*Mathilde JAY et Denis TRYSTRAM*

**Algorithmique Avancée**

*ENSIMAG 2A – Alternants 2021*

Allocation de ressources dans un data-centre

# 1 Vue globale (Aspects fonctionnels)

Le projet que l'on se propose de réaliser ici concerne **la gestion efficace des ressources de calcul dans un *Data Center* en mettant un focus sur la consommation énergétique**. Le projet est étalé sur 5 séances où les problèmes seront abordés progressivement (avec 3 étapes).

1. vendredi 24 septembre, prise en main du problème, réalisation d'une première version avec validation sur des tests sur des jeux d'essai donnés.

2. jeudi 7 et vendredi 8 octobre, amélioration(s) par plusieurs heuristiques (algorithmes). Comparaisons et analyses (théoriques et expérimentales).

3. jeudi 28 et vendredi 29 octobre, prise en compte des aspects énergétiques. Etudes des compromis.

Pour chacune de ces 3 grandes étapes, il sera demandé aux groupes d'étudiants d'écrire un rapport synthétique d'une ou deux pages maximum qui présentera la démarche, les *résultats* obtenus, la répartition du travail, les difficultés rencontrées avec une annexe avec le code produit et les expérimentations.

**Ce rapport est à rendre pour le vendredi soir minuit** (dépot électronique).

# 2 Présentation du problème

Un data-centre typique (bien entendu, simplifié pour les besoins d'une étude en temps limité) est composé de $m$ ordinateurs (multi-coeurs que l'on supposera identiques, ayant chacun $q$ coeurs) et reliés par un réseau d'interconnexion local et rapide. On négligera les temps de communications ou toute autre considération sur la capacité mémoire des jobs.

$m$ est grand, mais il y a un temps et un coût non négligeable pour *ouvrir* un nouveau serveur.

Les jobs soumis sont des applications séquentielles dont on connait les temps d'exécution, notés $p_j$ pour le job $j$. En pratique, on n'a qu'une estimation de ces temps. Ces jobs sont placés dans une queue. Le problème est d'exécuter les jobs de la queue le plus vite possible.

Le data-centre est opéré par un *cloud provider* dont l'objectif est de faire un maximum de profit. Le provider cherche donc à remplir au maximum les machines de telle sorte que le nombre de machines en service à un instant donné soit minimum. Chaque machine a un coût de mise en service $S$.

# 3 Travaux demandés à la première étape : Brute force

On vise un objectif de performances, mesuré par la date où se termine l'exécution du dernier des jobs de la queue.

On demande

1. de **concevoir un algorithme qui génère toutes les configurations possibles**.

   La sortie de l'algorithme donnera la liste (et la séquence) des jobs exécutés sur chaque coeur.

2. Programmer cet algorithme.

3. Conduire une campagne de tests.

   Les fichiers de tests sont fournis pour une étude expérimentale.

Les groupes peuvent choisir leur langage de programmation favori.

# 4 Travaux demandés à la seconde étape : Résolution approchée

### Heuristiques et approximation (guarantie)

Prenant le même contexte que la séance précédente, on demande :

1. de **concevoir un algorithme qui génère une solution réalisable**.

2. Programmer cet algorithme.

3. Si besoin, calculer une approximation de votre solution.

4. Conduire une campagne de tests, soit pour confronter le rapport d'approximation (pire cas) sur les instances, soit valider l'heuristique.

5. Quel est le temps d'exécution de votre heuristique/algorithme ?