

# Mini-projet d'algorithmique avancée

Kévin VYTHELINGUM, Jean-Michel NOKAYA

10 décembre 2013

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Préliminaires</b>	<b>2</b>
<b>3</b>	<b>Méthodes des essais successifs</b>	<b>2</b>
3.1	Analyse . . . . .	2
3.2	Algorithme . . . . .	3

## 1 Introduction

## 2 Préliminaires

1.

$$\begin{aligned} d_{max} &\Rightarrow c_{min} \\ \sum_{i=1}^k d_{max} &\Rightarrow \sum_{i=1}^n c_{min} \\ D &\Rightarrow C_{opt} \end{aligned}$$

2. complexité =  $k^n$  (On peut faire un arbre pour le démontrer)

## 3 Méthodes des essais successifs

### 3.1 Analyse

Solution : un candidat est un vecteur de taille  $n$  où chaque coefficient est une durée choisie parmi l'ensemble  $\{1, \dots, k\}$  (à chaque tâche on associe une durée). On choisit d'enregistrer les choix réalisés dans un tableau  $T$  de taille  $n$ .

$S_i$  : l'ensemble des durées possibles de 1 à  $k$

$$\text{satisfaisant}(x_i) = \sum_{j=1}^i x_j \leq D$$

(la somme partielle des durées choisies est inférieure à la durée maximale autorisée)

$$\text{enregistrer}(x_i) = T[i] \leftarrow x_i$$

$$\text{soltrouvee} : i = n$$

$$\text{defaire}(x_i) = T[i] \leftarrow 0$$

Pour simplifier les vérifications au niveau de satisfaisant et des conditions d'élagage, on utilisera les variables entières *cout* et *duree* initialisée à 0, qui représenteront le cout courant et la durée courante due à nos choix de durées. Elles seront mises à jour dans *enregistrer* et dans *défaire*. En effet, en notant  $CD$  le tableau à deux dimensions ayant les coûts en ligne et les durées en colonne, on effectuera dans *enregistrer* :

$$\begin{aligned}\text{coût} &\leftarrow \text{coût} + CD[i][x_i] \\ \text{durée} &\leftarrow \text{durée} + x_i\end{aligned}$$

Ensuite on effectura dans *défaire* :

$$\begin{aligned}\text{coût} &\leftarrow \text{coût} - CD[i][x_i] \\ \text{durée} &\leftarrow \text{durée} - x_i\end{aligned}$$

### 3.2 Algorithme

Procédure *ordonnancement<sub>simple</sub>(enti)* ; var ent k,  $x_i$ , D ; début  $S_i = [1..k]$  ;  
 pour  $x_i$  de 1 à k faire si  $\sum_1^i x_l \leq D$  alors  $T[i] < -x_i$  ;  $\text{cout} < -\text{cout} + cd[i][x_i]$  ;  
 $\text{duree} < -\text{duree} + x_i$  ;  
 si  $i = n$  alors *Affiche<sub>sol</sub>*() ; sinon *ordonnancement<sub>simple</sub>(i + 1)* ; fsi ;  
 $T[i] < 0$  ;  $\text{cout} < -\text{cout} - cd[i][x_i]$  ;  $\text{duree} < -\text{duree} - x_i$  ;  
 fsi ; fait ; fin ;  
 Appel : *ordonnancement<sub>simple</sub>(1)* ;