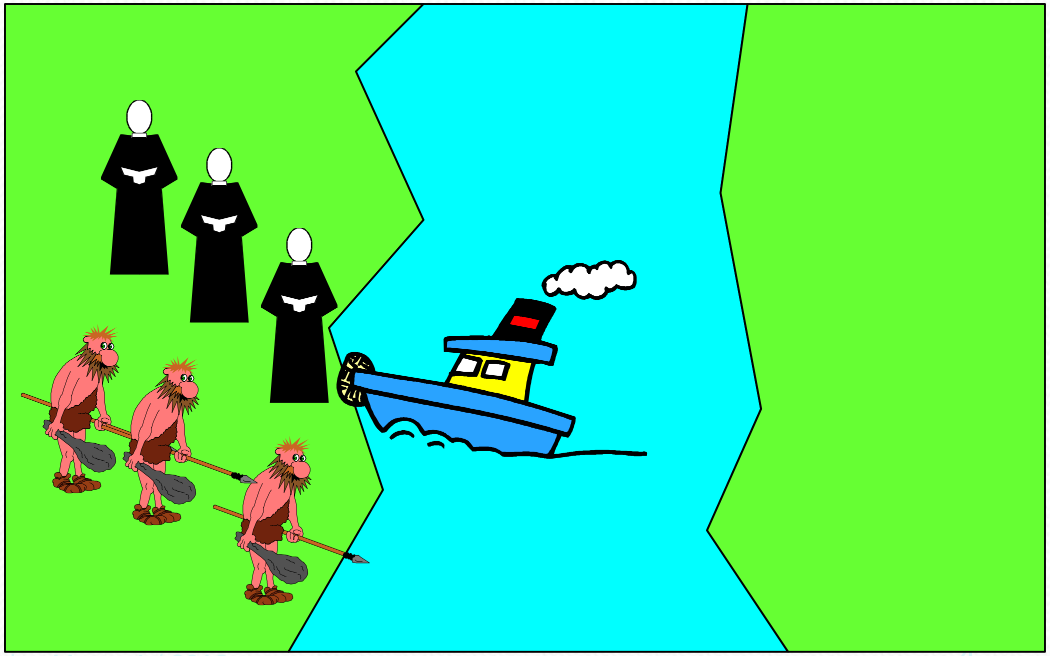
→ **formalisation** nécessaire avant la **résolution** par la machine

- Cette formulation doit être « la bonne » car contrairement à l'homme, la machine ne peut pas revenir sur cette étape

Énoncé d'un problème

Les k missionnaires et les k cannibales

- Faire traverser tout le monde avec la barque (n = 2 places)
- Il ne faut pas qu'il y ait plus de cannibales que de missionnaires sur une rive (sauf s'il y a 0 missionnaires !)
- Résoudre pour k = 3



Exemple en maths

- Énoncé informel
  - *trouver les racines de* $x^2 - 9$

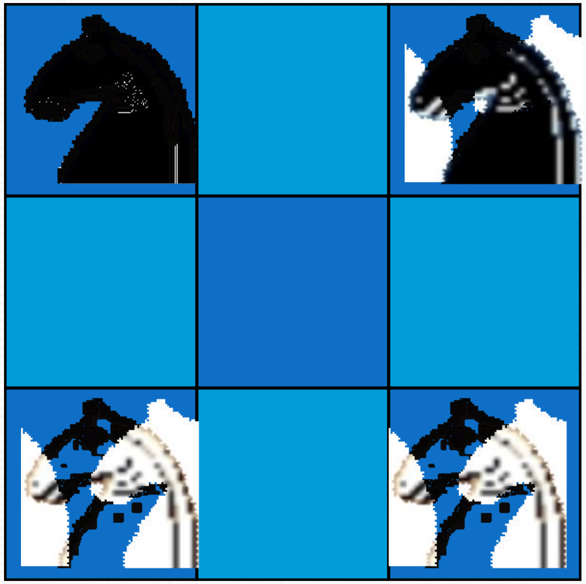
Exemple en maths

- Énoncé informel
  - *trouver les racines de* $x^2 - 9$
- Énoncé formalisé
  - *trouver* $\{x \mid x^2 - 9 = 0\}$

→ N'est-ce pas déjà la solution ?

→ Et $\{x \mid (x-3)(x+3) = 0\}$ alors ?

Exemple des 4 cavaliers



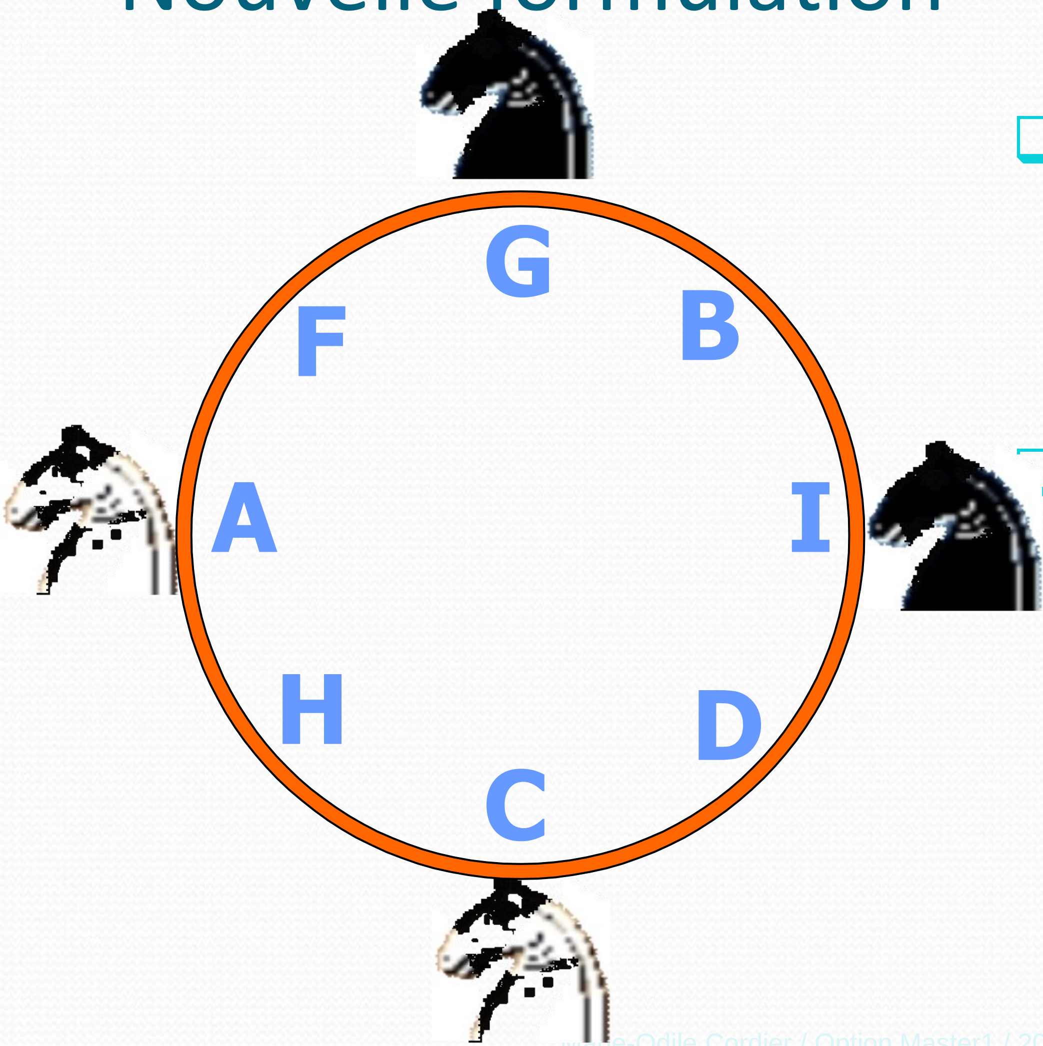
- Énoncé informel
  - *échanger (si cela est possible) en un nombre minimum de coups les 2 cavaliers blancs avec les noirs*

Exemple des 4 cavaliers (formalisation du problème)

- Se démarquer de l'échiquier
- S'intéresser aux déplacements

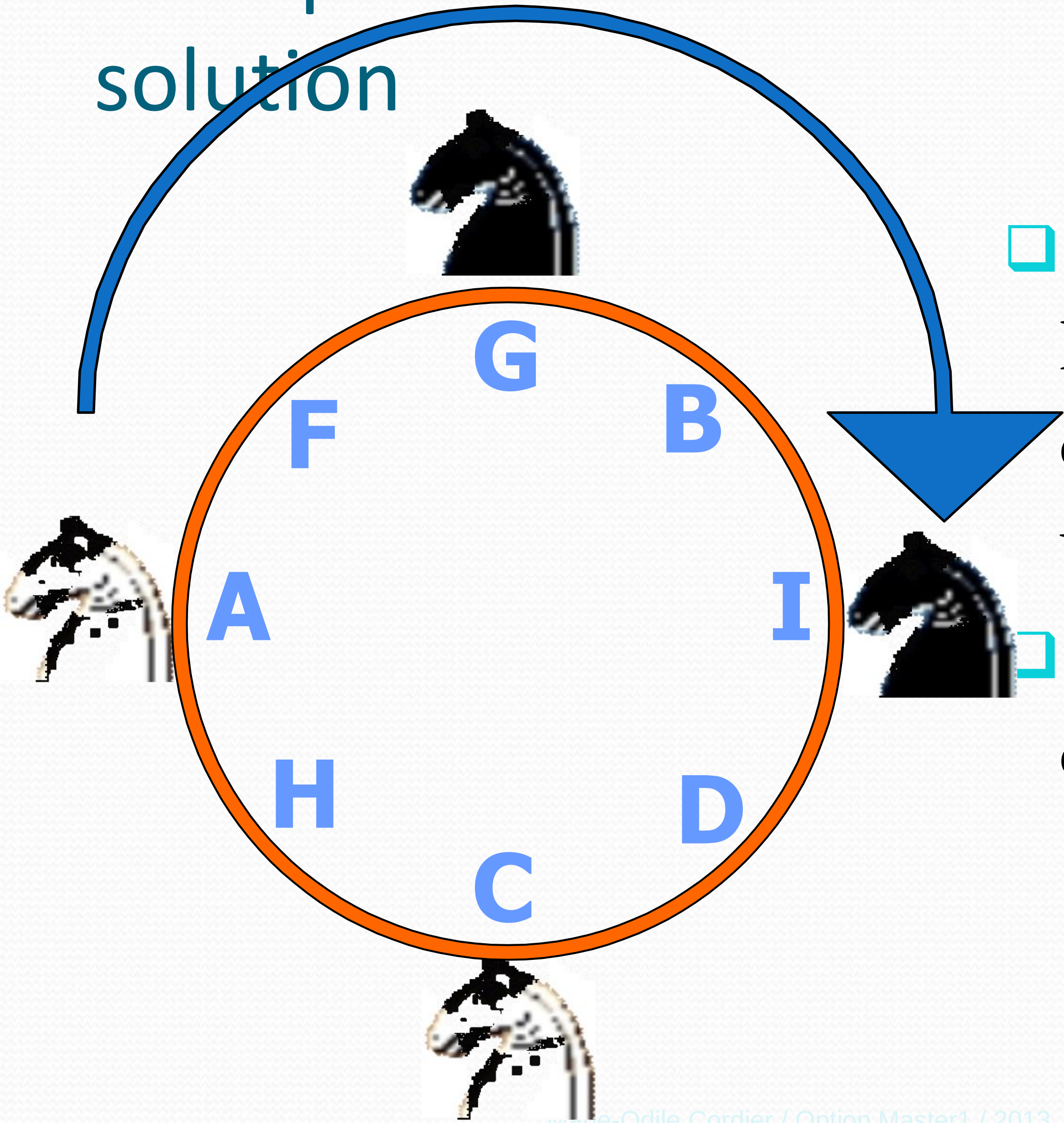
→ nommer les cases

Exemple des 4 cavaliers
Nouvelle formulation



- Points adjacents = cases accessibles en 1 coup
- Échanger les positions sur le cercle

Exemple des 4 cavaliers solution



- Faire effectuer a l'ensemble de la configuration un demi-tour
- 4 coups par cavalier = 16 coups

- Et aussi
  - Faire quatre triangles avec six allumettes
  - Passer par les neuf points d’un carré en quatre segments sans lever le crayon

Énoncés et représentations - Plan

- **Qu’est-ce qu’un problème ?**
- **Différents types d'énoncés**
  - énoncé de type combinatoire
  - énoncé avec états et opérateurs de changement d'états
    - énoncé avec buts et décomposition de buts
- **Recherche heuristique**

Énoncé combinatoire

- *Trouver, dans un ensemble (espace) X donné, les éléments (points) e satisfaisant un ensemble de contraintes K*

Ex : placer 8 reines sur un échiquier 8x8 sans qu'elles ne s'attaquent

Formalisation : espace + contrainte

Ex : $X \subset (\{1...8\}\times\{1...8\}) \times ... \times(\{1...8\}\times\{1...8\})$

card(X) = $64 x..x 57 \approx 1.8*10^{14}$

K : X → bool : indique si les reines sont en prise ou non

Énoncé combinatoire - résolution par énumération explicite

- « ***Generate & test*** » : génération de tous les e de X et élimination de ceux tq K(e) = faux
- ex. : prendre une configuration des reines sur l'échiquier et tester si aucune ne s'attaquent
  - Facile à mettre en œuvre
  - Seulement faisable si X est fini et petit

- Améliorations possibles :
  - Intégrer les contraintes dans la représentation, par exemple une reine par ligne, et une par colonne : {1…8}x {1…8}x … {1…8} soit 8 x 7 x…x 1
  - Construire les solutions petit à petit : si une solution partielle ne satisfait pas les contraintes, cela élimine toutes les solutions étendant cette solution partielle

Énoncés et représentations - Plan

- **Qu’est-ce qu’un problème ?**
- **Différents types d'énoncés**
  - énoncé de type combinatoire
  - énoncé avec états et opérateurs de changement d'états
    - énoncé avec buts et décomposition de buts
- **Recherche heuristique**

Énoncé états/opérateurs

- *À partir d’un état initial et final(s) et d’opérateurs de changement d’états, trouver une suite d’opérateurs permettant de passer de l’état initial à un état final*
- Formalisation : spécifier
  - l’état initial
  - le ou les états finaux
  - les opérateurs sous la forme <préconditions,effets>

Énoncé états/opérateurs
exemple du taquin

État initial

4	3	5
1	6	2
7	8	

État final

1	2	3
8		4
7	6	5

- Matrice M3 x 3
- Case vide : (l_v, c_v)
- Opérateur : déplacement de la case vide (HAUT, BAS, GAUCHE, DROITE)

- Opérateur HAUT
  - Précondition : lv ≠ 1
  - Effets : M(lv,cv) ← M(lv-1, cv)

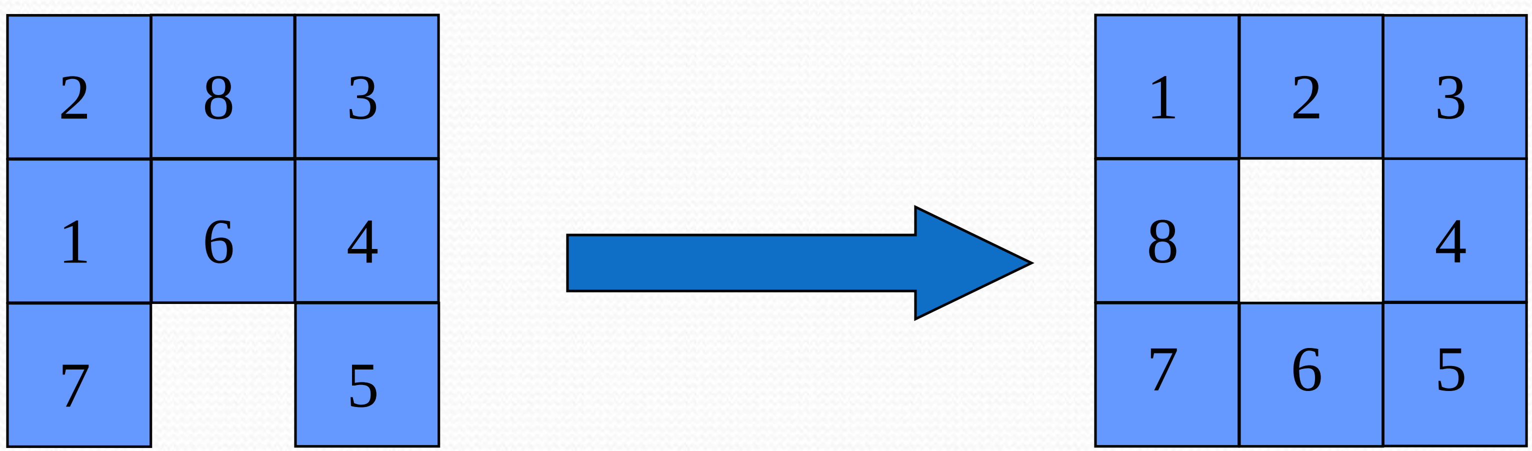
    lv← lv -1
- …

Énoncé états/opérateurs Exercices - solution

- Opérateur DOWN
  - Précondition : lv $\neq$ 3
  - Effets : $M(l_v, c_v) \leftarrow M(l_v+1, c_v)$; $l_v \leftarrow l_v+1$
- Opérateur LEFT
  - Précondition : $c_v \neq 1$
  - Effets : $M(l_v, c_v) \leftarrow M(l_v, c_v-1)$; $c_v \leftarrow c_v-1$
- Opérateur RIGHT
  - Précondition : $c_v \neq 3$
  - Effets : $M(l_v, c_v) \leftarrow M(l_v, c_v+1)$; $c_v \leftarrow c_v+1$

Énoncé états/opérateurs

- Développer quelques niveaux du graphe d'états en essayant d'appliquer d'abord BAS puis HAUT puis DROITE puis GAUCHE.

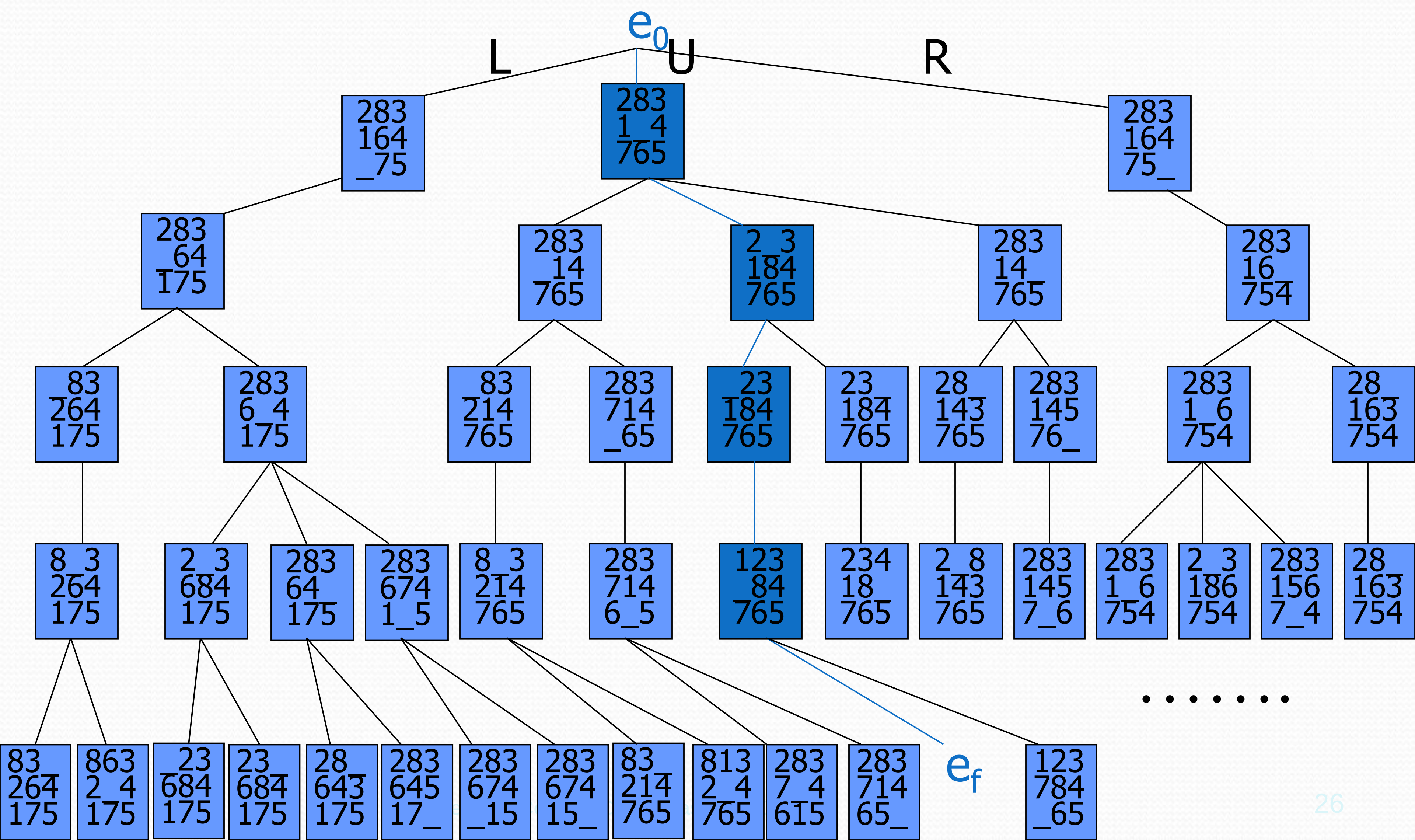


2	8	3
1	6	4
7		5

→

1	2	3
8		4
7	6	5

Énoncé états/opérateurs Exercices - solution



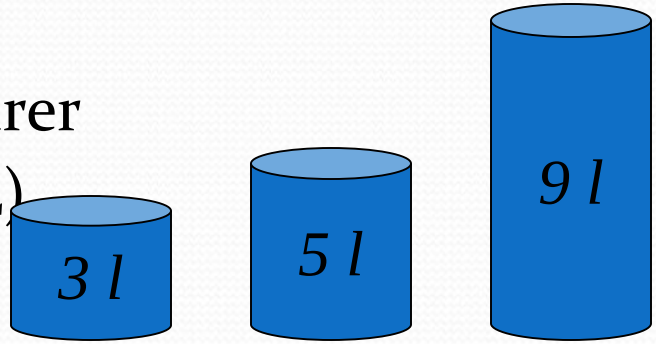
Énoncé états/opérateurs graphe d'états

- Le graphe implicite du pb s'appelle graphe d'états
  - Nœuds = états (9! = 362880)
  - Feuilles = états terminaux ou nœuds échecs (pas d'opérateur applicable)
  - Arcs = opérateurs (arc valué = coût de l'opérateur)
  - Racine = état initial
- Une solution est un chemin de la racine à une feuille d'état final
- Le coût de la solution est le coût du chemin (souvent la somme des coûts des opérateurs le long du chemin)

Énoncé états/opérateurs Exercices

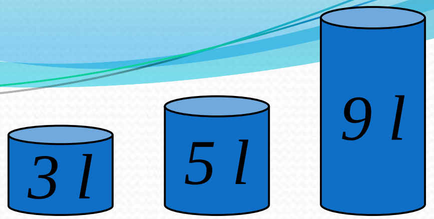
Problème des verres mesureurs

- À l'aide de ces trois récipients, mesurer 7L (mettre 7L dans le récipient de 9L)
- On peut remplir (entièrement) un récipient à la source et en vider une partie dans un autre récipient



- Les récipients sont vides au départ
- Formaliser le problème (états, opérateurs…)
- Développer quelques nœuds du graphe d'états

Énoncé états/opérateurs



<u>Problème des verres mesureurs</u>

- état = triplet (Q1,Q2,Q3)
- état initial = (0,0,0)
- états finaux = (x,y,7)
- Opérateurs
  - remplir_1 : $(q1 \neq 3, q2, q3) \rightarrow (3, q2, q3)$
  - remplir_2 : $(q1, q2 \neq 5, q3) \rightarrow (q1, 5, q3)$
  - remplir_3 : $(q1, q2 , q3 \neq 9) \rightarrow (q1, q2, 9)$
  - vider_1_ds_2 : $q1 \neq 0$, $q2+q1 < 5$, $(q1, q2, q3) \rightarrow (0, q1+q2, q3)$
  - remplir_2_avec_1 : $q1+q2 \geq 5$, $q2 \neq 5$, $(q1, q2, q3) \rightarrow (q1-(5-q2),5,q3)$
  - …

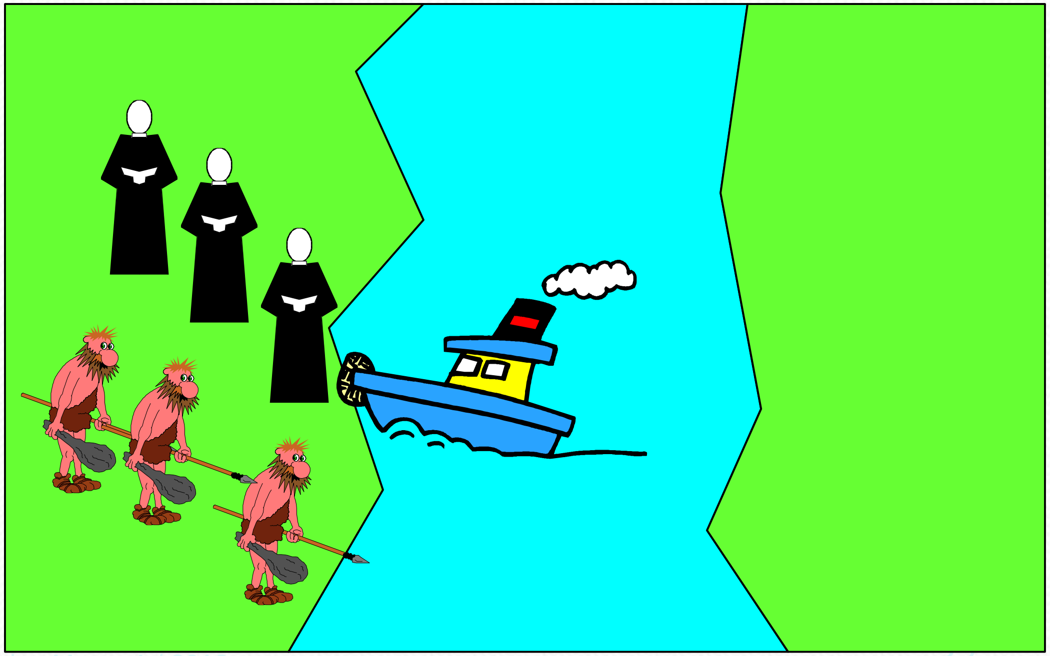
Une solution meilleure qu'une autre ?

- Dépend de la fonction de coût :
  - eau consommée -> solution qui gaspille le moins d'eau
  - force nécessaire pour soulever les seaux -> solution qui demande le moins de force (le seau 2 rempli est moins lourd que le 3, etc.)
  - coût fixe pour chaque action -> solution qui demande le moins d'actions
  - …

Énoncé états/opérateurs exercices (fait en TD)

Les n missionnaires et les n cannibales

- Faire traverser tout le monde avec la barque (k=2)
- Il ne faut pas qu'il y ait plus de cannibales que de missionnaires sur une rive (sauf s'il y a 0 missionnaires !)
- Formaliser et résoudre pour $n = 3$



Énoncé états/opérateurs


Les n missionnaires et les n cannibales

- Triplet (M,C,P) avec nbre de M et de C sur la rive de départ + position de la barque : 1 pour départ, 0 pour arrivée
  - État initial (n,n,1)   État final (0,0,0)

Contraintes :

0 <= M <= n et 0 <= C <= n et (M >= C ou M = 0) et (n-M >= n-C ou n-M = 0)   SOIT **M= 0 ou M=n ou M=C**

Et   p=0 => M+C # 2n ET p=1 => M+C #0

Nombre d'états possibles :

(n+1 (M=0) + n+1 (M=n) + n-1 (M=C))*2 – 2 (la barque ne peut pas être laissée seule (0,0,1) et (n,n,0) imp) => 6n

Si n = 3 : 18 états et si n=5 30 états

Remarque : ne dépend pas de k (capacité de la barque) / contraintes sur les états pas sur les opérateurs de changement

Énoncé états/opérateurs exercices - solution

Formaliser les opérateurs :

Deux opérateurs (1 gauche-droite et 1 droite-gauche)

Opeg>d (Mt,Ct) (Mt: nbre de missionnaires transportés, Ct : ..)

préconditions :

- position de la barque : P = 1
- capacité de la barque : 0 < Mt + Ct <= k

+ l'état que l'on laisse avant débarquement est ok

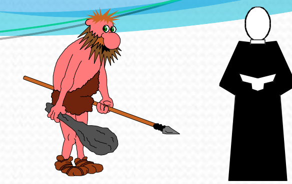
+ l'état après débarquement est ok

qu'on exprime par POSSIBLE(<Mg-Mt,Cg-Ct,0>

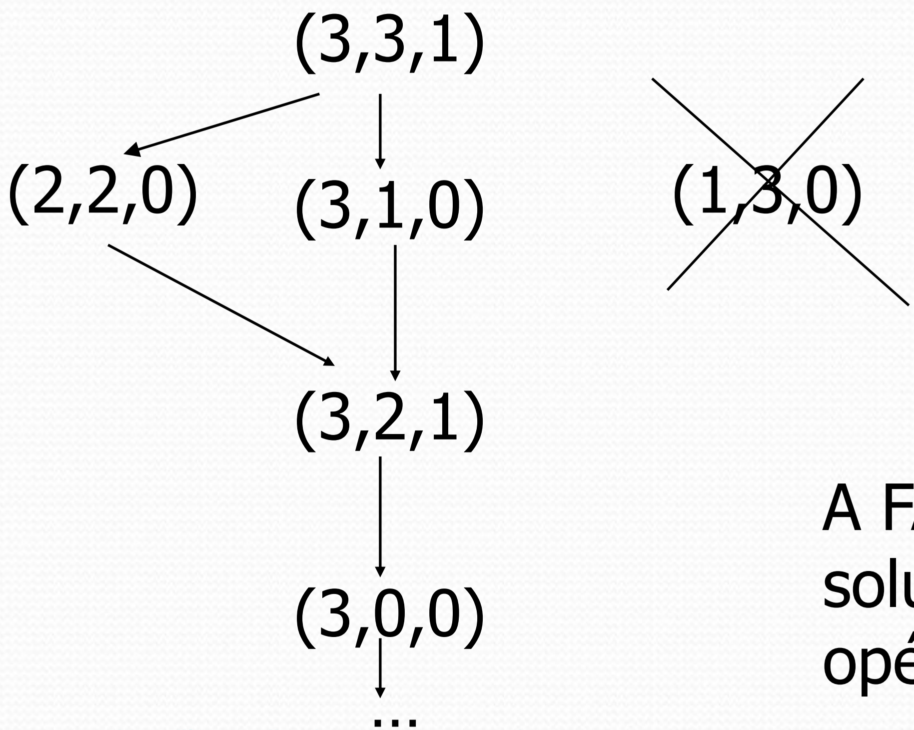
(on pourrait distinguer les deux cas, mais laisser partir le bateau sans qu'il puisse débarquer ne sert à rien)

effets : P <- 0 et Mg <- Mg- Mt et Cg <- Cg-Ct

Énoncé états/opérateurs exercices 3 - solution



- Les missionnaires et les cannibales : le graphe implicite (pour n = 3 et k = 2)



A FAIRE… la solution a 11 opérateurs

Énoncés et représentations - Plan

- **Qu’est-ce qu’un problème ?**
- **Différents types d'énoncés**
  - énoncé de type combinatoire
  - énoncé avec états et opérateurs de changement d'états
    - énoncé avec buts et décomposition de buts
- **Recherche heuristique**

Énoncé avec décomposition du problème

- *Étant donné un but, des opérateurs de décomposition du but en sous-buts, des buts primitifs (triviaux), trouver les opérateurs à appliquer pour décomposer le but initial en un ensemble de sous-buts primitifs*

→ Il s'agit donc de décomposer le problème en sous-problèmes plus simples jusqu'à n'avoir que des problèmes élémentaires

Énoncé avec décomposition du problème

- Formalisation : il faut spécifier
  - les opérateurs de décomposition d'un pb en sous-pb
  - les pb primitifs : pb dont la solution est connue, (résolue ou triviale)
- Exemples : problème de planification, tours de Hanoï, décomposition d'une intégrale, vérification syntaxique…

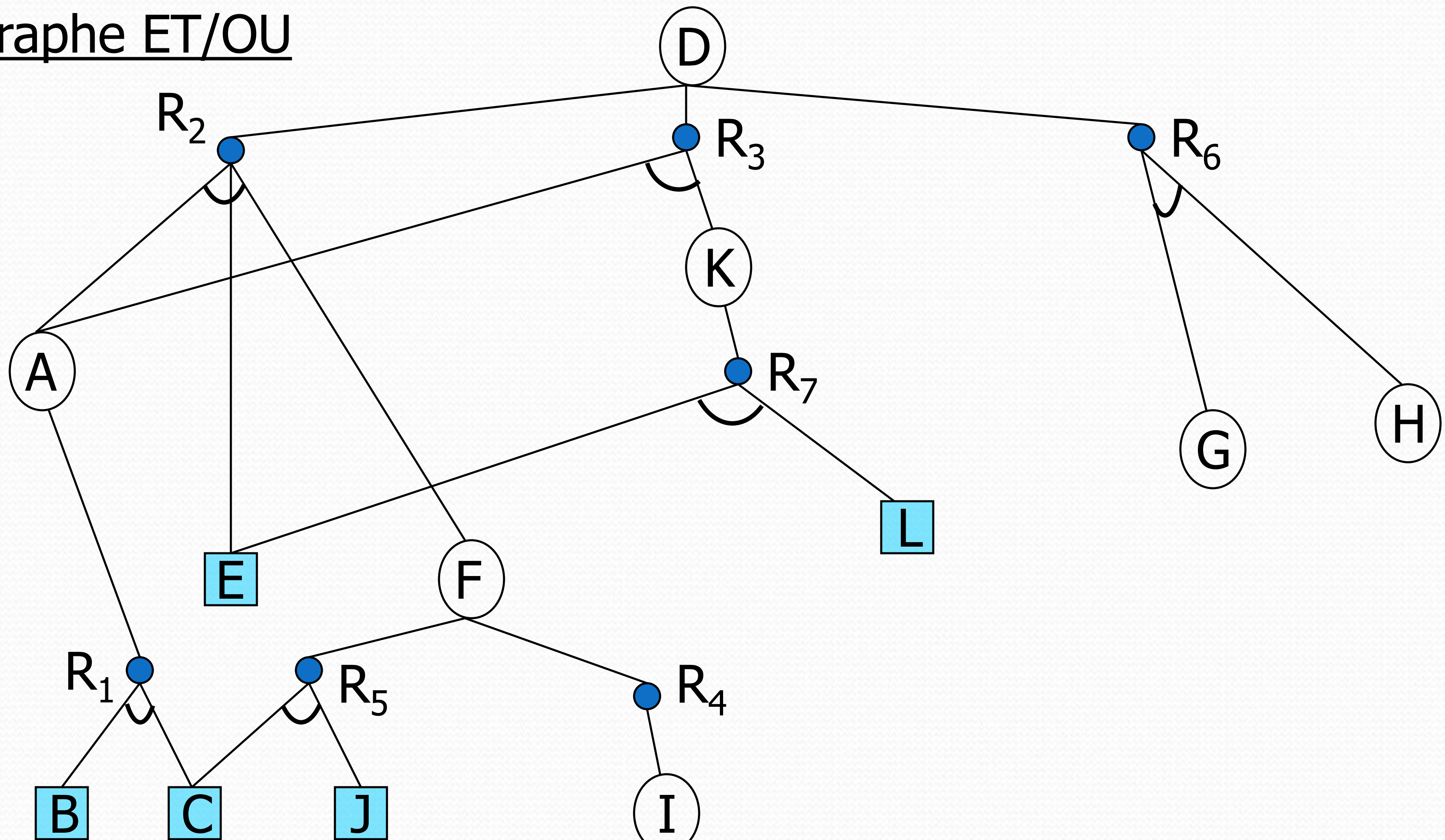
Énoncé avec décomposition du problème - Exemple

- Pb terminaux : B, C, E, J, L
- Règles de décomposition
  - R1 : A -> B,C
  - R2 : D -> A,E,F
  - R3 : D -> A,K
  - R4 : F -> I
  - R5 : F -> C,J
  - R6 : D -> G,H
  - R7 : K -> E,L

- Pb à résoudre : D
- Dessiner le graphe ET/OU de ce pb
- Donner le(s) sous-graphe(s) solution(s)

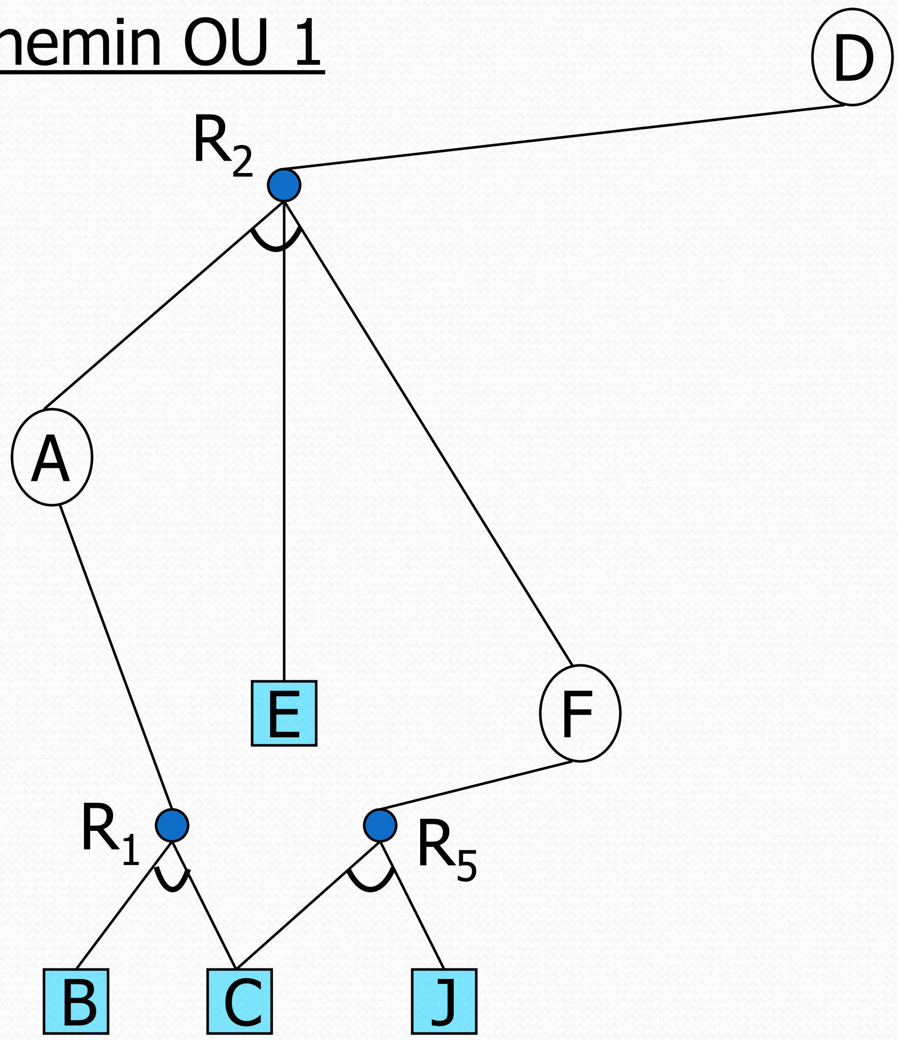
Énoncé avec décomposition du problème - Exemple - solution

Graphe ET/OU



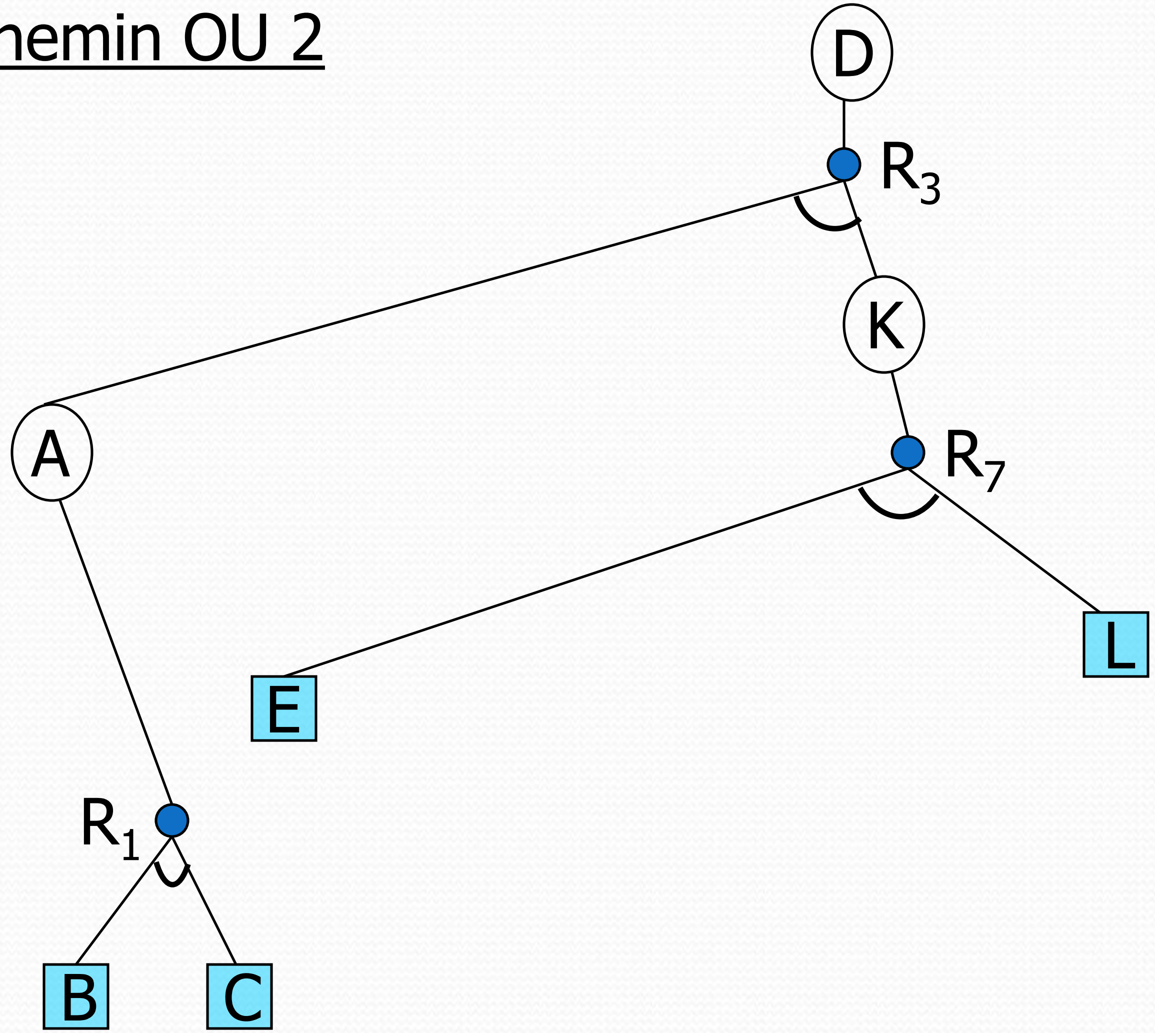
Énoncé avec décomposition du problème - Exemple - exo - sol

Chemin OU 1



Énoncé avec décomposition du problème - Exemple - exo - sol

<u>Chemin OU 2</u>



Énoncé avec décomposition du problème - Graphe ET/OU

- Le graphe de résolution s'appelle graphe ET/OU
  - racine = le problème à résoudre
  - nœuds OU = sous-problèmes
  - arcs issus de OU = opérateurs de décomposition applicables au nœud
  - nœuds ET = règle de décomposition
  - arcs issus de ET = mènent aux différents sous-pb
  - feuilles = nœuds échecs ou pb primitifs

Énoncé avec décomposition du problème - Graphe ET/OU

- Le graphe est dit résolu si
  - c'est une feuille correspondant à un pb primitif
  - sa racine est un OU et qu'un de ses sous-graphes est résolu
  - sa racine est un ET et que tous ses sous-graphes sont résolus
- La solution est un graphe résolu dont on ne retient pour les nœuds OU qu'un successeur ; on parle aussi de chemin-OU

Énoncés et représentations - Plan

- Qu’est-ce qu’un problème ?
- Différents types d'énoncés
- **Recherche heuristique**

Résolution - recherche

Résolution de problème ≈ parcourir un graphe

- Extrêmement coûteux (181 440 états pour le taquin)

⇒ **Construire le graphe au fur et à mesure**

⇒ **Explorer le graphe**

- Recherche aveugle

  recherche en largeur ou en profondeur d'abord

- Recherche informée (ou heuristique) :

  stratégie permettant de choisir le nœud à développer sur des critères propres au problème

Propriétés d'une recherche

- Terminaison : si le pb admet des solutions, un algo complet doit terminer en en fournissant une (s'il n'y a pas de solution, un algo de décision s'arrête et l'indique, un algo de semi-décision ne s'arrête pas
- Complexité : taille du graphe de recherche exploré
- Admissibilité ou optimalité : une recherche est admissible si elle fournit la solution optimale (de coût minimal)

- Quid d'une recherche en profondeur? En largeur?
- A* pour la recherche heuristique

Énoncés et représentations - Plan

- **Qu'est-ce qu'un problème ?**
- **Différents types d'énoncés**
- **Recherche heuristique**
  - Faire les choix les plus contraints
  - Réduire les différences pr au but
  - Utiliser une fonction d'évaluation
  - Stratégie ordonnante vs élaguante

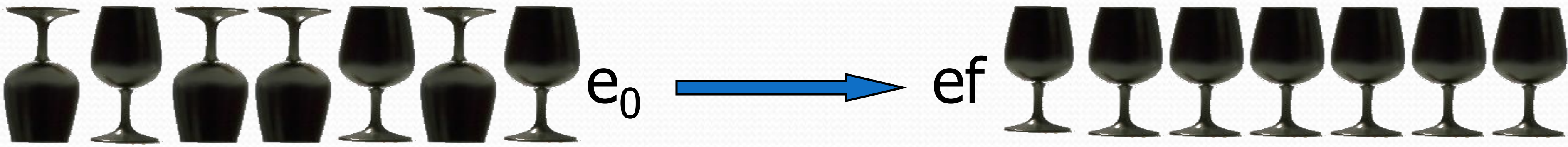
Fonction d'évaluation

Utiliser une fonction numérique d'évaluation des nœuds qui évalue la chance qu'ils ont d'appartenir à la solution

- Nécessite souvent une bonne connaissance du domaine et une longue expérimentation
- Vérifier que le temps de calcul nécessaire à l'heuristique ne soit pas plus long que d'appliquer la stratégie « bête » !

Fonction d'évaluation Exercice

Exemple des verres retournés



- Seule possibilité : retourner simultanément 2 verres adjacents

Fonction d'évaluation Exercice

Exemple des verres retournés

$e_0 \longrightarrow ef$

- Seule possibilité : retourner simultanément 2 verres adjacents
- Dérouler le processus avec la fonction *f1* définie par le nombre de verres mal placés
- Idem avec *f2* = distance entre les deux verres mal placés les plus éloignés

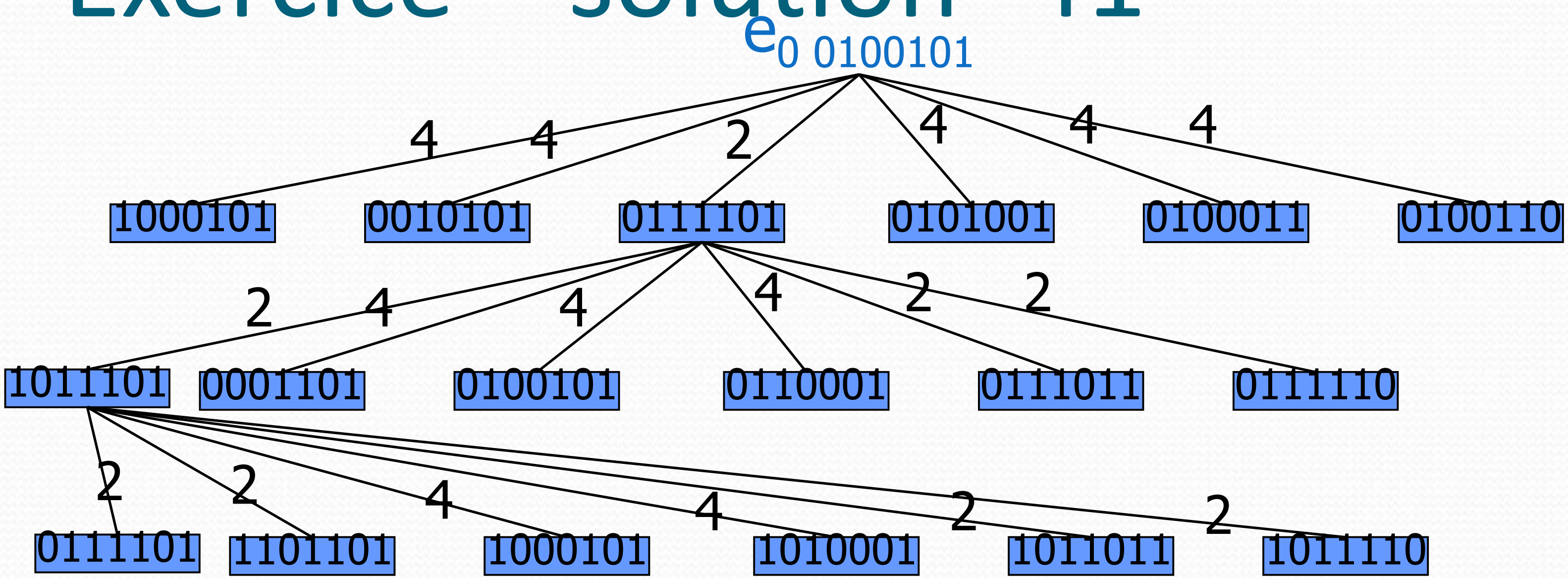
$(f1(e_0) = 5,\ f1(ef) = 0\ ;\ f2(e_0) = 5,\ f2(ef) = 0)$

Fonction d'évaluation Exercice - solution

Exemple des verres retournés

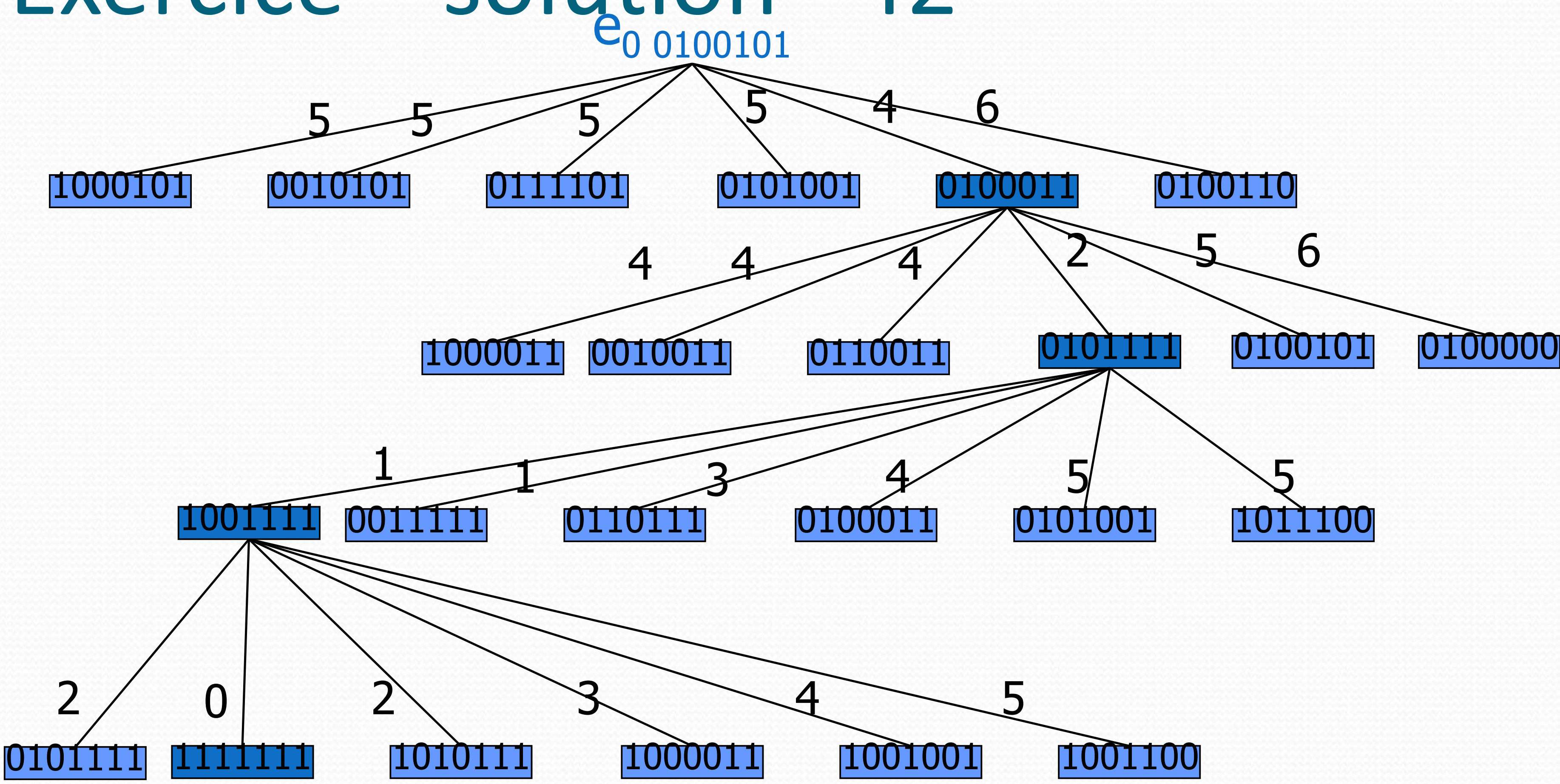
- état = (v_1 v_2 v_3 v_4 v_5 v_6 v_7) avec v_i = 1 si le verre i est bien placé, 0 sinon
- e_o = (0 1 0 0 1 0 1), e_f = (1 1 1 1 1 1 1)
- 6 opérateurs O_i, i dans [1..6], retourne les verres i et i+1
- Cf. graphe (f1 peu informative)

Fonction d'évaluation Exercice – solution - f1



f1 peu informative !

Fonction d'évaluation Exercice – solution – f2



Fonction d'évaluation

Remarque sur l'exemple des verres retournés

- Ce type de fonction est un gradient
  - La fonction évalue la distance d'un état par rapport au but
  - À partir de l'état courant, on choisit l'état qui fait le plus varier la fonction (plus grande pente)

- On risque de tomber dans un optimum local si on ne considère que les états successeurs de l'état courant : on parle alors d'*hill-climbing*
  - sinon « meilleur d'abord »

Énoncés et représentations - Plan

- **Qu’est-ce qu’un problème ?**
- **Différents types d'énoncés**
- **Recherche heuristique**
  - Faire les choix les plus contraints
  - Réduire les différences pr au but
  - Utiliser une fonction d’évaluation
    - Ordonnancement vs élagage

Ordonnancement versus élagage

- Les stratégies précédentes ne contrôlent que l'ordre des nœuds explorés
- D'autres stratégies permettent d'élaguer (*i.e.* écartent définitivement) certaines alternatives
  - Risque de non-terminaison ou de perte de la solution optimale