Parallel Computing for Data Science
Lab x003 : Embarransingly parallel

Jairo Cugliari

S1 2019–2020

1 Prise en main

Utilisez la fonction `clusterCall` du package `parallel` pour obtenir 30 réalisations d'une variable aléatoire en utilisant 2 cores de calcul. Pour ce faire:
- créez un cluster de 2 processeurs
- appellez à la fonction `clusterCall`
- fermez le cluster.

Quelle est la nature de l'objet obtenu? Vous pouvez le convertir en vecteur à l'aide de la fonction `unlist`. Répétez l'exercice maintenant en utiisant `foreach`.

2 Calcul de la somme en parallel

1. Simulez un vecteur de données de taille 1e5 et obtenez sa somme.
2. Décopez le vecteur en 20 sous vecteurs de même taille à l'aide de la fonction `split`.
3. Utilisez la fonction `lapply` pour obtenir les sommes partialles de chaque slot de la liste de sous vecteurs, puis obtenez la somme.
4. Répéter le point 3 maintenant en utilisant du calcul en parallèle.
5. Comparez les résultats et chronometrez les temps d'exécution.

3 Parallélisme embarrassant

On souhaite estimer la qualité de prédiction d'un modèle linéaire de la largeur d'une pétale sur sa longueur sur le jeu de données `iris`.

La technique de *leave-one-out* consiste estimer l'erreur de généralisation de la manière suivante: on estime le modèle avec tous les individus sauf un, on fait une prédiction pour cet individu, et on regarde l'erreur quadratique entre la prédiction et la valeur connue; puis on répète l'opération pour chacun des individus et on somme les erreurs obtenues.

1. Charger le jeu de données `iris`.
2. Écrire la fonction `leave.one.out(i)` qui renvoie l'erreur de prévision de la i-ème observation à partir du modèle entraîné sur les individus restantes.
3. Estimer l'erreur de généralisation par une boucle `for`.
4. Idem avec la fonction `lapply`.
5. Remplacer `lapply` par la version parallèle du paquet `parallel`.
6. Utiliser le paquet `foreach` pour estimer l'erreur de généralisation de manière séquentielle.
7. Utiliser les paquets `foreach` et `doParallel` pour estimer l'erreur de généralisation en parallèle.

Pour les différents approches, chronométrez le temps d'exécutions et comparez.

4 Application : clustering en parallèle

Utilisez les données `Boston` du paquet `MASS` pour réaliser une classification non supervisée par k-means avec 2000 initialisations aléatoires.

Évaluez le gain de passer en parallèle les initialisations dans les contextes de `parallel` d'une part et de `foreach` d'autre part.om

5 Application : Bagging CARTs in parallel

Utiliser le jeu de données `diamonds` du paquet `ggplot2` pour entraîner un classifier de type CART avec bagging. Paralléliser l'étape de construction selon les deux paradigmes vus précédemment.

6 Application : Random Forest in parallel

Utiliser le jeu de données `diamonds` du paquet `ggplot2` pour entraîner un classifier de type forêt aléatoire. Paralléliser l'étape de construction selon les deux paradigmes vus précédemment.