Scaling a solution of an NP-hard problem in a cluster of machines using Apache ZooKeeper

Кирилл Голоднов

Grammarly

Содержание

- Постановка задачи
- Как работает Apache ZooKeeper и почему мы выбрали именно его
- Как будут работать машины в кластере, с использованием Apache ZooKeeper
- Насколько хорошо всё получилось
- Как можно построить кластер на случай произвольной задачи

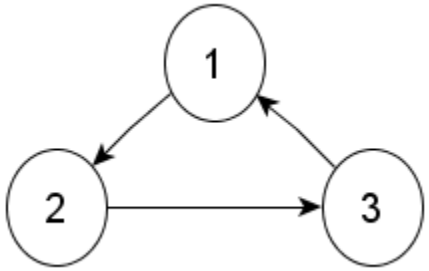
Постановка задачи

Задача на графе:

- Дан ориентированный граф, вершины которого имеет неотрицательный вес
- Найти подмножество вершин, индуцированный подграф которого является ациклическим, и которое имеет максимально возможный суммарный вес

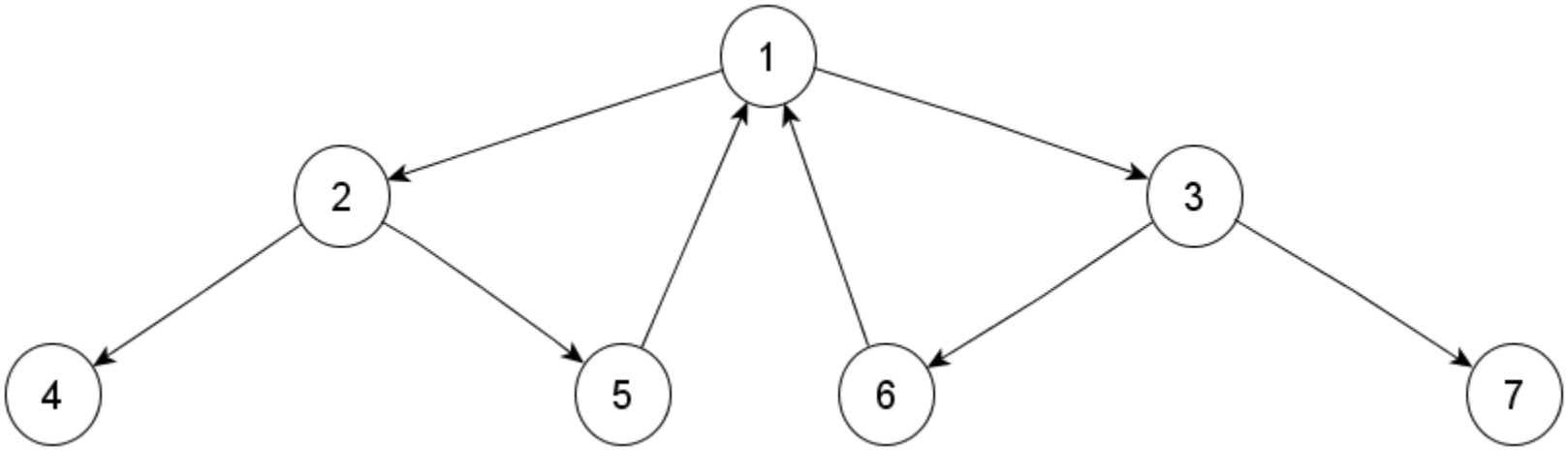
Постановка задачи

Подходят множества вершин: (1,2), (1,3), (2,3) и др.



Постановка задачи

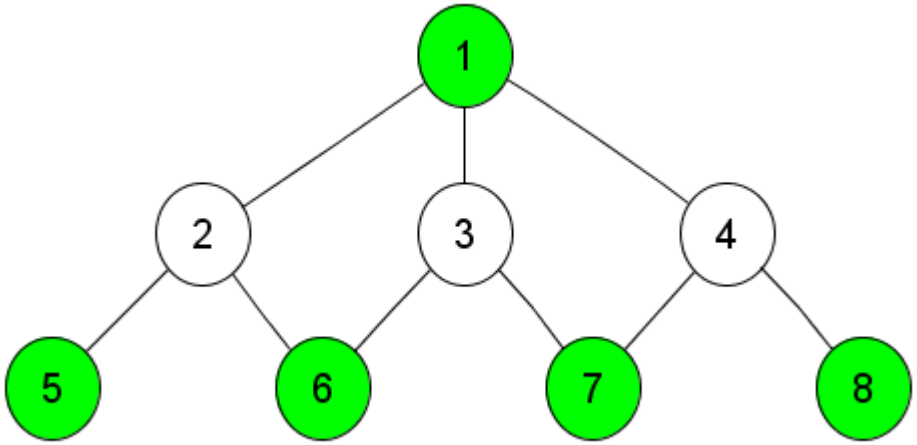
Подходят множества вершин: (1,2,3,4,7), (2,3,4,5,6,7) и др.



Построение алгоритма

Задача NP-трудная?

- При определённых входных данных совпадает с задачей о независимом множестве



Построение алгоритма

- Для каждого подмножества вершин проверить, является ли оно ациклическим подграфом. Для тех, которые являются – выбрать такое, которое имеет максимальный суммарный вес входящих в него вершин

Построение алгоритма

- Алгоритм эффективно распараллеливается. Промежуток $[0, 2^{|V|})$ (маски всех возможных подмножеств вершин графа) разбиваем на куски, и обрабатываем на разных машинах

Распараллеливание

Как можно решать задачу в кластере:

- MapReduce (с использованием Hadoop)
- Apache Kafka
- JMS (с использованием очередей)
- Сервера-координаторы (Apache ZooKeeper)

Распараллеливание

Недостатки подходов:

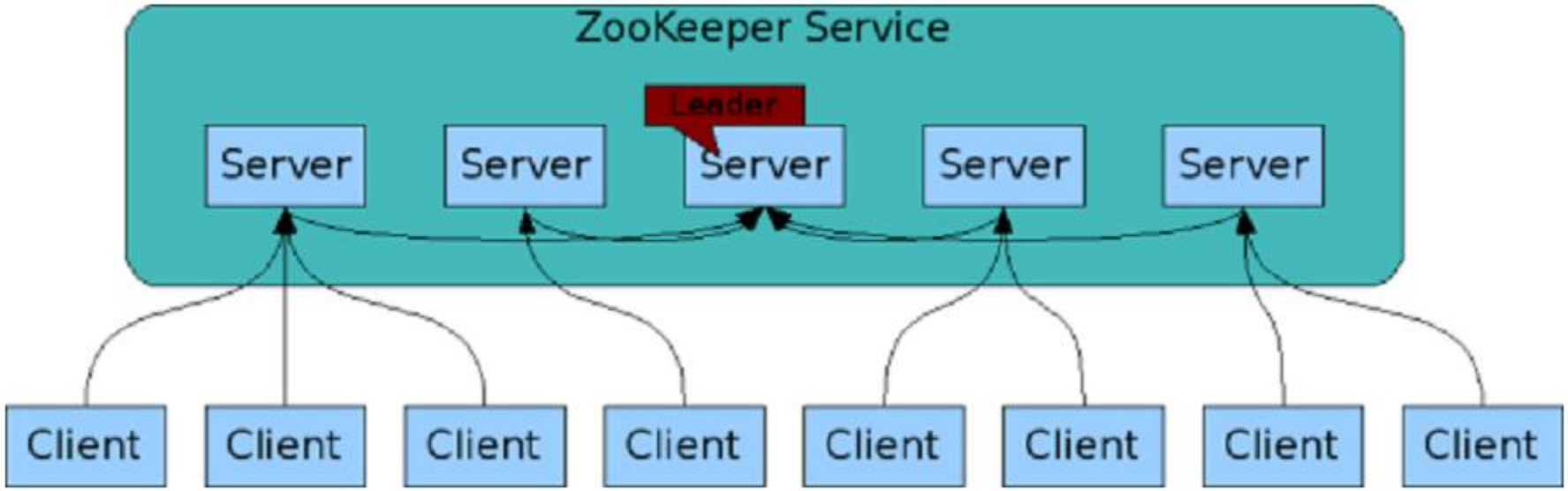
- MapReduce слишком «тяжелый» для этой задачи
- У остальных методов проблемы при добавлении/удалении машин из кластера

Apache ZooKeeper

- Обеспечивает координацию между серверами
- Хорошо следит за тем, когда появляются и исчезают машины из кластера

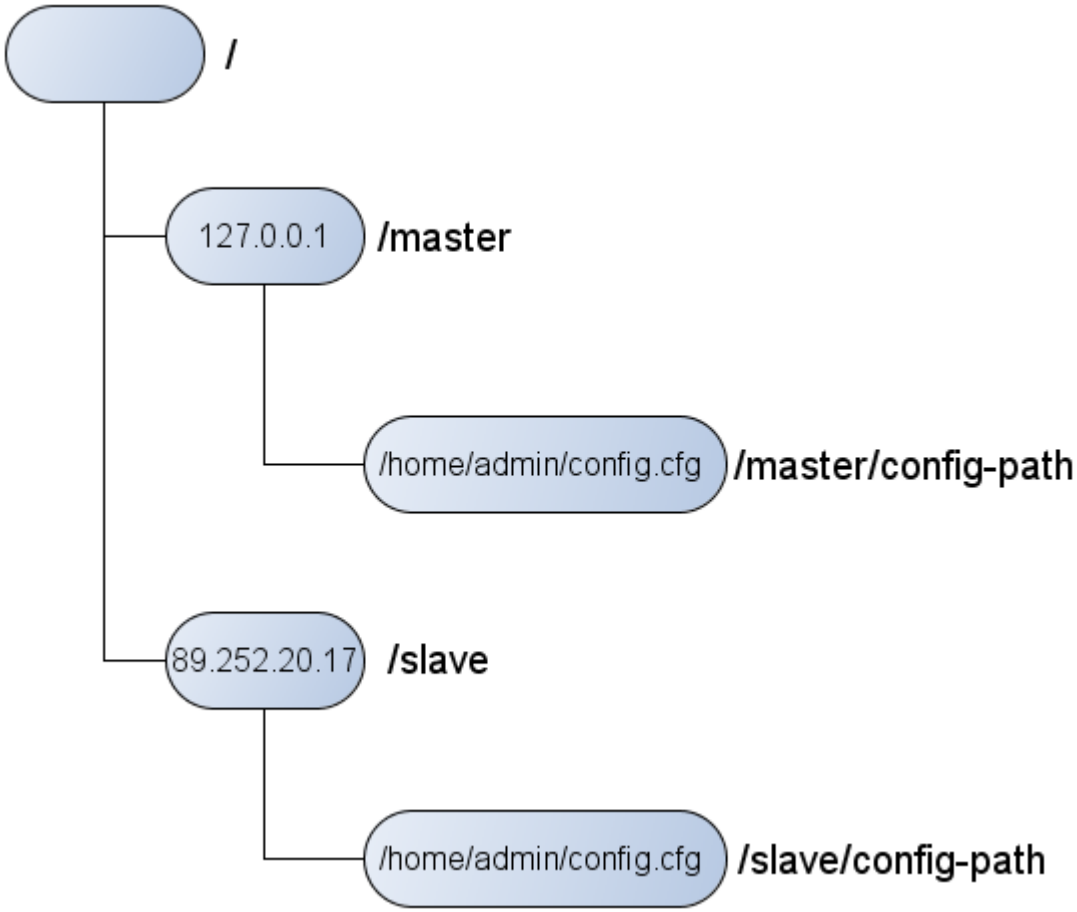
Apache ZooKeeper

Кластер ZooKeeper:



Apache ZooKeeper

Z-ноды:



Apache ZooKeeper

Операции над Z-нодами:

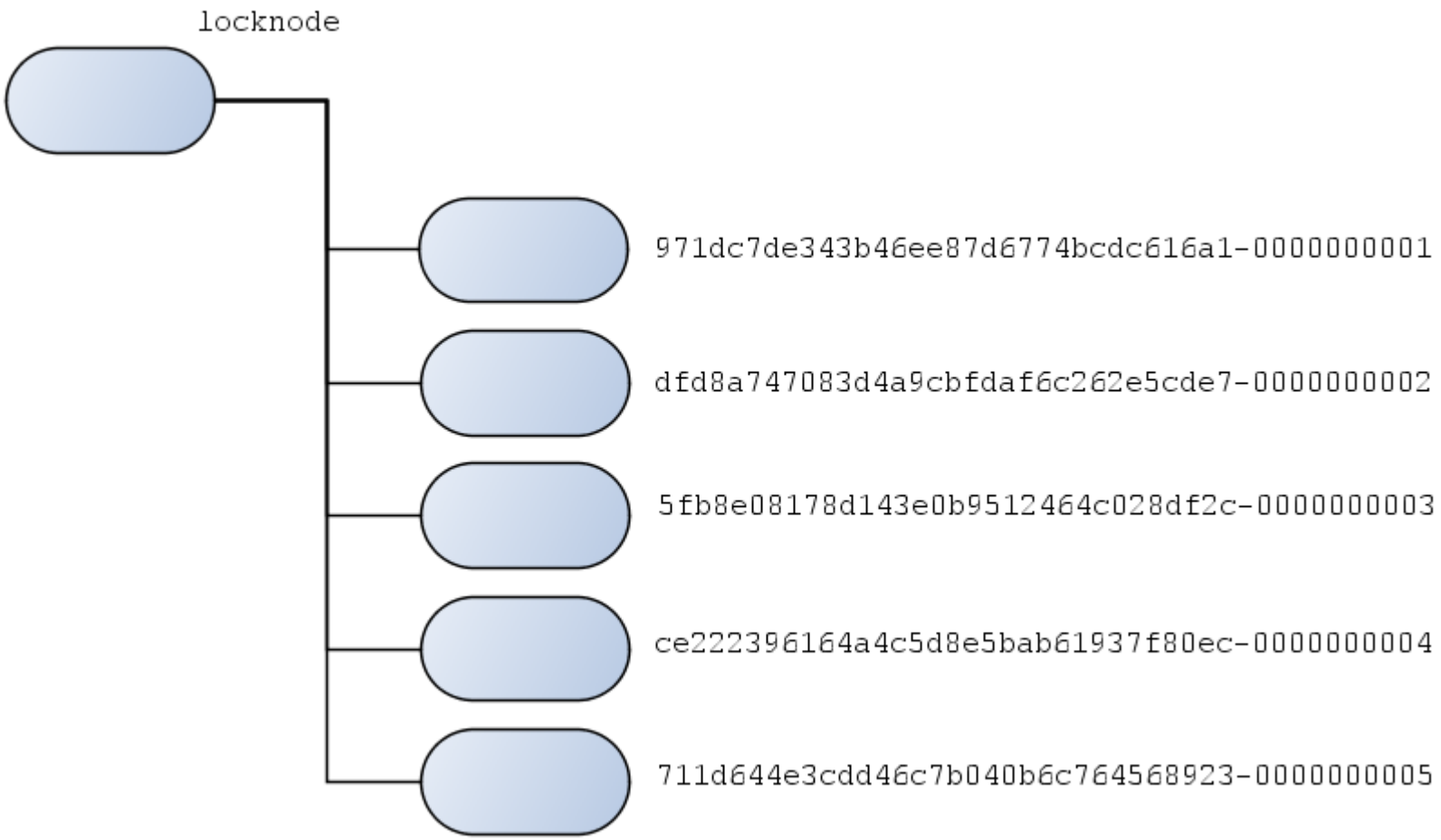
- Create
- Delete
- Exists
- Get data
- Set data
- Get children
- Sync
- Set watch

Apache ZooKeeper

Гарантии, которые предоставляет Apache ZooKeeper:

- Sequential consistency
- Атомарность (Atomicity)
- Single System Image
- Надёжность (Reliability)
- Своевременность (Timeliness)

Распределённый мьютекс



Apache Curator Framework

High-level API над Apache ZooKeeper

- Значительно упрощает работу с API ZooKeeper
- Содержит в себе реализацию готовых «рецептов», которые можно использовать в распределённых системах

Apache Curator Recipes

- Elections
- Locks
- Barriers
- Counters
- Queues

Apache Curator Recipes

pom.xml:

```
<dependency>
        <groupId>org.apache.curator</groupId>
        <artifactId>curator-recipes</artifactId>
        <version>2.7.1</version>
</dependency>
```

Apache Curator Recipes

Из «рецептов» будем использовать:

- InterProcessSemaphoreMutex (распределённый лок)
- PathChildrenCacheListener (слушает изменение дочерних Z-нод для заданной ноды)

Структура Z-нод

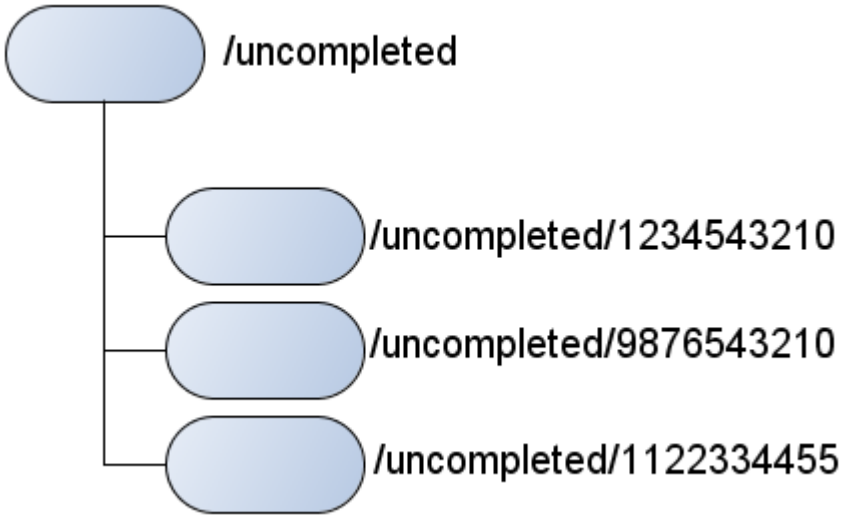
Три основные Z-ноды:

- /completed – содержит результаты работы
- /uncompleted – входные данные о графах
- /calculations – промежуточные вычисления

Структура Z-нод

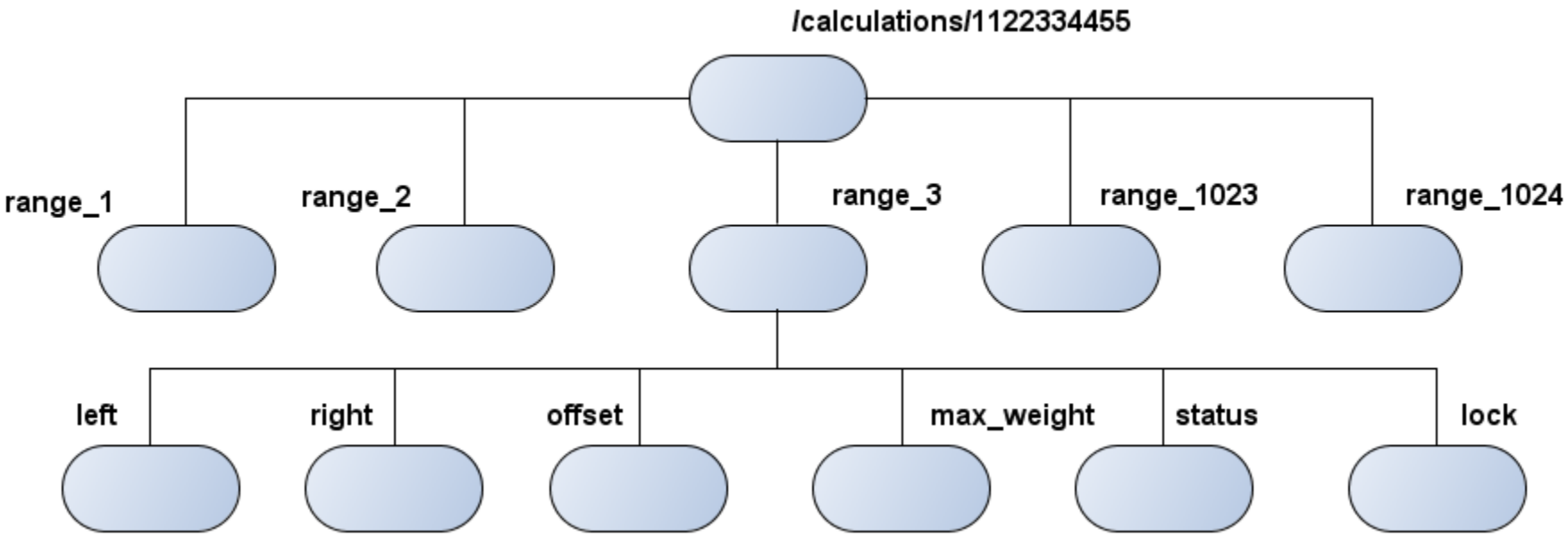
/uncompleted:

- Чайлд-ноды вида /uncompleted/<graph_id>
- Все машины в кластере подписаны на изменение чайлдов Z-ноды /uncompleted



Структура Z-нод

/calculations:



Структура Z-нод

/calculations:

- Разобьем интервал $[0, 2^{|V|})$ на 1024 (или меньше) интервала, но так, чтобы каждый интервал был не меньше 2^{27}
- Если вершин мало (меньше 28), то интервал будет один
- Каждый маленький интервал в любой момент времени будет обрабатываться не более чем одной машиной

Структура Z-нод

/calculations:

- left, right – границы интервала
- offset – смещение от левой границы (left)
- max_weight – текущий максимальный вес (и маска)
- status – было ли уже высчитано (0 или 1)
- lock – Z-нода для создания лока

Структура Z-нод

/completed:

- После того, как максимальные веса на всех интервалах посчитаны, одна из машин обходит все range-ноды, и высчитывает максимальный вес

Как работает машина в кластере

- При захвате лока, берётся смещение (offset), текущий максимальный вес (max_weight)
- Обрабатывается часть интервала и пересчитывается вес
- Обновлённый вес и обновлённое смещение записываются в Z-ноды

Изменение количества машин

При добавлении машины в кластер:

- Она «захватывает» лок на свободный интервал
- Начинает обрабатывать интервал, начиная с нужного смещения (offset)

Изменение количества машин

При удалении машины из кластера:

- Она отпускает лок
- Теряются только данные, обработанные с текущего смещения (offset)

Изменение количества машин

При «мигании» соединения (сети) во время обработки интервала:

- Лок становится «грязным»
- Не записываем обновлённые вес и смещение

Параметры распределённой системы

На сколько интервалов разбивать весь интервал?

- Было выбрано 1024
- Не забыть обернуть создание интервалов в транзакцию

Параметры распределённой системы

Сколько данных обрабатывать перед каждым обновлением offset?

- Было выбрано 2^{27} масок

Масштабируемость

- Время на поиск и захват лока малО
- При увеличении машин в кластере в несколько раз – можем обрабатывать в такое же количество раз больше графов одновременно

Недостатки

Связанные с Apache ZooKeeper:

- На создание Z-нод тратится время
- На каждый граф создаётся до 7175 Z-нод (после работы остаётся только 5)

Недостатки

Связанные с работой машин в кластере:

- Когда необработанных промежутков остаётся меньше, чем количество машин – часть машин будут простаивать

Делаем кластер своими руками

- Есть произвольный алгоритм, который принимает на вход текстовую строку, и возвращает текстовую строку
- Есть распределённое хранилище с файлами, которые содержат строки
- Обработать файлы в кластере, создав файлы с результатами в хранилище

Делаем кластер своими руками

- Для каждого файла создаём свою Z-ноду в ZooKeeper
- Для каждой Z-ноды создаём дочерние Z-ноды lock, offset, status

Вопросы