Simulation d'algorithmes d'équilibrage de charge dans un environnement distribué

Kevin Barreau Guillaume Marques Corentin Salingue

15 avril 2015

université de BORDEAUX

Explication du sujet

Environnement distribué

- Base de données répartie sur plusieurs machines physiques
- Réplication multi-maîtres

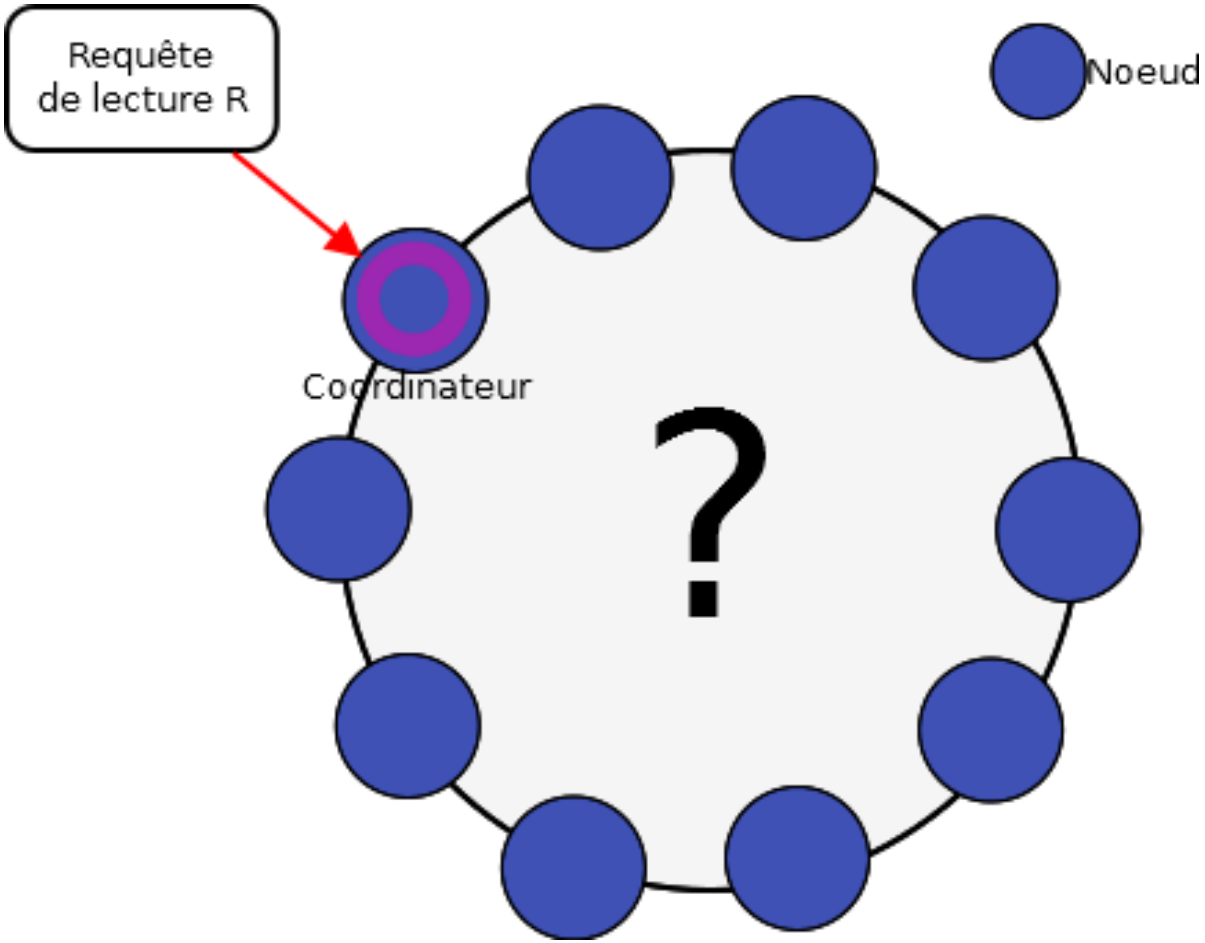
Algorithmes d'équilibrage de charge

- Créés par le client
- Basés sur la réplication des données

Simulation

- Comparaison de l'efficacité des différents algorithmes
- Objectif du projet $\neq$ mise en production

Explication du sujet



Axes de développement

- Base de données (*Cassandra*)
  - Gestion des requêtes
  - Gestion de la réplication
- Application cliente (*Driver Java Cassandra*)
- Visualisation (*Graphite*)

Base de données Cassandra

Originellement créée et développée par **Facebook** en 2008 (maintenant un projet de la **Fondation Apache**), elle possède comme caractéristique d'être :

- NoSQL, orientée colonnes
- Open-source (licence Apache 2)
- Écrite en Java
- Décentralisée

Le choix de Cassandra

- Open-source
- Développement actif
- Proche du projet à réaliser
- Connaissances dans l'équipe

Solutions alternatives : HBase, CouchBase, CouchDB, from scratch...

Fonctionnement

Alert

Expliquer comment fonctionnent le client et cassandra avec des schemas, capture d'écran et tout...

Fonctionnement de Cassandra

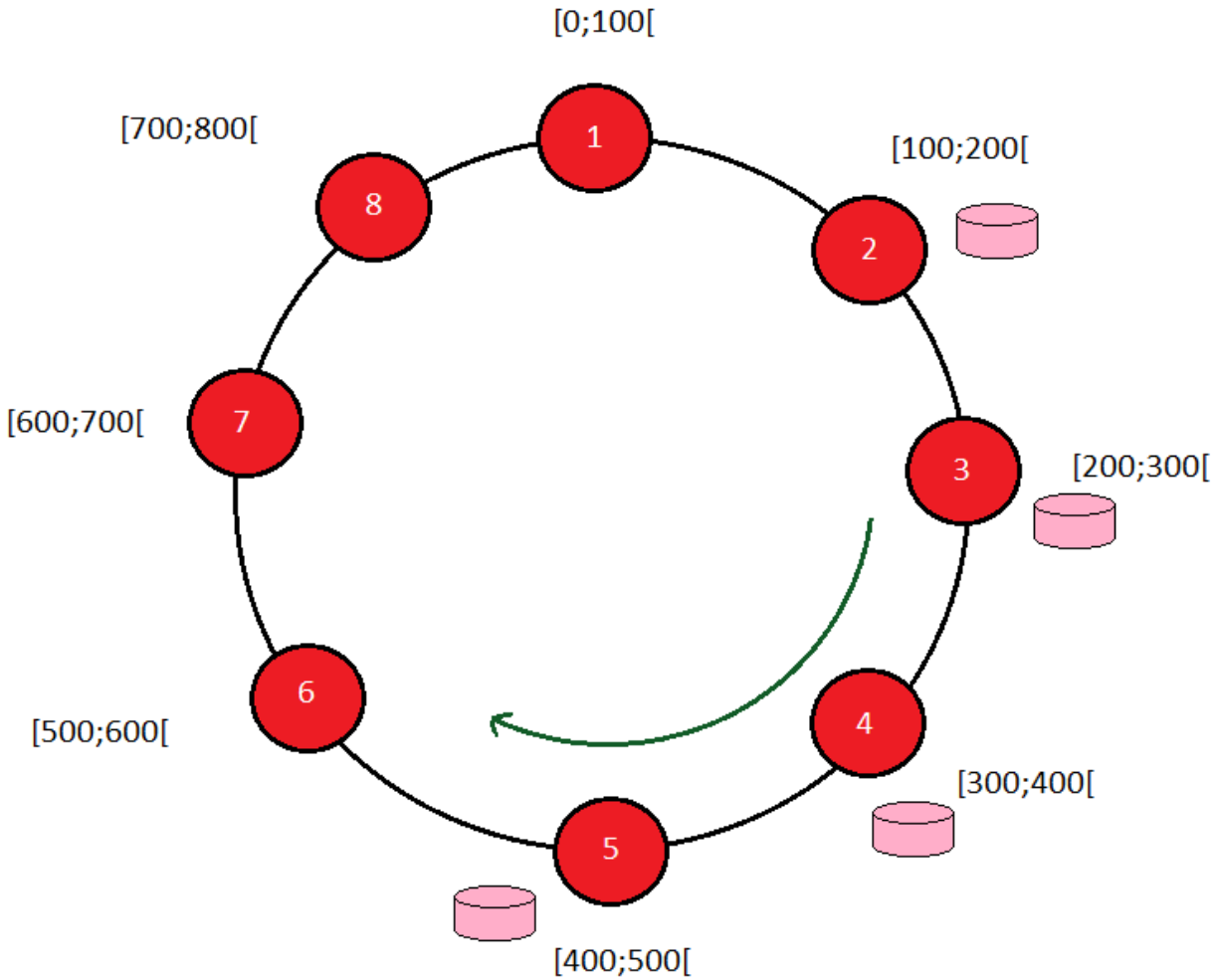
Gestion des requêtes

✔A chaque requête de lecture traitée, un message est envoyé pour notifier aux autres noeuds de ne pas effectuer la requête

Gestion de la réplication

✔Le placement des données et de leurs copies selon différentes fonctions de hachage.
✖La gestion de popularité des objets.

Stratégie de réplication simple



Gestion de la popularité

Paramètres

r = Nombre de requêtes total effectuées durant l'intervalle de temps T ;
n = Nombre de noeuds dans le réseau ;
p = Popularité d'un objet ;
k = Nombre de copies de l'objet.

- Augmenter le nombre de copies si $2 \times \frac{r}{n} \geq \frac{p}{k}$ vraie.
- Diminuer le nombre de copies si $\frac{r}{2n} \leq \frac{p}{k}$ vraie.

Application cliente

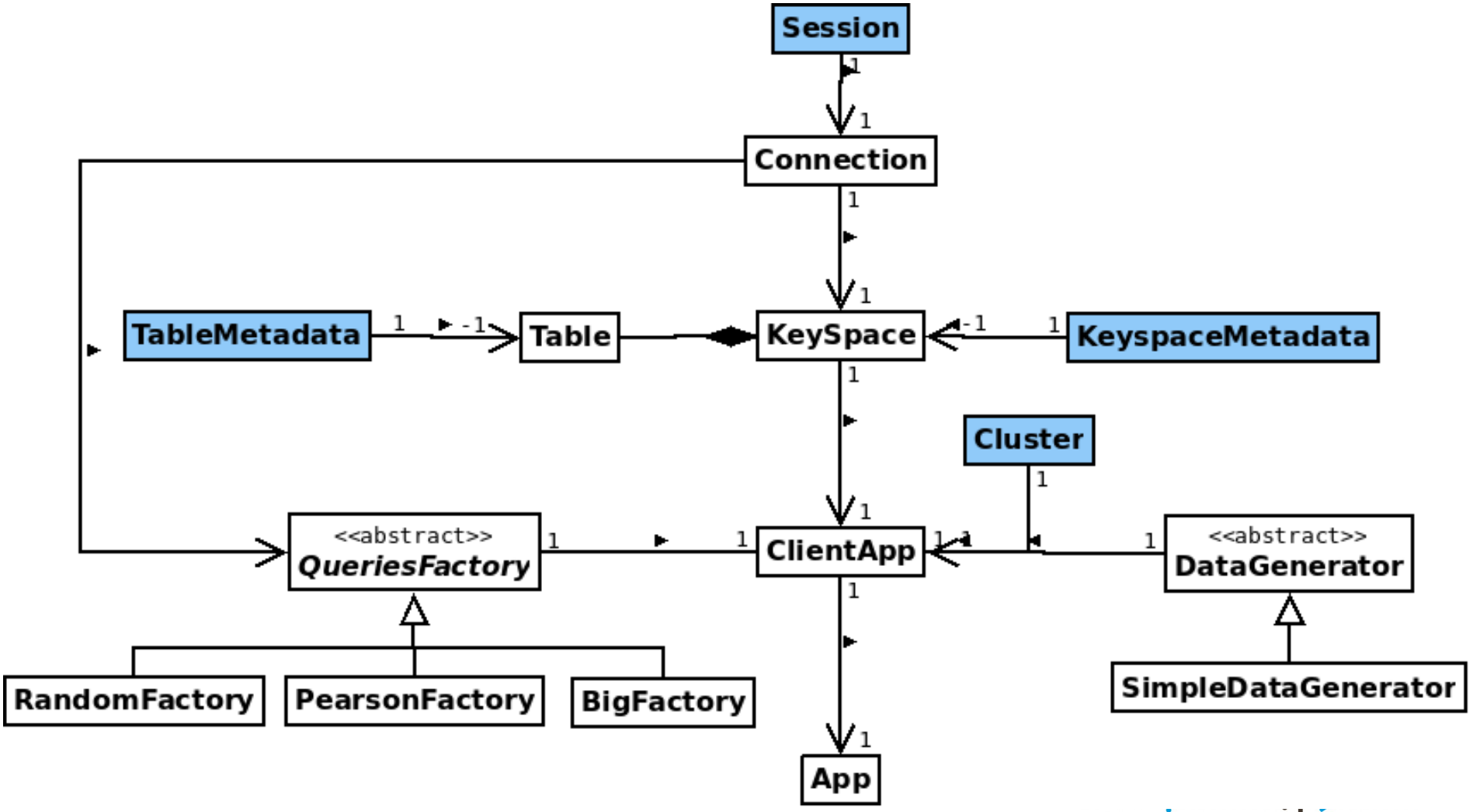
Technologies employées

- Développé en Java
- Utilisation d'un pilote informatique

Pilote utilisé

- DataStax Java Driver 2.0
- Développé par l'entreprise DataStax
- Communication avec la base de données Cassandra

Architecture du client



Fonctionnement du client

Initialisation

- Connexion à la base de données
- Choix du keyspace

Console

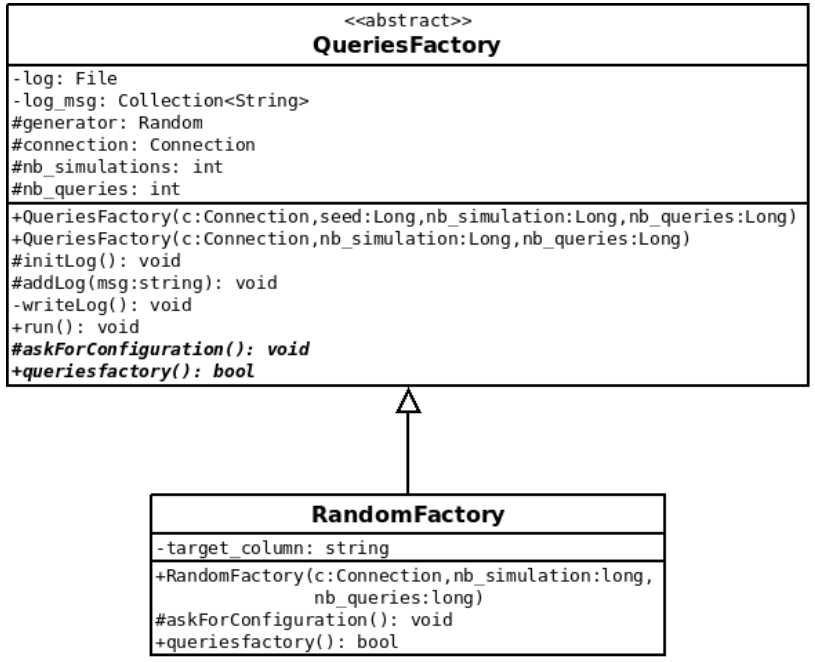
L'utilisateur saisie la commande qu'il souhaite exécuter, notamment :

- Changement de cluster
- Création de jeu de données
- Exécution d'un générateur de requêtes

Fonctionnement du client

capture d'écran

Générateur de requêtes



Personnalisable

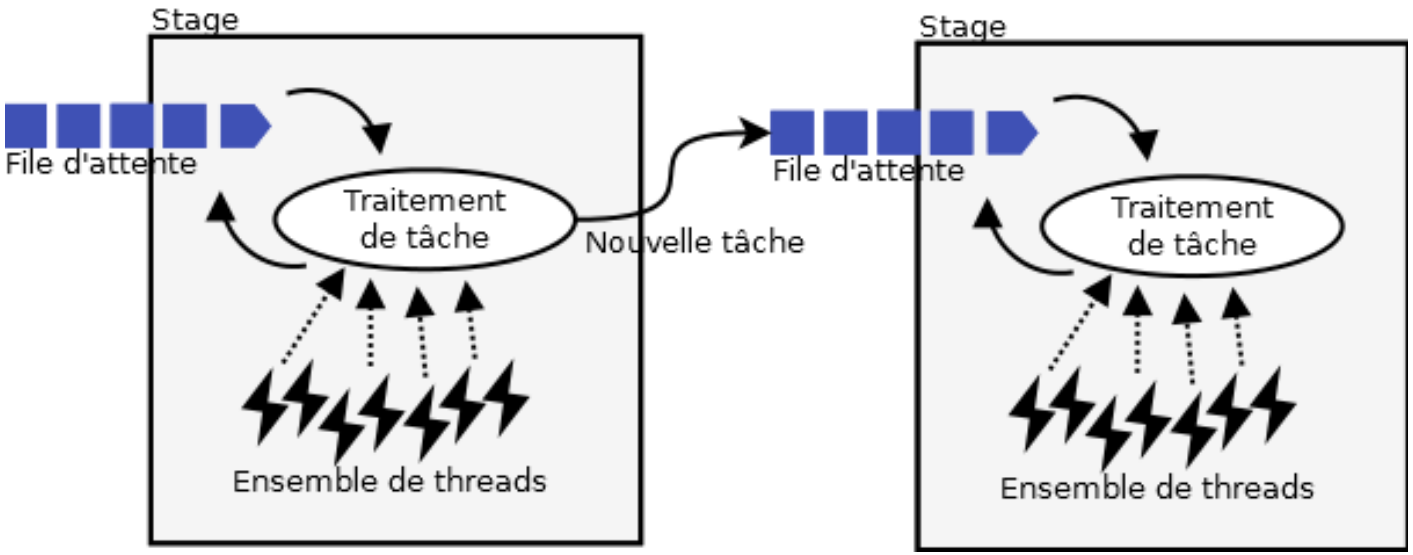
- Possibilité d'ajouter des générateurs de requêtes
- Choix du générateur
  `queries NomGenerateur <seed> <nb_simulations> <nb_requetes>`

Tests

Architecture de Cassandra

Staged event-driven architecture (SEDA)

- **Stage** → emplacement pour réaliser des tâches
  - **File d'attente** → messages de tâches à traiter
  - **Threads** → exécuteurs de tâches



Architecture de Cassandra

Staged event-driven architecture (SEDA)

- **Stage** → emplacement pour réaliser des tâches
  - **File d'attente** → messages de tâches à traiter
  - **Threads** → exécuteurs de tâches

Stages présents dans Cassandra :

- READ
- **READ_REMOVE**
- MUTATION
- GOSSIP
- ...

Points techniques : Réplication

Solution initiale		Solution implémentée	
Donnée n° 1	Donnée n° 2	Donnée n° 1	Donnée n° 2
$H_0(c1)$	$H_0(c2)$	$H_0(c1)$	$H_0(c2)$
1er réplica	1er réplica	1er réplica	1er réplica
$H_1(c1)$	$H_1(c2)$	$H_1(H_0(c1))$	$H_1(H_0(c2))$
2nd réplica	2nd réplica	2nd réplica	2nd réplica
$H_2(c1)$	$H_2(c2)$	$H_2(H_0(c1))$	$H_2(H_0(c2))$

Points techniques

Alert

Point technique sur le client (distribution ?), car peu de choses intéressantes avec Cassandra

université de BORDEAUX

Tests

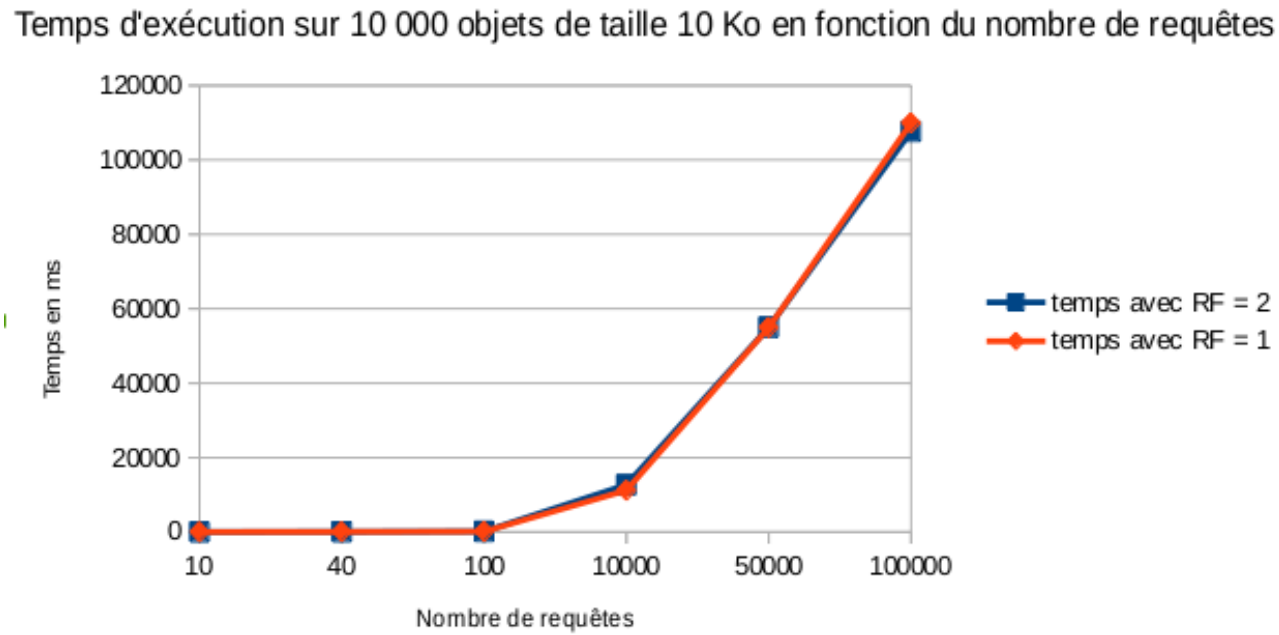
Environnement

- Les tests de mesures de performances se déroulent dans un réseau d'Amazon EC2 de 10 noeuds.
- La base de données est composée de 10 000 objets de taille unique.

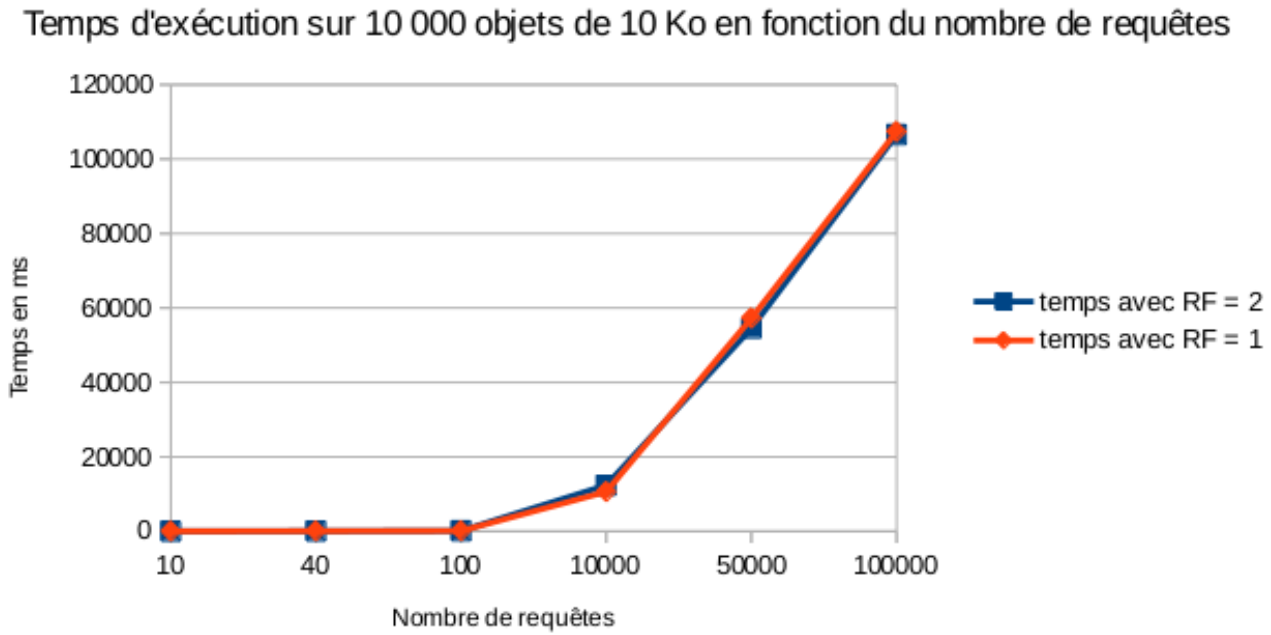
Dénomination

- RF = nombre de copies + donnée originale
- petits objets = un texte généré aléatoirement de 10 Ko
- gros objets = un texte généré aléatoirement de 1 Mo

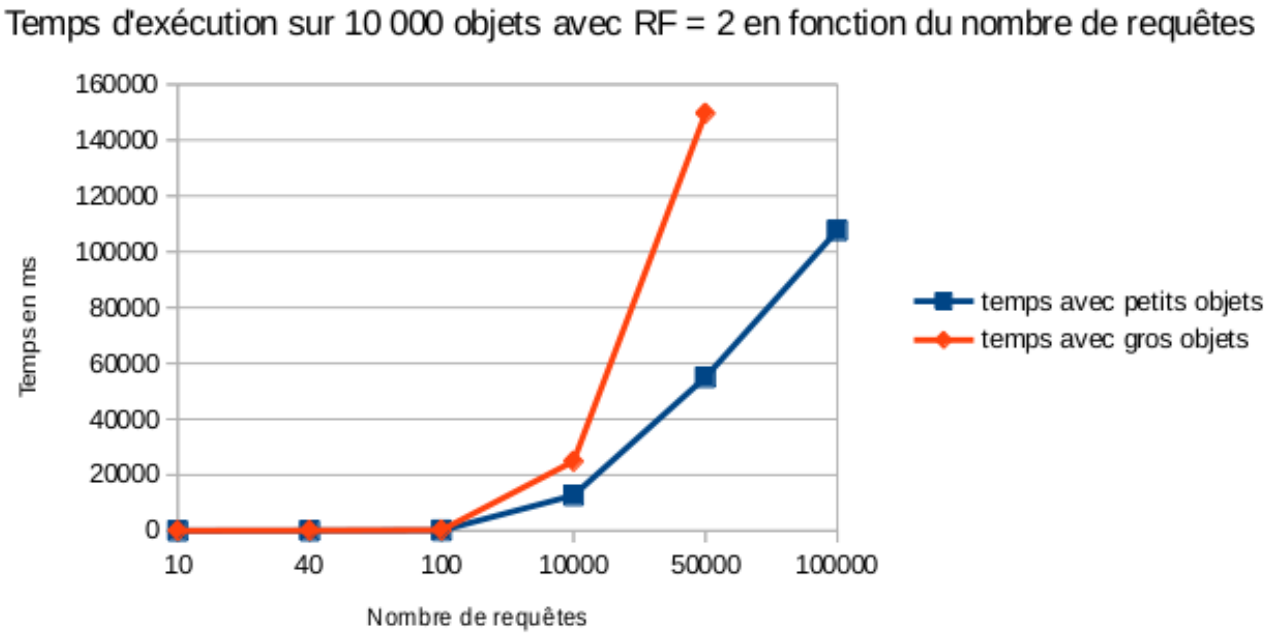
Sur Cassandra modifiée



Sur Cassandra non modifiée



Sur Cassandra modifiée



Améliorations possibles sur Cassandra

Gestion des requêtes

- Algorithmes de réaffectation SVLO et AverageDegree

Gestion de la réplication

- Popularité des objets

Blocks

Standard

This is a standard block.

Example

This is an example.

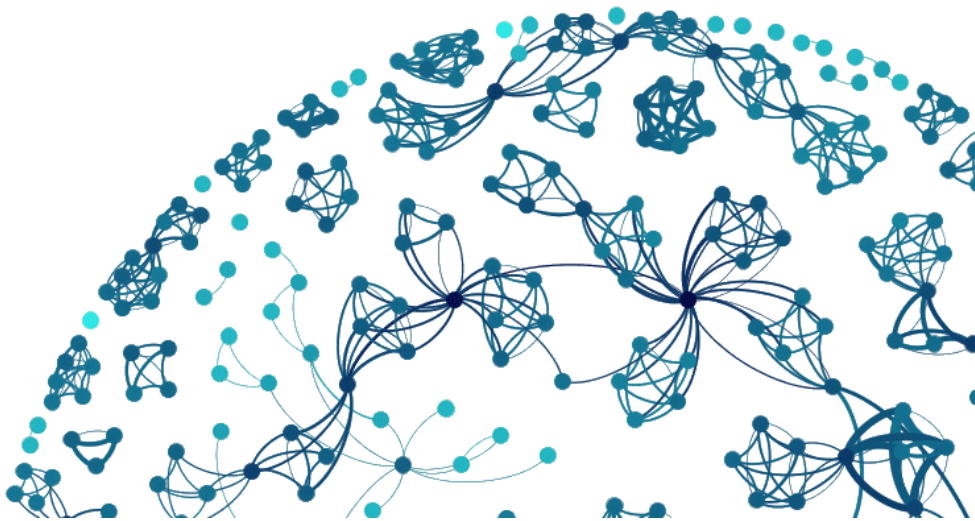
Alert

This is important.

Example

Complex networks

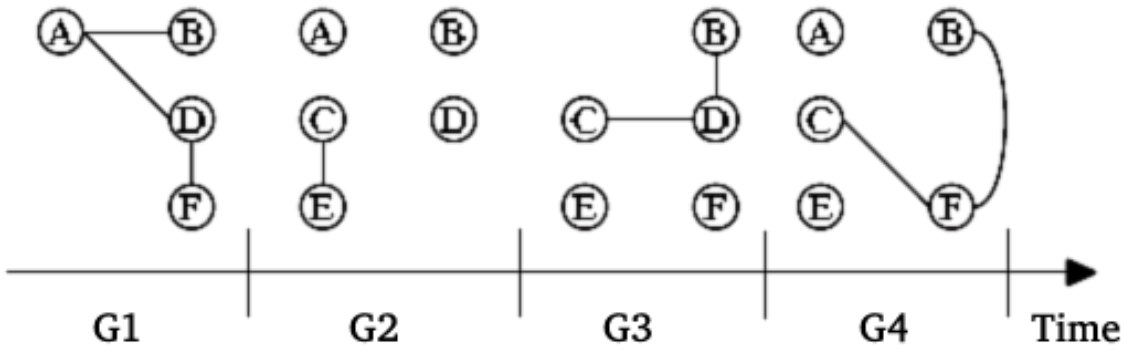
- Sociology : social networks, call networks
- Informatics : Internet, Web, peer-to-peer networks
- Biology, linguistics, etc.



Example

Evolving network

Nodes and links appearing over time.



Questions ?

Thank You !

Title

<first.lastname@lip6.fr>