CentraleSupélec

Algorithmique avancée

Épreuve pour le bloc 2 : algorithmes *online*

Le concours consiste à concevoir et à implémenter des algorithmes *online* pour un problème d'ordonnancement que nous appellerons *Slow-Fast Scheduling* et calculer expérimentalement son ratio de compétitivité [1].

Notre problème est défini comme suit :

Instance :

- m machines exécutants des tâches, dont
  - m_f rapides et identiques, leur vitesse $s = 1$,
  - $m_s = m - m_f$ lentes et identiques, leur vitesse $s \in (0, 1)$,
- une séquence $\boldsymbol{\sigma}$ de tâches définies par leur durée, σ_i, $i = 0, 1, \dots n$.

Objectif : Attribuer les tâches de $\boldsymbol{\sigma}$ aux machines en **minimisant le *makespan*** (c.-à-d. la machine la plus occupée est prise le moins longtemps).

Bien que la version décisionnelle du problème soit *NP*-complète, il est possible de calculer une solution optimale pour des instances de taille "raisonnable". Grâce à la valeur de solution exacte qui vous est fournie dans le fichier d'une instance, vous jugerez la qualité de votre algorithme *online* en produisant le ratio de compétitivité empirique pour une instance donnée.

Le critère du classement est la moyenne des ratios de compétitivité empiriques calculés sur l'ensemble des instances fourni à chaque étape.

Lors de la phase préparatoire, vous pouvez observer le comportement de vos algorithmes déterministes et randomisés, le résultat de ceux derniers produit pour chaque instance est la moyenne de plusieurs lancements afin de s'affranchir du hasard. Pour indiquer le type de votre algorithme, adaptez la valeur `True/False` de la variable `response` (`True` par défaut) dans la fonction `mon_algo_est_deterministe()` dans `MinSetCover_online.py`.

Pour la phase de compétition, vous choisissez celui de vos algorithmes qui vous paraît le plus prometteur.

1. Nous nous permettons ici un abus de langage ; ≪ notre ≫ ratio n'est pas conforme avec la définition de cette mesure, calculée sur toutes les instances.