Разработка и оптимизация алгоритма репликации в облаке Cirrostratus

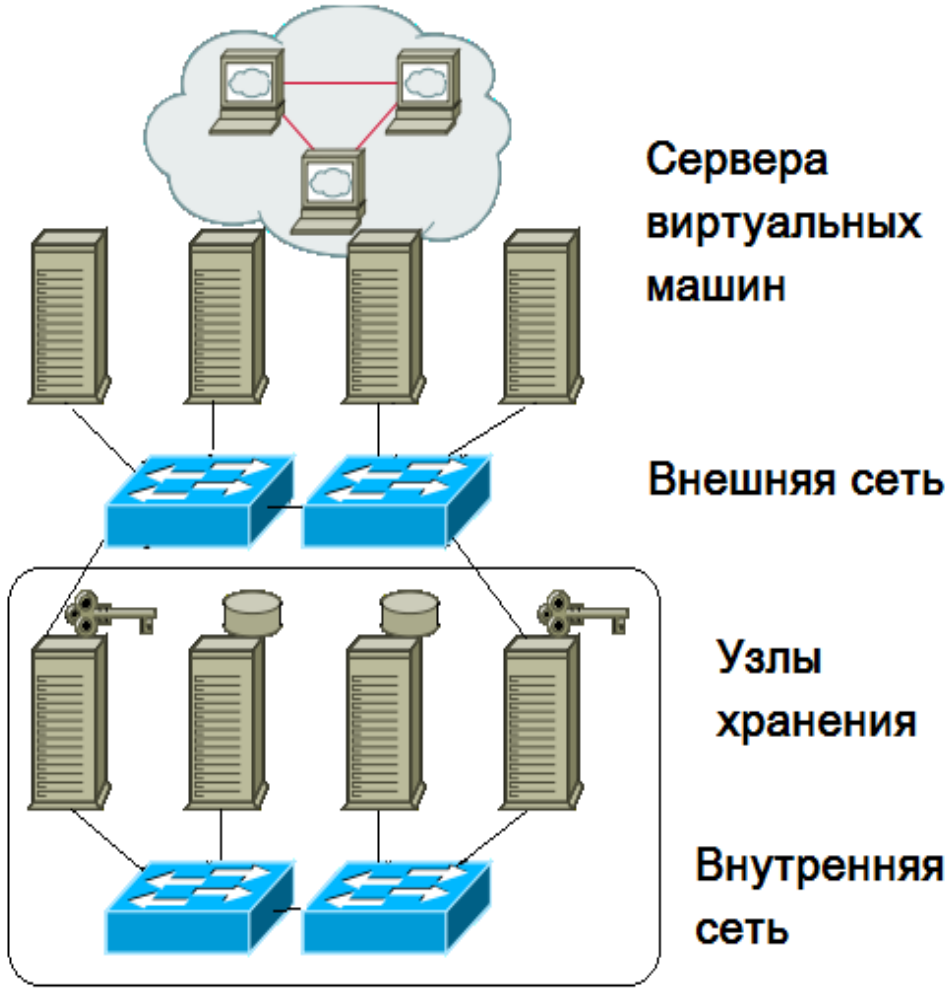
Кирилл Кузнецов, 461 группа

Научный руководитель:
ведущий технический специалист
ООО «АйТи Хаус»
С. В. Богатырев

Рецензент:
Ведущий разработчик
ООО «М-Клауд»
А.Н. Косякин

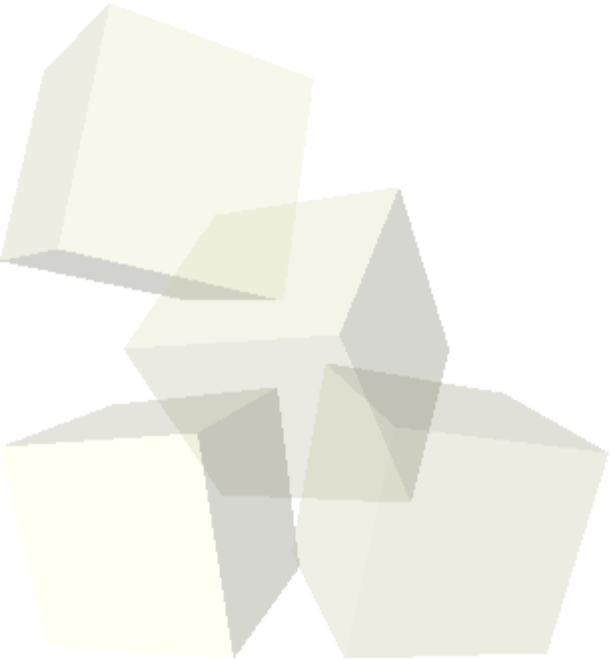
Cirrostratus

- Cirrostratus —распределенное блочное хранилище для для облачной инфраструктуры
- Рассчитано на массовое использование



Алгоритм репликации

- Отвечает за распределение реплик по хранилищу
- Влияет на производительность и масштабируемость хранилища



Постановка задачи

1. Сформулировать требования к алгоритму репликации
2. Проанализировать существующие подходы
3. Адаптировать выбранный подход для Cirrostratus
4. Определить оптимальные параметры алгоритма для различных вариантов структуры кластера
5. Осуществить интеграцию алгоритма в прототип Cirrostratus

Требования

- Сбалансированная нагрузка на устройства с учетом разных возможностей
- Минимальная миграция блоков при изменении кластера
- Минимальное использование метаданных
- Гибкость настройки

Существующие подходы

Подходы не использующие метаданные

	Устойчивое хеширование (Sorrento FS)	Алгоритм Бринкманна	SCADDAR	RUSH/ CRUSH
Сбалансированное размещение	-	+	+	+
Поддержка добавления устройств	+	+/-	+	+
Поддержка удаления устройств	+	+/-	+	+
Гибкость настройки репликации	-	-	-	+

CRUSH: карта сети

- Репликация осуществляется на основе карты сети и правил
- В узлах — контейнеры, каждый определяет свой алгоритм выбора элемента нижнего уровня

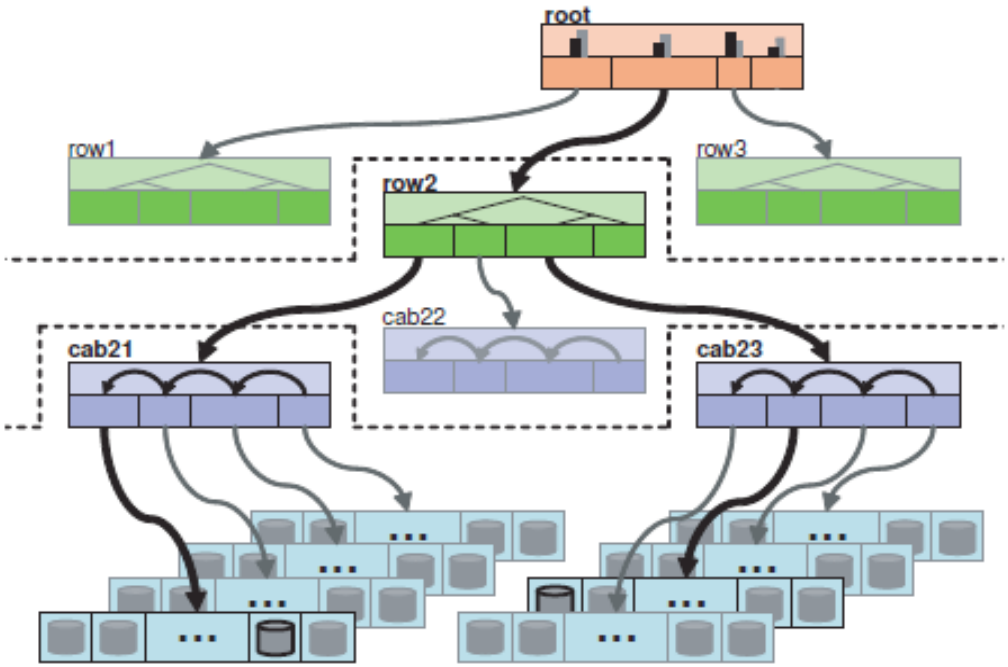
take（root）

select（1, row）

select （ 2, cab)

select （ 1, disk)

emit



CRUSH: Контейнеры

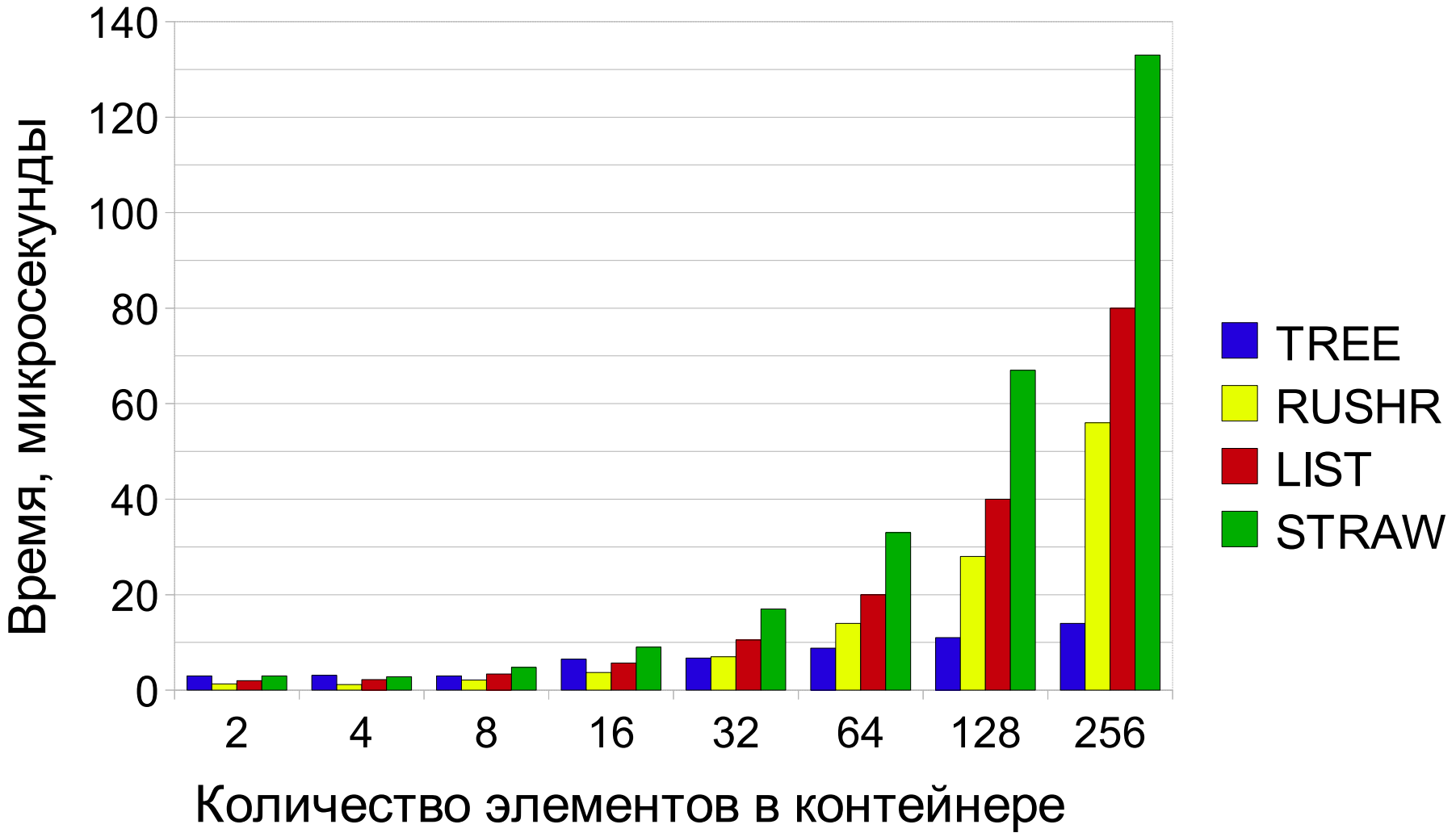
	Скорость	Затраты на добавление устройств	Затраты на удаление устройств
Tree ($RUSH_T$)	O(log n)	Близкие к оптимальным	Средние
List ($RUSH_P$)	O(n)	Оптимальные	Нет поддержки
Straw	O(n), на 30-40% ниже List	Оптимальные	Оптимальные

+

$RUSH_R$	O(n), на 20-25% выше List	Оптимальные	Близкие к оптимальным

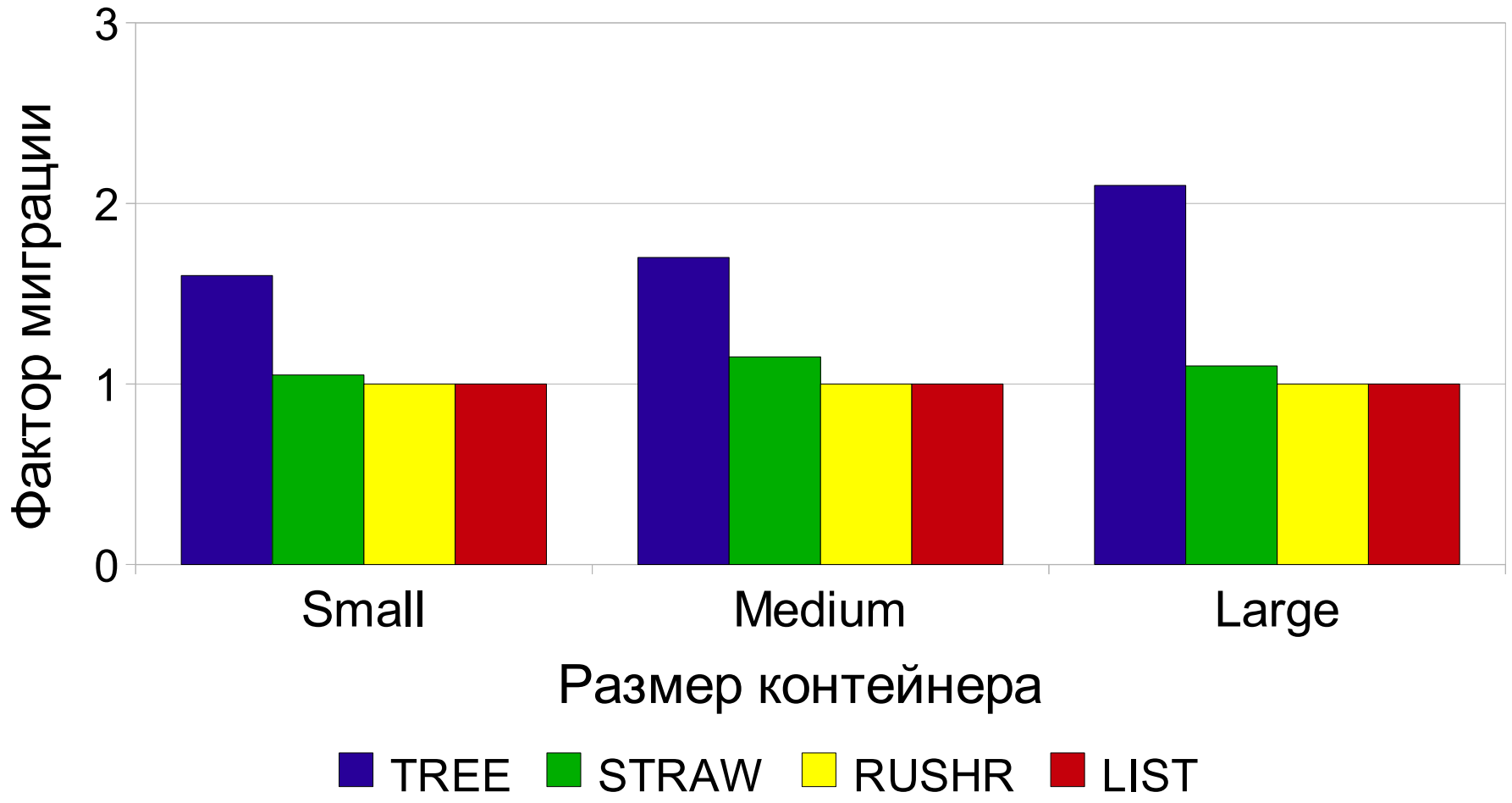
Нет поддержки избыточного кодирования

Время выбора в контейнерах

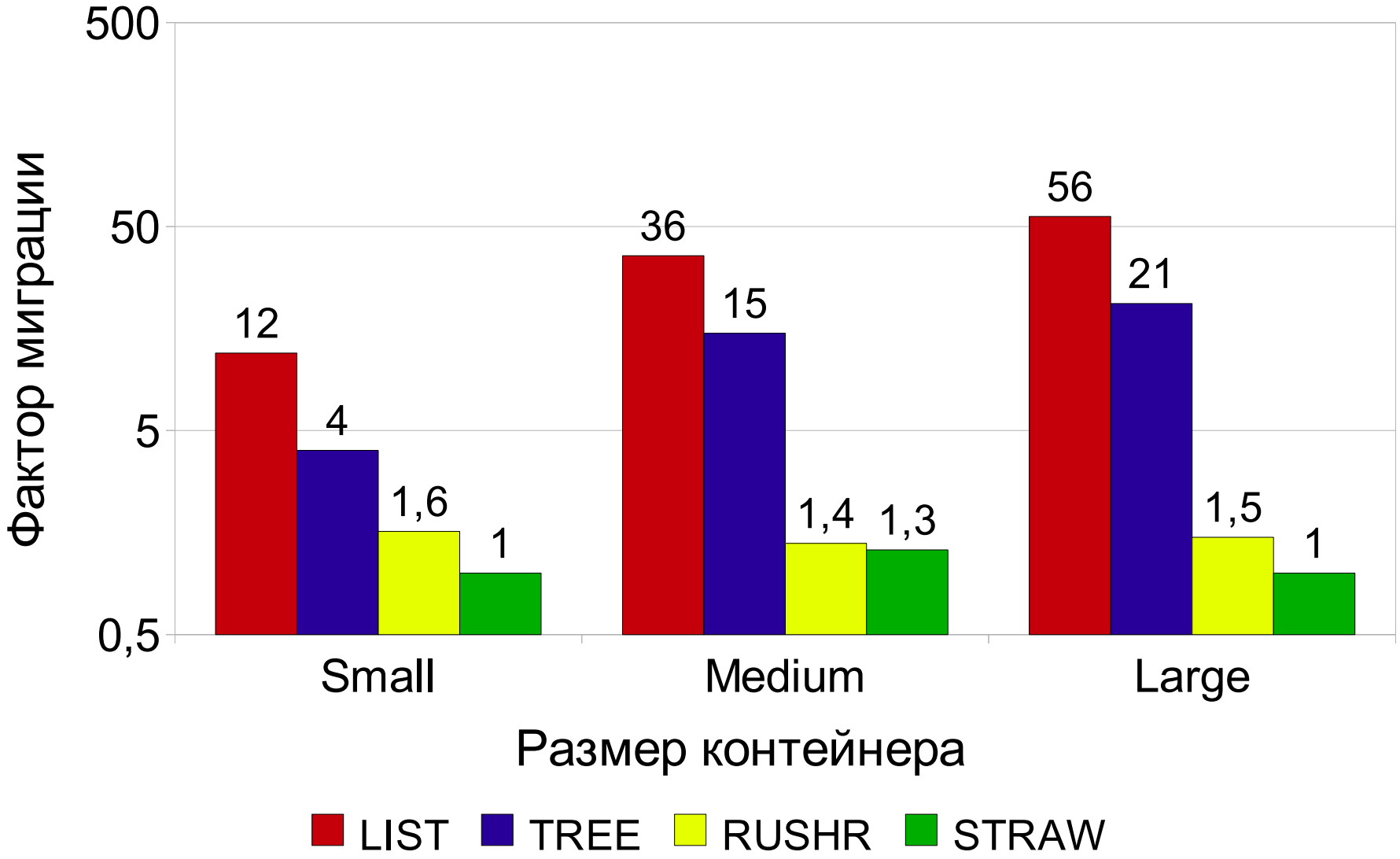


Миграция при добавлении

Фактор миграции: $f = m_{actual}/m_{optimal}$

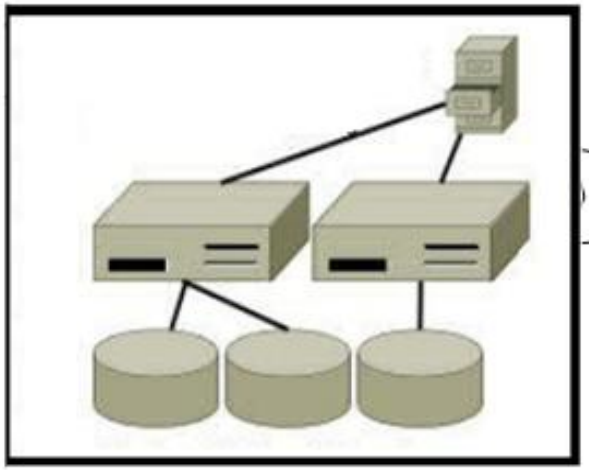


Миграция при удалении

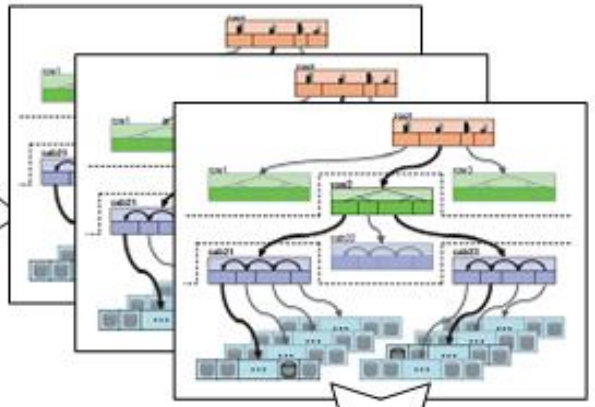


Тестирование карт

Данные кластера:
узлы, диски, веса



Карты сети



- Скорость работы
- Данные по миграции блоков
- Равномерность заполнения устройств

Результаты

1. Сформированы требования к алгоритму репликации

2. Проанализированы существующие подходы, на основе требований выбран алгоритм CRUSH

3. Реализован контейнер, оптимальный для узлов хранения, когда не используется избыточное кодирование

4. Разработан и реализован механизм тестирования карт сети для определения оптимальных параметров

5. Алгоритм интегрирован в прототип Cirrostratus