Big Data : Informatique pour les données et calculs massifs

2A – campus de Metz – 2018-2019

TD1 : Passage à l'échelle et algorithmique *Map-Reduce*

Exercice 1 : Analyse des mesures de performances d'une application distribuée

La figure 1 montre des allures de temps d'exécutions d'une application distribuée, sur 3 clusters différents (cluster a (Ca), cluster b (Cb) et cluster c (Cc)), et pour 3 tailles successives de problèmes (N1, N2 et N3). Les échelles en abscisses et ordonnées sont logarithmiques.

Les 3 clusters ont des caractéristiques différentes en termes de puissance de calcul des nœuds et de capacité de communication des réseaux d'interconnexion. L'application utilise le même algorithme pour les trois tailles de données (avec N1 < N2 < N3).

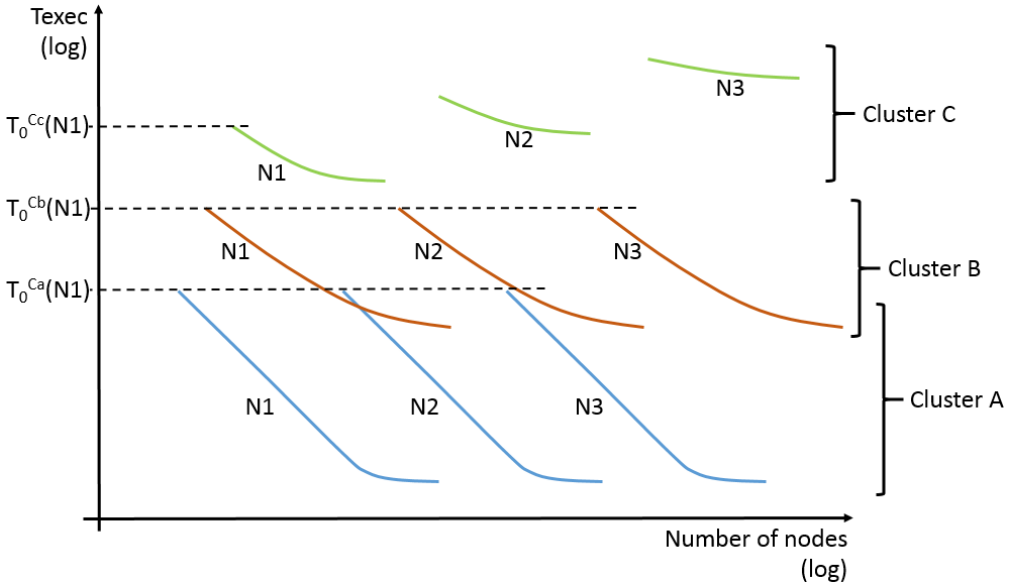

Figure 1 : temps d'exécution sur 3 clusters différents, et pour 3 tailles de problème

Question : Que pouvez-vous conclure de la capacité de passage à l'échelle de cette implantation distribuée ? et des caractéristiques relatives des 3 clusters ?

Exercice 2 : Dimensionnement d'un nombre de ressources lors d'un passage à l'échelle

On considère un algorithme d'analyse de données mono-nœud et optimisé, mais limité à la mémoire et à la puissance de calcul d'une seule machine. Afin de traiter des données plus volumineuses on distribue cette application. Cependant l'adaptation de l'algorithme optimisé mono-nœud est profonde et entraîne un surcoût significatif de temps d'exécution sur 2 nœuds, avant d'entamer une décroissance régulière, comme illustré sur la figure 2.

On suppose que l'on mesure le temps d'exécution sur 2 nœuds, et on note : $T(N,2) = k.T_{1\text{-node}}(N)$

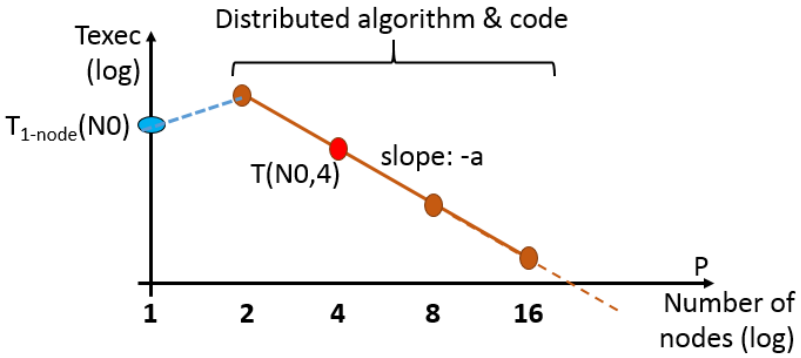

Figure 2 : Passage à une version distribuée avec surcoût initial sur 2 nœuds

Question 1 : Modélisez le temps d'exécution T(N0,P) de la figure 2, pour P ≥ 2.

Question 2 : On suppose que les calculs dominants des deux versions de l'algorithme sont en $O(N^{3/2})$. On s'intéresse maintenant à un problème ayant deux fois plus de données : N1 = 2×N0. On souhaite conserver le temps de traitement obtenu avec N0 données sur P_s = 4 nœuds ($T(N1,P_x) = T(N0,P_s)$). Calculez le nombre de nœuds nécessaires (P_x) avec les deux hypothèses suivantes sur k = k(N) :

- $k(N) = k_0 = Cte$
- $k(N) = k_0 \times (N/N0)$

Faites des applications numériques avec : a = 1 (pente idéale, décroissance parfaite du temps d'exécution).

Question 3 : Tracez les deux nouvelles courbes de temps d'exécution pour P ≥ 2 et pour N = N1, correspondant aux deux hypothèses sur k(N).

Exercice 3 : Analyse statistique de la longueur des mots d'un texte en *Map-Reduce*

On souhaite établir des statistiques sur la longueur des mots dans un document, ou un ensemble de documents, à partir d'une application *Map-Reduce*. Proposez un algorithme dans le paradigme *Map-Reduce* pour les 3 analyses suivantes :

Question 1 : Compter le nombre de mots de chaque longueur présente dans le texte (en vue d'établir un histogramme des longueurs de mots).

Question 2 : Compter le nombre de mots de 1 à 5 caractères (inclus), de 6 à 10 caractères (inclus), de 11 à 15 caractères (inclus) et de plus de 15 caractères présents dans le texte.

Rmq : si on génère plusieurs fichiers de sorties, il est nécessaire que l'ordre des noms de fichiers soit celui des sous-ensembles de longueurs de mots.

Question 3 : Obtenir les listes de mots de 1 à 5 caractères (inclus), de 6 à 10 caractères (inclus), de 11 à 15 caractères (inclus) et de plus de 15 caractères présents dans le texte. Il n'est pas demandé de trier les mots à l'intérieur d'une liste, ni d'éliminer les doublons.

Dans tous les cas on décrira les paires clé-valeur utilisées à chaque étape de la solution *Map-Reduce*, et on essaiera d'optimiser ces solutions. Les actions des différentes tâches seront décrites en pseudo-code de haut niveau.