

2 P-RAM : Rupture de symétrie déterministe

On veut concevoir un algorithme EREW qui permet de sélectionner « beaucoup » d'objets dans une liste chaînée sans jamais choisir deux objets adjacents dans la liste. Chaque objet est associé à un processeur mais le numéro du processeur n'est pas relié à l'ordre des éléments dans la liste.

▷ **Question 4** *Concevoir un algorithme qui permet de sélectionner un nombre optimal d'objets en temps $O(\log n)$ sur une EREW.*

Dans la suite on va montrer qu'il est possible d'extraire « beaucoup » d'éléments en un temps $O(\log^* n)$, où

$$\log^* n = \min\{i \mid \log^i n \leq 1\}$$

Dans l'expression précédente, \log^i représente la composition de i fois la fonction \log . La fonction \log^* a une croissance très lente, puisque $\log^*(2^{65536}) = 5$.

Nous allons maintenant préciser le sens de « beaucoup » à partir de la notion d'ensemble indépendant maximal.

Définition 1. Un ensemble de sommets V' d'un graphe $G = (V, E)$ est indépendant ssi

$$\forall (a, b) \in E, \text{ au plus un élément de } \{a, b\} \text{ est dans } V'$$

Un ensemble de sommets V' d'un graphe $G = (V, E)$ est indépendant et maximal ssi l'ajout de tout sommet à V' en fait un ensemble non indépendant.

Dans ce qui suit, notre objectif est d'arriver à extraire un ensemble indépendant maximal de la liste en temps $O(\log^* n)$. Pour cela, on va commencer par colorier la liste (avec 6 couleurs). On va construire un algorithme qui part d'une n -coloration (où la couleur de chaque objet est déterminée par la couleur du processeur qui lui est associé) et qui diminue à chaque étape le nombre de couleurs utilisées.

▷ **Question 5** *Donner un algorithme astucieux pour diminuer le nombre de couleurs utilisées tout en gardant une coloration (on pourra raisonner sur le codage binaire des couleurs).*

▷ **Question 6** *Pourquoi obtient-on 6 couleurs ? Montrer qu'on obtient 6 couleurs après $O(\log^* n)$ étapes.*

3 Ordonnancement

3.1 Ordonnancement sans communication

On veut ordonnancer un ensemble de n tâches indépendantes T_1, T_2, \dots, T_n . La durée de T_i est p_i . On suppose que l'ordonnancement débute au temps $t = 0$, et on note C_i la date de la fin de l'exécution de la tâche T_i .

▷ **Question 7** On dispose d'un seul processeur dans cette question.

1. Quel est l'ordre d'exécution optimal si l'objectif est de minimiser la somme des dates de fin $\sum_{i=1}^n C_i$?
2. Quel est l'ordre d'exécution optimal si l'objectif est de minimiser la somme pondérée des dates de fin, i.e. $\sum_{i=1}^n w_i.C_i$, où w_i est un poids positif associé à la tâche T_i ?

▷ **Question 8** On dispose de $p \geq 2$ processeurs identiques dans cette question. Quel est l'ordre d'exécution optimal si l'objectif est de minimiser la somme des dates de fin $\sum_{i=1}^n C_i$?

3.2 Comparaison d'heuristiques

On considère le modèle avec coût de communication du cours. Pour le graphe de tâches de la Figure 4, les poids des tâches sont indiqués entre parenthèses, et ceux des arêtes le long de celles-ci.

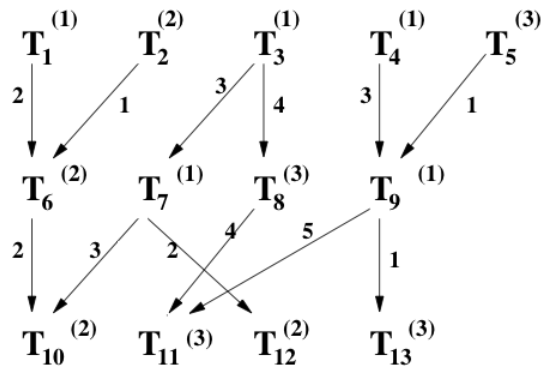


FIG. 4 – Graphe de tâches à ordonnancer.

▷ **Question 9** On suppose disposer d'un nombre illimité de processeurs. Ordonnancer le graphe avec l'heuristique du chemin critique modifié, l'heuristique de Kim et Browne, puis l'heuristique de Sarkar.

▷ **Question 10** On suppose maintenant disposer de seulement $p = 3$ processeurs. Quel est l'ordonnancement optimal ?