Master 1 - Systèmes Distribués

Année 2023-2024

Épreuve pratique Systèmes Distribués • à rendre avant le 15 janvier 2024

Cadre général du projet

L'exercice que vous devez rendre a pour but d'appliquer les principes des systèmes distribués pour réaliser un programme correspondant à un cas particulier.

Contraintes pour la réalisation

L'exercice est à réaliser par groupe de un à deux étudiants. La programmation peut être réalisée en Java, C/C++ ou Scala. Il n'est pas obligatoire de développer une interface graphique (bien au contraire). Il est très fortement conseillé de respecter les conventions d'écriture du langage utilisé. La lisibilité du code sera évaluée.

Vous devrez remettre un rapport au format PDF, le code source et expliquer comment le compiler et l'exécuter. Pour le document, les consignes **strictes** de présentation sont : au moins 12 pages, police 12pts maximum, marges 2cm maximum. Le rapport doit comporter au moins les éléments suivants :

- **une analyse** du sujet, précisant, entre autre, les structures de données et les algorithmes, le paradigme retenu (sa justification), une décomposition en sous-problèmes éventuels ;
- **la spécification des classes principales ou des fonctions** ;
- **l'architecture logicielle détaillée** de votre application ;
- **une évaluation / comparaison des performances** ;
- une documentation pour compiler et exécuter votre code ;
- **un jeu de tests** montrant que votre programme fonctionne.

Il est fortement conseillé de rédiger le rapport en utilisant LaTeX car il permet d'inclure facilement des extraits de code source avec une mise en évidence des éléments syntaxiques. On trouve des applications permettant de rédiger collaborativement des documents LaTeX (`https://fr.sharelatex.com/`, `https://fr.overleaf.com/`). Pour le partage de code l'outil git et son interface Web GitHub sont recommandés. Dans le document, il est conseillé d'utiliser des schémas pour illustrer vos propos.

Une archive [1] (tgz, jar ou zip) de l'ensemble du code source et de l'exécutable de votre programme devra être produite et déposée sur Plubel dans la zone de dépôt prévue à cet effet.

L'évaluation se fait sur la base du rapport, de la qualité technique de la réalisation.

Sujet 1 : HITS en Map Reduce - 1 étudiant

L'algorithme HITS permet de trouver des nœuds importants dans un graphe. Ces sont les hubs et authorities. Vous devez implanter votre algorithme sur un grand graphe construit à partir d'un jeu de données disponible et utiliser le paradigme Map-Reduce sur Hadoop.

Références :

- `https://fr.wikipedia.org/wiki/Algorithme_HITS`
- `http://www.cs.cornell.edu/home/kleinber/auth.pdf`
- jeux de données `http://snap.stanford.edu/data/index.html`

Sujet 2 : PageRank en Map Reduce - 1 étudiant

L'algorithme PageRank permet d'ordonner les nœuds les plus importants dans un graphe. Vous devez implanter votre algorithme sur un grand graphe construit à partir d'un jeu de données disponible et utiliser le paradigme Map-Reduce sur Hadoop.

1. Les autres formats d'archive ne sont pas autorisés

Références :

- https://fr.wikipedia.org/wiki/PageRank
- jeux de données http://snap.stanford.edu/data/index.html
- PageRank en map reduce http://www2.ift.ulaval.ca/~quimper/Seminaires/files/Introduction_aux_algorithmes_MapReduce.pdf
- https://michaelnielsen.org/blog/using-mapreduce-to-compute-pagerank/

Sujet 3 : HITS et PageRank en MPI - 1 étudiant

Le sujet est identique aux sujets 1 et 2 sauf que l'implémentation doit être réalisée avec MPI. Vous devez réaliser les deux algorithmes sur des graphes de taille plus petite.

Sujet 4 : Jeu de la vie A - 1 étudiant

Le jeu de la vie est un automate cellulaire bien connu du monde informatique. Il s'agit ici de l'étendre à 3 dimensions et de permettre de lancer des simulations en utilisant plusieurs machines afin d'accélérer les traitements et de trouver des configurations stables.

Références :

- https://boowiki.info/art/automates-cellulaires/jeu-de-la-vie.html
- https://en.wikipedia.org/wiki/3D_Life

Sujet 5 : Jeu de la vie B - 2 étudiants

Le jeu de la vie est un automate cellulaire bien connu du monde informatique. L'algorithme hashlife permet de calculer efficacement les états du jeu sur plusieurs générations, et d'anticiper les patterns futurs. Le but de ce projet est de réaliser une implémentation de l'algorithme hashlife de manière distribuée. La technologie est au choix.

Références :

- https://johnhw.github.io/hashlife/index.md.html
- https://www.drdobbs.com/jvm/an-algorithm-for-compressing-space-and-t/184406478

Sujet 6 : Ray Tracing distribué en MPI - 2 étudiants

L'objectif est de tester la capacité de passage à l'échelle d'un algorithme de lancer de rayons pour des scènes simples comprenant des primitives comme sphères, cubes, cylindres, tores, etc.

Les éléments constituant la scène seront codés dans le programme. Il pourront être mobiles. Vous devrez ensuite générer une série d'images et éventuellement les reconstituer pour former une animation.

Votre solution devra également proposer une mesure de performance. L'implémentation doit être réalisée en MPI.

Sujet 7 : Gestion de comptes bancaires - 1 étudiant

On cherche à développer un système de gestion de comptes bancaires à l'aide d'Akka. Un banquier est en charge de plusieurs comptes, et travaille dans une banque. Lorsqu'un client se présente à la banque pour déposer ou retirer de l'argent, la banque doit transmettre la demande du client au banquier responsable de son compte.

Un mécanisme de persistance doit être implémenté en liaison avec un SGBD, afin de pouvoir retrouver l'état des comptes des clients même en cas d'arrêt du système.

Sujet 8 : SVD distribuée - 2 étudiants

La décomposition en valeur singulière (SVD pour *singular value decomposition*) est utilisée dans de nombreux systèmes de recommandation ou de compression. Elle permet de réduire la dimensionnalité des variables et de les organiser selon deux espaces. Par exemple un espace pour décrire des utilisateurs et un autre pour décrire des films à partir d'une matrice originale utilisateurs-films qui contient les notes attribuées aux films par différents utilisateurs.

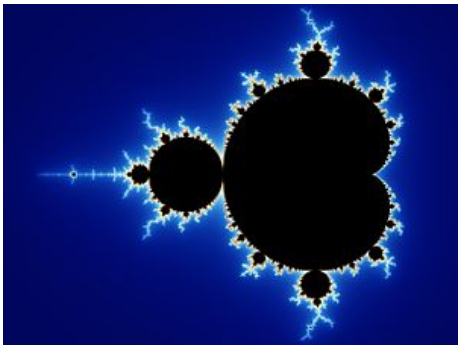

FIGURE 1 – Illustration de l'ensemble de Mandelbrot

À partir des documents et vidéos de G. Strang, programmer la SVD pour des données volumineuses (matrices de plusieurs millions d'éléments) en utilisant les capacités de calcul d'un système distribué. Il s'agit plus précisément d'implanter une version distribuée de l'algorithme ALS.

Références :

- `https://www.youtube.com/watch?v=TX_vooSnhm8`
- `http://www.math.kent.edu/~reichel/courses/intr.num.comp.1/fall11/lecture7/svd.pdf`
- `https://wwwpub.utdallas.edu/~herve/Abdi-SVD2007-pretty.pdf`

Sujet 9 : Alègbre linéaire, valeurs propres - 1 étudiant

Programmer la méthode de la puissance itérée pour déterminer la valeur propre dominante d'une matrice. Cette valeur est souvent utilisée pour calculer le PageRank d'un graphe dirigé.

Déterminer le principe de programmation permettant d'utiliser plusieurs machines pour accélérer la calcul. Effectuer les tests de performance sur plusieurs tailles de matrices et comparer avec la version séquentielle utilisant un seul coeur de CPU.

- `https://fr.wikipedia.org/wiki/M%C3%A9thode_de_la_puissance_it%C3%A9r%C3%A9e`
- `https://cel.archives-ouvertes.fr/cel-01066570/file/AnalyseNumerique.pdf`

Sujet 10 : Exclusion mutuelle - 2 étudiants

Implanter les trois types d'algorithmes d'exclusion mutuelle vus en cours pour les systèmes distribués et des exemples courants (gestion distributeurs de billets par exemple) pour montrer leur efficacité (technologie au choix : MPI, JMS ou Akka).

Sujet 11 : Ensemble de Mandelbrot - 2 étudiants

Utiliser un cluster de machines pour calculer l'ensemble de Mandelbrot et le représenter (figure 1). L'ensemble de Mandelbrot est une fractale définie comme l'ensemble des points $c \in \mathbb{C}$ du plan pour lesquels la suite définie suivante est bornée :

$$\begin{cases} z_0 = 0 \\ z_{n+1} = z_n^2 + c \end{cases}$$

Votre programme doit pouvoir permettre de spécifier une zone pour effectuer un zoom et il devra disposer d'une interface graphique pour représenter une image en couleurs des frontières de l'ensemble de Mandelbrot, c'est-à-dire, ne pas se concentrer uniquement sur les points de l'ensemble en noir mais représenter en couleur les autres en fonction de leur vitesse de divergence. La couleur est attribuée aux points n'appartenant pas à l'ensemble, elle dépend du nombre d'itérations au bout desquelles la suite correspondante est déclarée divergente. Par exemple un théorème montre que la suite est divergente si le module de z est strictement supérieur à 2.

Vous pourrez également représenter des ensembles de Julia ou un Mandelbulb (en option).

Références :

— A. K. Dewdney, Computer recreations : A computer microscope zooms in for a close look at the most complicated object in mathematics, Scientific American 1985, https://static.scientificamerican.com/sciam/assets/media/inline/blog/File/Dewdney_Mandelbrot.pdf.
— https://fr.wikipedia.org/wiki/Ensemble_de_Mandelbrot
— http://images.math.cnrs.fr/Mandelbulb.html

Sujet 12 : Opérateurs de l'algèbre relationelle - 2 étudiants

Implanter au moins 5 opérateurs de l'algèbre relationnelle en utilisation Map Reduce ou bien MPI. Si vous choisissez Map Reduce, vous prendrez l'hypothèse que les tables sont des fichiers CSV.

Sujet 13 : Simulation N-corps - 2 étudiants

Le problème des N corps consiste à calculer la position de N planètes soumises à la loi de la gravitation. Ce problème nécessite de grandes quantités de calculs et vous devez utiliser un système distribué pour réaliser des exécutions efficaces. On suppose que les corps interagissent en 2D.

Références :
— https://fr.wikipedia.org/wiki/Probl%C3%A8me_%C3%A0_N_corps
— https://www.cs.usask.ca/~spiteri/CMPT851/notes/nBody.pdf

Sujet 14 : Transactions distribuées - 1 étudiant

La notion de transaction distribuée est une extension de la notion de transaction à n systèmes. Par exemple le transfert d'argent entre 2 comptes dans 2 banques différentes est une transaction distribuée. Proposer une bibliothèque Java utilisant un système de messages (JMS) ou des acteurs (Akka) pour réaliser cela. Montrer sur des expériences que vos opérations sont fiables.

Références :
— https://ecariou.perso.univ-pau.fr/cours/sd-m1.old/cours-7-algo-distribuee-2.pdf

Sujet 15 : Algorithmes d'élection - 2 étudiants

Implanter l'algorithme d'élection de leader de Chang et Roberts en Java et en utilisant un intergiciel orienté message (JMS) ou bien le principe des acteurs (Akka). Tester votre algorithme sur un exemple de simulation. Dans le document expliquer l'utilité de l'algorithme sur des cas d'utilisation. Votre implémentation doit être utilisable comme une librairie.

Références :
— https://en.wikipedia.org/wiki/Chang_and_Roberts_algorithm
— https://members.femto-st.fr/sites/femto-st.fr.Laurent-Philippe/files/content/lessons/election.pdf
— https://www.i3s.unice.fr/~tettaman/Classes/SD/distributed-systems-tp5.pdf

Sujet 16 : Commit à 2 phases - 1 étudiants

Implanter l'algorithme de commit à deux phases opérants sur des fichiers au moyen de Java RMI ou de MPI. Mettre en place un scénario de cas de tests pour évaluer sa robustesse.