Исследование и разработка адаптивного объединения запросов для RAID-массива на основе твердотельных накопителей

Автор: Васенина Анна Игоревна, 444 группа
Научный руководитель: д.ф.-м.н., проф. Терехов А.Н.

Санкт-Петербургский государственный университет
Кафедра системного программирования
15 мая 2019

Введение

- NVMe-накопители
  - Быстрые
  - Дорогие
- Система хранения данных RAIDIX ERA
  - Программно-определяемая СХД
  - Ориентирована на RAID-массивы на базе контрольных сумм
- RMW-операции
  - Воникают при записи в полосу данных блока длиной меньше полосы данных
  - Тяжеловесные
  - Количество может быть сокращено за счет объединения запросов

Планировщики ввода-вывода Linux

С единой очередью

Примеры: noop, deadline, CFQ

Из них выполняют объединение запросов: deadline, CFQ

Технология blk-mq

Примеры: none-mq, deadline-mq, kyber

Из них выполняют объединение запросов: deadline-mq

Тип нагрузки	none	blk-sq			blk-mq		
		noop	deadline	cfq	none-mq	mq-deadline	kyber
Случ. чтение	7273	348	323	3240	3130	352	2971
Случ. запись	5411	350	322	3296	2983	352	2924
Посл. чтение	7021	345	321	190	3079	355	2920
Посл. запись	**5111**	**346**	**320**	**192**	**2952**	**352**	**2875**

Тестирование предела производительности планировщиков ввода- вывода. Результаты представлены в тысячах операций в секунду

Постановка задачи

Целью работы является создание подсистемы, управляющей объединением входящих запросов записи на основании анализа входящей нагрузки

Задачи

- Изучить возможные стратегии обнаружения последовательных запросов к СХД
- Проанализировать эффективность объединения последовательных запросов на различных паттернах нагрузки
- Разработать алгоритм, управляющий объединением запросов
- Реализовать и внедрить алгоритм в систему RAIDIX ERA
- Выполнить тестирование алгоритма

Анализ производительности системы

Паттерн 1: размер блока 4k

Количество потоков

Глубина очереди	1	8	32
1	29.9	214	569.3
4	62.8	408.6	990.2
8	111.6	543.7	1241
16	134.1	664.6	1431.5
32	165.9	1016.8	2023.4

Без объединения запросов

x3

x0.5

Количество потоков

Глубина очереди	1	8	32
1	20.3	154.6	446.5
4	31.3	216.1	699.4
8	50.1	317.4	966.6
16	333.8	3818.5	6347.8
32	416.8	5065.7	6369.3

С объединением запросов

Паттерн 2: размер блока 8k

Количество потоков

Глубина очереди	1	8	32
1	38.9	280.6	643.1
4	77.8	491.5	1068
8	134.1	690.2	1285.1
16	200.7	849.9	1598.5
32	258.1	1257.5	2193

Без объединения запросов

Количество потоков

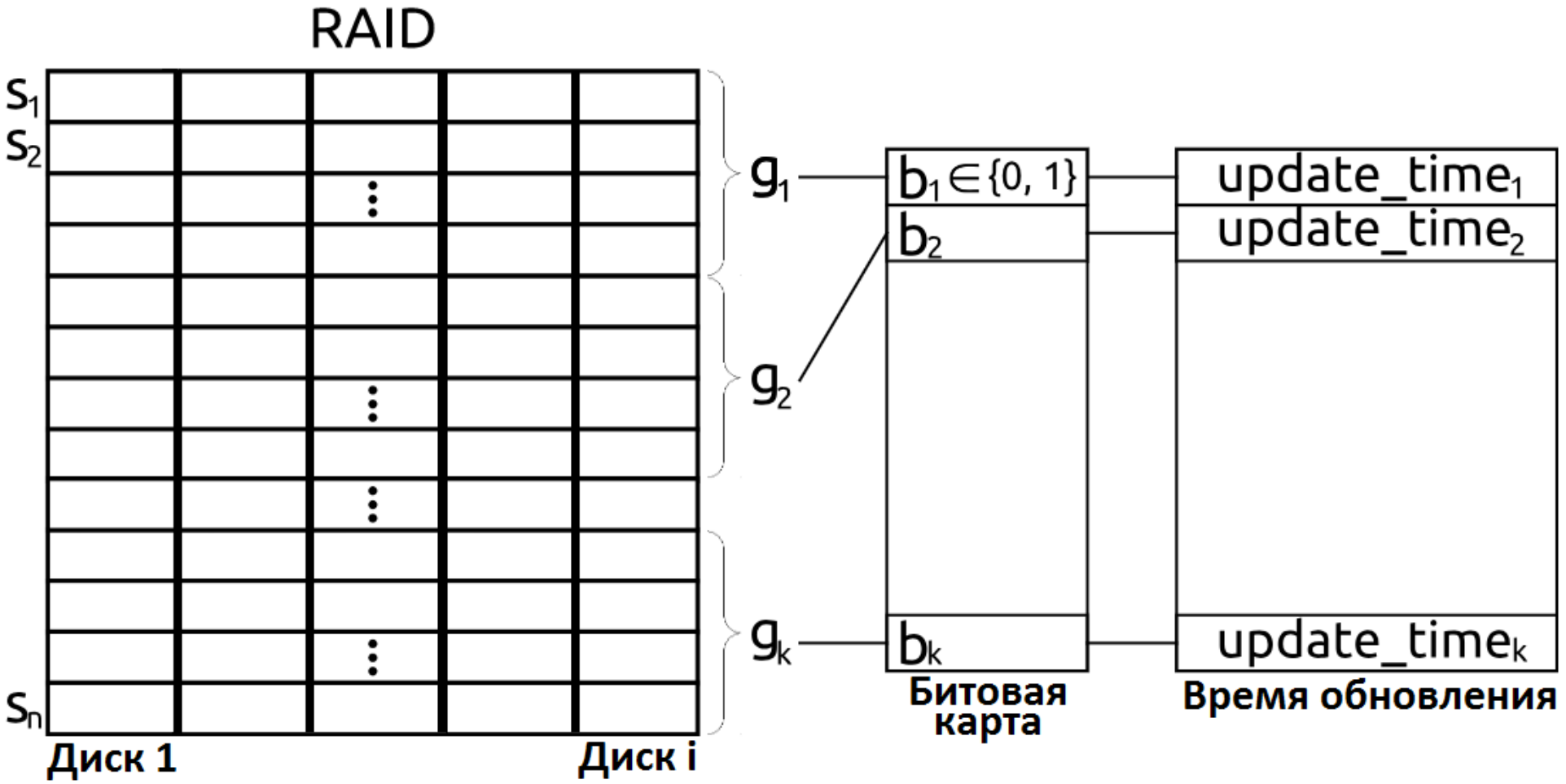
Глубина очереди	1	8	32
1	27.9	206.8	632.8
4	46.7	314.4	936.9
8	547.1	4595.7	6346.7
16	576.5	5732.3	6392.8
32	735.2	6328.3	6429.7

С объединением запросов

Условие эффективности объединиения запросов: QD*bs >= stripe data len

Алгоритм

Зонирование RAID-массива

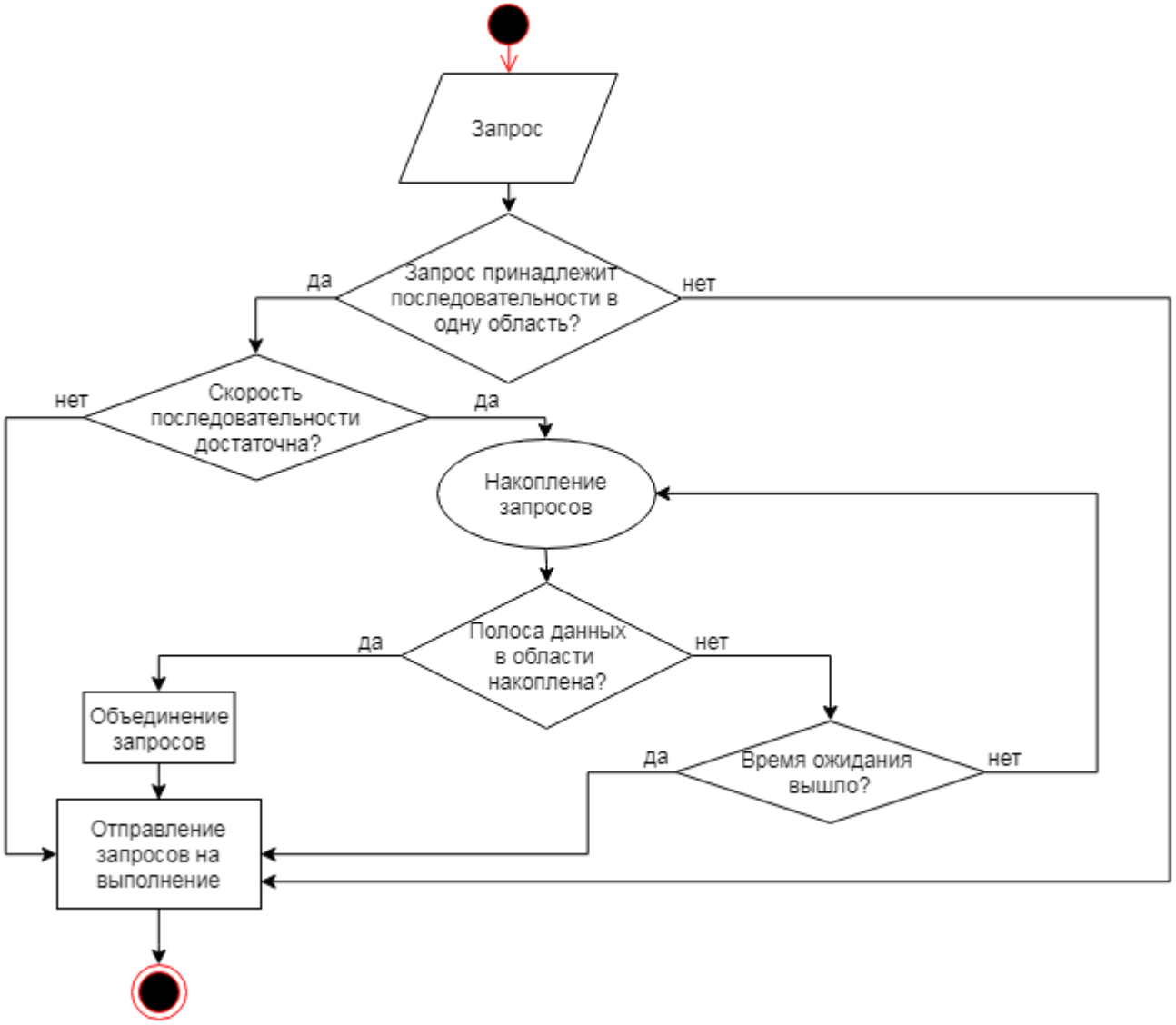


Алгоритм

Общее описание

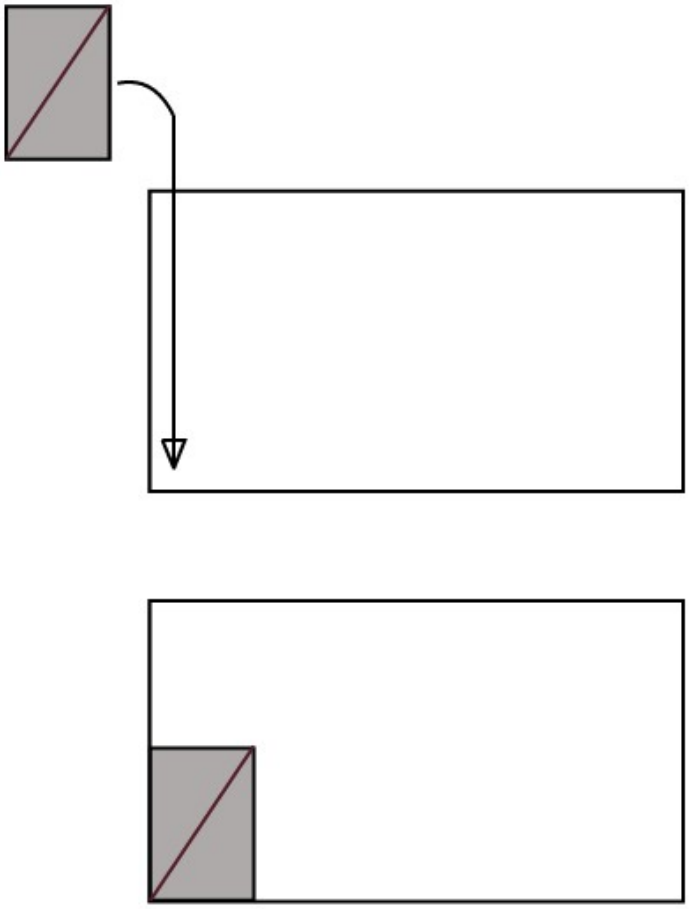
input:
- размер запроса
- тип запроса
- адрес запроса

output:
- время ожидания ($t \in [0,t_{max}]$)



Алгоритм

Детектор последовательной нагрузки



Полоса данных

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

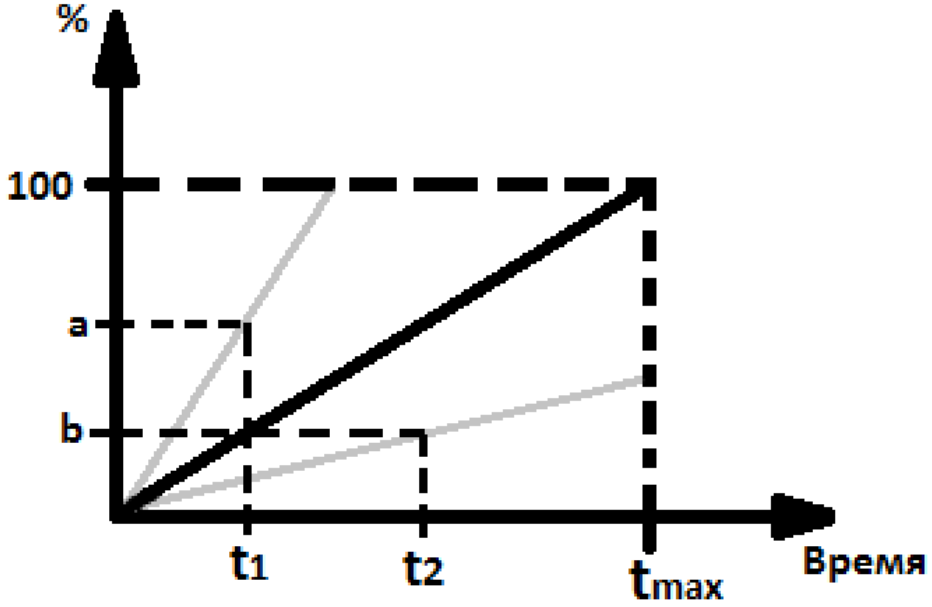
0	0	0	0	0
0	0	0	0	0
1	0	0	0	0
1	0	0	0	0

Битовая карта заполненности полосы данных

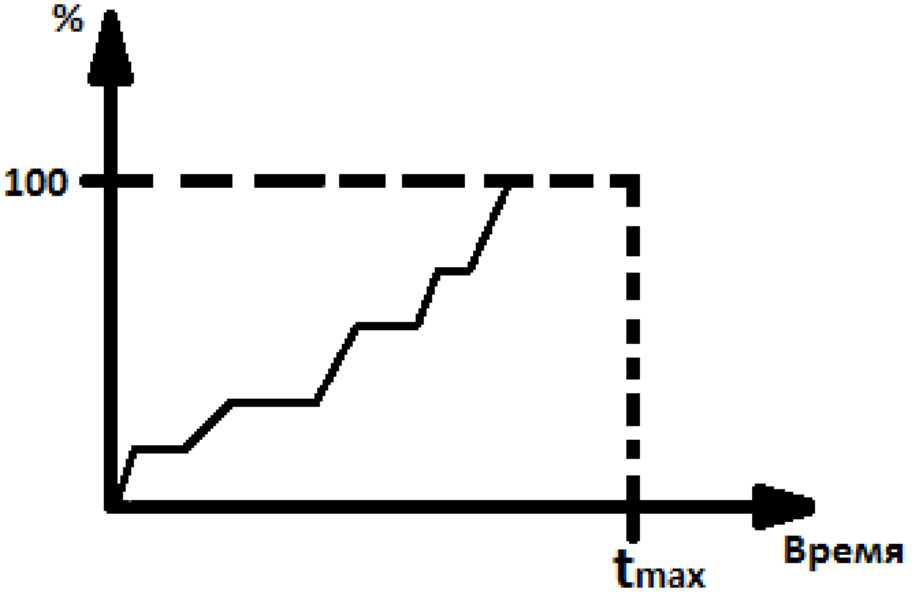
Алгоритм

Детектор скорости поступления запросов

$t \in (0, t_{max}]$



Теоретическая оценка скорости поступления запросов



Скорость поступления запросов в условиях, приближенных к реальным

Реализация

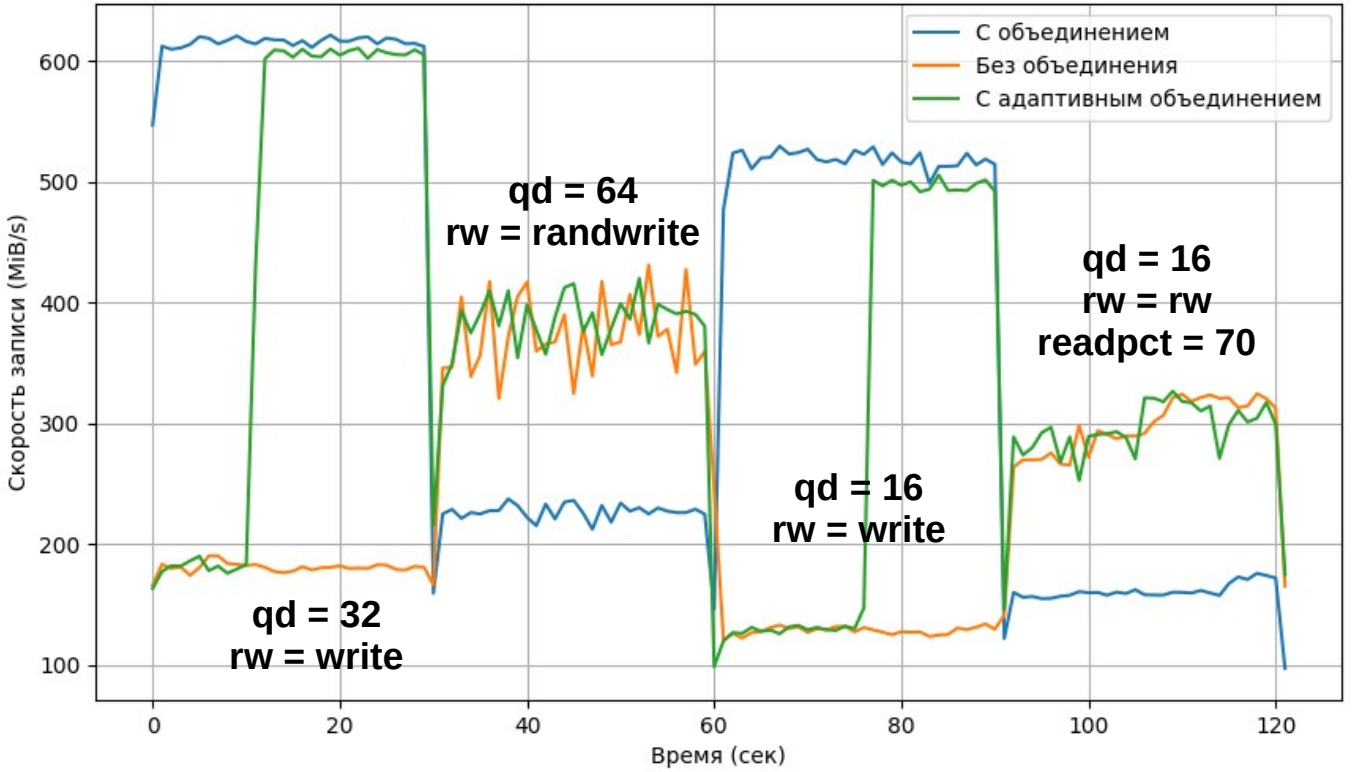
Три функции в драйвере блочного устройства Linux:

- bitmap_clear - оценивает актуальность бита в битовой карте
- merge_check() - обрабатывает запросы направленные на объединение
- seq_detect - определяет последовательную запись

Производительность алгоритма

RAID 6 6 NVMe
HGST Ultrastar SN150 HUSPR3216AHP301
Linux kernel 3.10

Тестирование при помощи FIO 3.1 в один поток



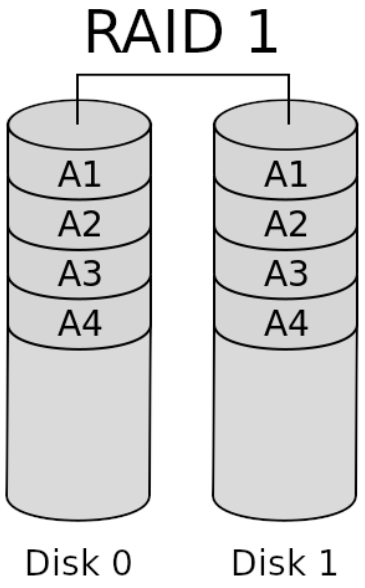
Тестирование производительности при изменяющейся нагрузке в области

Результаты

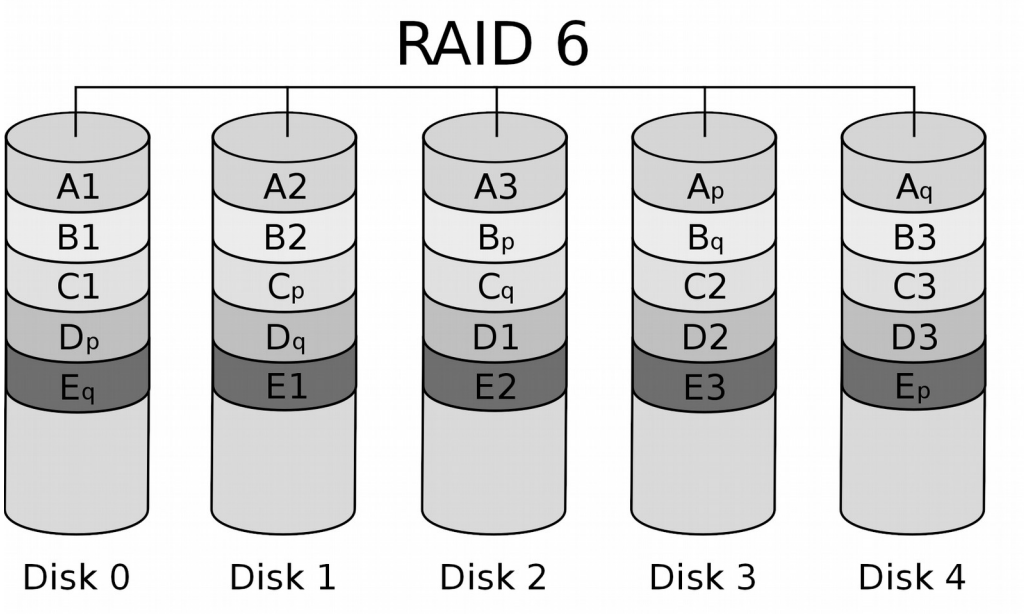
- Выполнен обзор существующих решений для обнаружения последовательных запросов в различных технологиях СХД
- Проведено сравнение работы системы RAIDIX ERA в режиме объединения последовательных запросов и без него
- Разработан алгоритм, управляющий объединением запросов. Алгоритм основывается на использовании битовых карт
- Управляющий алгоритм реализован на языке C и внедрен в систему RAIDIX ERA
- Производительность алгоритма протестирована с помощью стандартных инструментов тестирования блочных устройств FIO и Oracle VDbench

Технология RAID-массивов

С зеркалированием



С использованием контрольных сумм



Источник: https://ru.wikipedia.org/wiki/RAID

Объединение запросов

Производительность алгоритма

Тип нагрузки	Без объединения	С объединением	С адаптивным объединением
С глубиной очереди 8	**111.6**	**50.1**	**108.9**
С глубиной очереди 16	**134.1**	**338.8**	**335.1**

Производительность адаптивного объединения для последовательной записи в один поток блоком 4k (Мб/сек)