Google Trace

Big Data Frameworks

Mention Mathématiques et Informatique
Parcours HPDA

23/02/2025

Rédigé par :
BERDOYES Florent
florent.berdoyes@cy-tech.fr
CLEMENCEAU Maxime
maxime.clemenceau@cy-tech.fr
HONOR Julien
julien.honor@cy-tech.fr

PAULY Alexandre
alexandre.pauly@cy-tech.fr
SABADIE Laura
laura.sabadie@cy-tech.fr

Contents

1 Introduction

L'analyse des traces d'utilisation des clusters informatiques est un enjeu récurrent pour l'optimisation des infrastructures de calcul à grande échelle, surtout dans le cas de Google. L'exploitation efficace des ressources telles que le CPU et la mémoire vive est essentielle pour garantir des performances optimales tout en réduisant les coûts énergétiques. Dans ce contexte, Google a rendu publiques des traces de son infrastructure de cloud computing, permettant d'étudier en détail l'usage des ressources par les différents jobs et tasks exécutés sur un cluster de production. Ces données massives, collectées sur une période d'un mois, offrent une opportunité unique d'analyser le comportement des charges de travail dans un environnement de calcul distribué.

Récemment, l'essor des techniques d'analyse de données à grande échelle a permis d'extraire des insights précieux à partir de ces traces, facilitant ainsi l'identification des jobs les plus consommateurs en CPU et mémoire. Cette étude repose sur des outils de Big Data tels que Pandas et Spark, qui permettent de traiter et d'agréger efficacement ces volumes de données massifs. Toutefois, malgré ces avancées, l'identification et la classification des jobs dominants restent des défis majeurs, nécessitant des approches adaptées à la volumétrie et à la complexité des traces de Google.

Dans ce projet, nous nous attachons à caractériser les jobs et tasks les plus consommateurs en CPU et mémoire, en nous appuyant sur des métriques telles que les valeurs maximales et moyennes de consommation. Ces analyses permettront d'identifier les jobs dominants, afin d'optimiser la gestion des ressources et d'améliorer la performance des infrastructures de cloud computing.

2 Jeu de données

Le jeu de données [1] utilisé dans ce projet est issu des traces de Google Cluster, une collection de données publiques publiée par Google, qui documente l'utilisation des ressources d'un cluster de calcul distribué sur une période d'un mois. Ces traces fournissent des informations détaillées sur l'exécution des tâches (tasks) et des travaux (jobs), incluant l'utilisation du CPU, de la mémoire vive, ainsi que divers autres indicateurs de performance.

Structure des données

Les traces sont organisées sous forme de plusieurs fichiers compressés (.csv.gz), répartis dans différents dossiers en fonction de la nature des événements enregistrés :

- job_events : Contient des informations sur la soumission, l'annulation et l'échec des jobs.
- task_events : Retrace les modifications d'état des tâches associées aux jobs.
- task_usage : Fournit des mesures précises sur la consommation de CPU et de mémoire vive pour chaque tâche.
- machine_events : Enregistre les changements d'état des machines du cluster.
- machine_attributes : Contient des caractéristiques des machines *(ex : capacité CPU et mémoire)*.
- task_constraints : Définit les contraintes d'affectation des tâches aux machines.

Données essentielles pour notre analyse :

Dans le cadre de notre étude, nous nous concentrons principalement sur le fichier task_usage, qui contient les indicateurs clés d'utilisation des ressources, notamment :

- job ID : Identifiant unique du job.
- task index : Identifiant de la tâche au sein d'un job.
- maximum CPU rate : Taux maximal d'utilisation du CPU par la tâche (normalisé entre 0 et 1).
- maximum memory usage : Quantité maximale de mémoire utilisée par la tâche.

Ces données sont agrégées par job et task afin d'identifier les jobs les plus consommateurs en CPU et en mémoire, aussi bien en valeurs maximales qu'en valeurs moyennes. L'analyse de ces métriques permet d'optimiser la gestion des ressources dans un environnement de calcul distribué et d'améliorer l'efficacité globale du cluster.

3 Modélisation

3.1 Compréhension du domaine

3.1.1 Schéma conceptuel

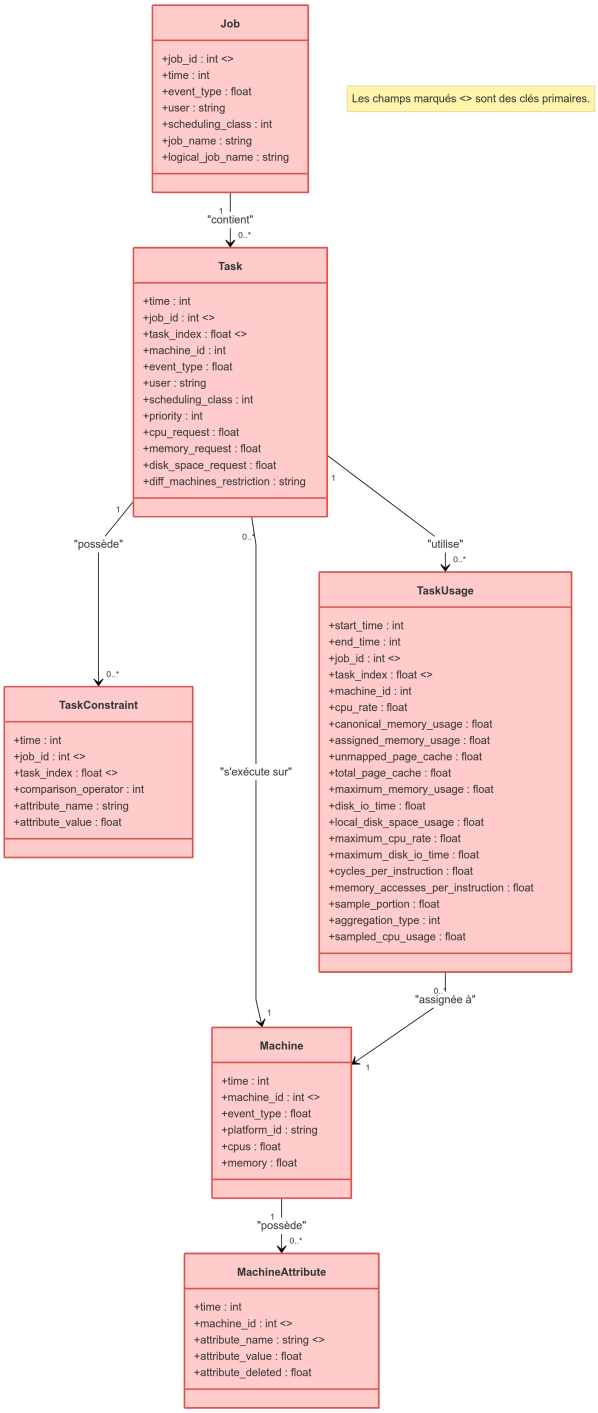

Figure 1: Schéma conceptuel

La figure ci-dessus illustre notre proposition de schéma conceptuel, qui nous a permis de mieux comprendre l'interaction entre les différentes tables présentes dans la trace de Google. Il est important de noter qu'il s'agit d'une estimation et non du schéma officiel fourni par Google. Nous l'avons construit à partir des informations publiques disponibles et de notre propre analyse des fichiers de traces, dans le but d'éclairer les relations logiques entre les entités principales (jobs, tasks, machines, etc.).

3.1.2 Choix de l'outil Apache Spark

Pour gérer la volumétrie conséquente des traces Google, nous avons opté pour Apache Spark dès les premières étapes du traitement. Ce choix a été motivé par la capacité de Spark à manipuler de grandes quantités de données de manière distribuée, garantissant ainsi une lecture et un traitement efficaces, même avec des fichiers compressés de type CSV.gz.

3.2 Sélection des données

Dans notre notebook, nous avons tout d'abord chargé le fichier contenant les mesures d'utilisation des ressources (`task_usage.csv.gz`) en utilisant PySpark. L'inférence automatique du schéma nous a permis de détecter précisément les types de données, facilitant ainsi leur manipulation ultérieure. Ce fichier a été choisi car il regroupe les indicateurs essentiels (CPU et mémoire) pour chaque tâche, offrant les informations indispensables pour l'analyse au niveau des jobs et tasks.

3.3 Préparation des données

La préparation des données a constitué une étape cruciale pour garantir la qualité de l'analyse. Dans un premier temps, nous avons procédé à un nettoyage rigoureux en éliminant les enregistrements comportant des valeurs nulles dans des colonnes critiques telles que `job_id`, `task_index`, `max_cpu_usage` et `max_memory_usage`.

Par la suite, nous avons converti les colonnes numériques relatives à la consommation du CPU et de la mémoire en types adaptés (par exemple, en `float`), afin d'assurer la précision des calculs statistiques. Ces opérations ont permis de transformer un jeu de données brut en une base homogène et fiable, prête à être exploitée pour des analyses plus poussées. Enfin, pour optimiser les performances lors des traitements futurs, le résultat du prétraitement a été sauvegardé au format Parquet, offrant ainsi une compression efficace et une exécution optimisée des requêtes analytiques.

3.4 Recherche des jobs/tasks les plus pertinents

L'étape finale a consisté à identifier les jobs et tasks dominants en termes de consommation de ressources. Pour ce faire, nous avons agrégé les données par `job_id` et calculé, pour chaque job, des indicateurs clés tels que la moyenne et la valeur maximale

d'utilisation du CPU et de la mémoire.

Afin de valider et d'affiner notre approche, nous avons démarré l'analyse sur de petits échantillons, en traitant d'abord 10 enregistrements à la fois, puis en augmentant progressivement la taille de l'échantillon à 50 et enfin à 150 enregistrements. À chaque étape, nous avons enregistré les résultats afin de vérifier la cohérence des agrégations et d'ajuster nos paramètres si nécessaire. Cette démarche itérative nous a permis d'isoler rapidement les jobs présentant des pics de consommation significatifs, tout en différenciant ceux qui affichent une utilisation ponctuelle élevée de ceux qui consomment de manière constante. Ces indicateurs jouent un rôle central dans l'optimisation des ressources, car ils orientent les analyses vers les charges de travail les plus critiques au sein du cluster.

4 Résultats et Discussion

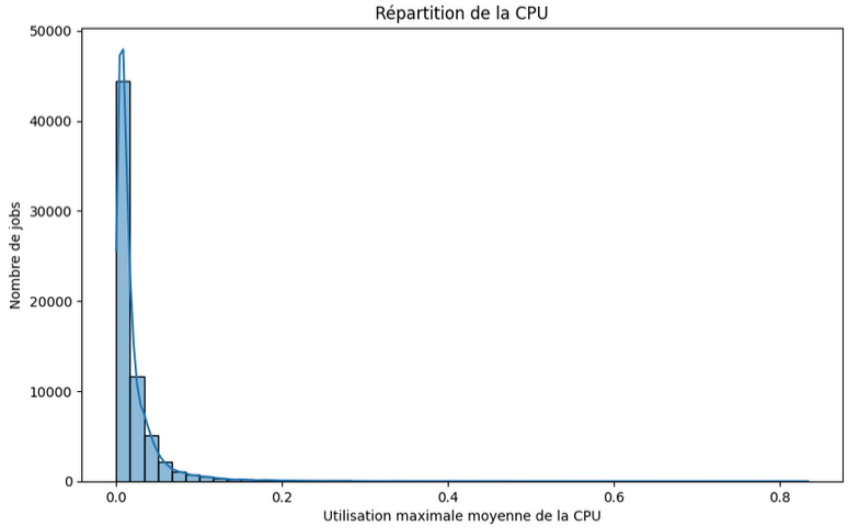

Figure 2: Répartition de l'utilisation maximale moyenne de la CPU pour l'échantillon de jobs (51 à 100).

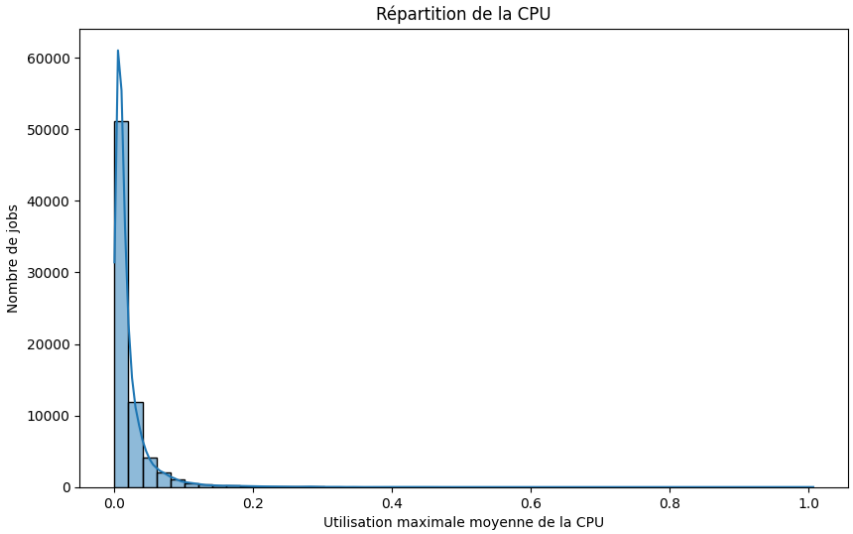

Figure 3: Répartition de l'utilisation maximale moyenne de la CPU pour l'échantillon de jobs (151 à 200).

Commentaire et interprétation :

Les deux graphiques ci-dessus (Figures 2 et 3) représentent la répartition de l'utilisation maximale moyenne de la CPU (axe des abscisses) en fonction du nombre de jobs (axe des ordonnées) pour deux sous-ensembles de données différents. On constate dans chaque histogramme que la majorité des jobs se concentre autour de faibles valeurs d'utilisation

CPU (proches de 0), tandis qu'une queue (ou *long tail*) s'étend vers des valeurs plus élevées, pouvant aller jusqu'à 0,8 ou 1,0.

Cette répartition fortement asymétrique indique qu'une large part des tâches consomme peu de CPU, alors qu'un petit nombre de jobs présente une utilisation très élevée. Ce phénomène est typique dans les environnements de calcul distribué, où la plupart des tâches sont légères, et un nombre restreint de jobs "dominants" monopolise une part importante des ressources.

La différence de forme entre les deux histogrammes (par exemple, la hauteur du pic principal ou l'étendue de la queue) suggère que la proportion de jobs très gourmands varie selon l'échantillon. De plus, le volume total de jobs analysés dans chaque sous-ensemble n'est pas identique, ce qui peut influer sur la distribution observée.

- **Optimisation des ressources :** Identifier les jobs se situant dans la partie haute de la distribution (vers 0,8–1,0) est essentiel pour mieux planifier l'allocation CPU et anticiper les goulots d'étranglement.
- **Hétérogénéité des charges de travail :** La concentration d'une grande partie des jobs autour de faibles valeurs de CPU reflète la diversité des tâches dans un cluster : beaucoup de tâches ont un impact faible, tandis que quelques-unes ont un impact important.
- **Priorisation :** Les résultats invitent à approfondir l'étude des jobs "hors norme" pour comprendre pourquoi leur consommation CPU est si élevée et, le cas échéant, optimiser leur exécution.

Ces histogrammes mettent donc en évidence une répartition dite *long tail*, dans laquelle la majorité des tâches reste peu consommatrice, tandis qu'un nombre restreint de jobs domine l'utilisation de la CPU. Cette observation est cruciale pour la suite de l'analyse, car elle oriente les efforts d'optimisation vers ces "jobs dominants" susceptibles d'améliorer significativement les performances globales du cluster.

Notez que cette analyse ne porte que sur 150 dossiers sur un total de 500, ce qui implique que les conclusions présentées sont basées sur un échantillon partiel des données et devront être validées par une étude de l'ensemble des dossiers pour une vision complète.

5 Références

References

[1] Wilkes, More Google cluster data, 2011, (Google research blog)., 2011. Posted at http://googleresearch.blogspot.com/2011/11/more-google-cluster-data.html.

[2] J. Wilkes and C. Reiss, ClusterData2011_2 traces, 2011, https://github.com/google/cluster-data/blob/master/ClusterData2011_2.md.

[3] J. H. C. Reiss, J. Wilkes, Google cluster-usage traces: format + schema, Technical Report, Google Inc., Mountain View, CA, USA, 2011 . Revised 2012.03.20. Posted at https://drive.google.com/open?id=0B5g07T_gRDg9Z0lsSTEtTWtpOW8& authuser=0.

[4] Google cluster data - discussions, https://groups.google.com/forum/!forum/google clusterdata-discuss.

[5] Google cluster traces bibliography, https://github.com/google/cluster-data/blob/master/bibliography.bib.