Распределенная NoSQL СУБД «Riak»

2013
CEE-SEC(R)
Разработка ПО

Рексофт: Инновации на заказ

NoSQL – Not only SQL

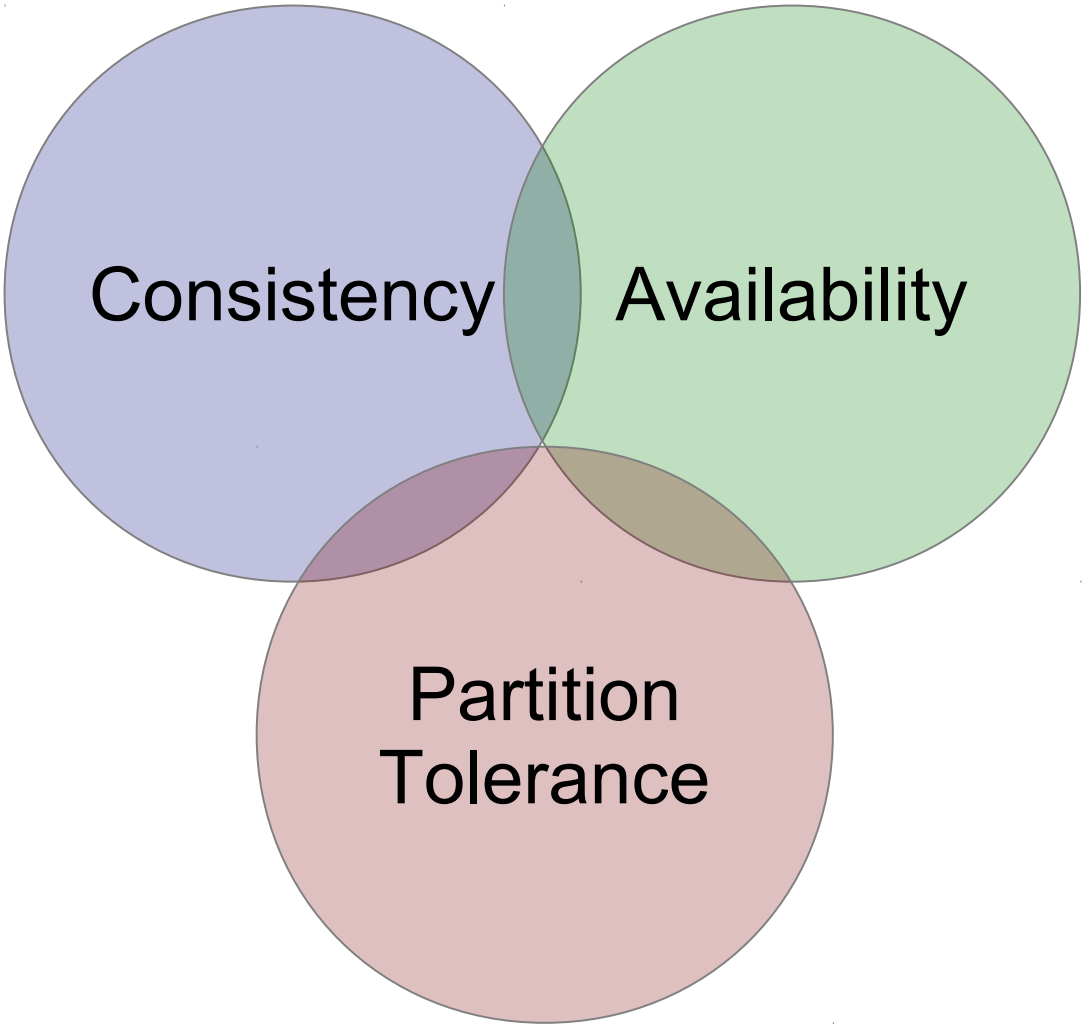
- Нереляционная модель данных
- Менее жесткая модель целостности
- Распределенность
- Горизонтальная масштабируемость

Примеры NoSQL СУБД

- HBase — колоночная http://hbase.apache.org
- Riak — документная http://basho.com/riak/
- Redis — ключ-значение http://redis.io
- Neo4j — графовая http://neo4j.org

CAP-теорема

ВЫБЕРИ ДВА

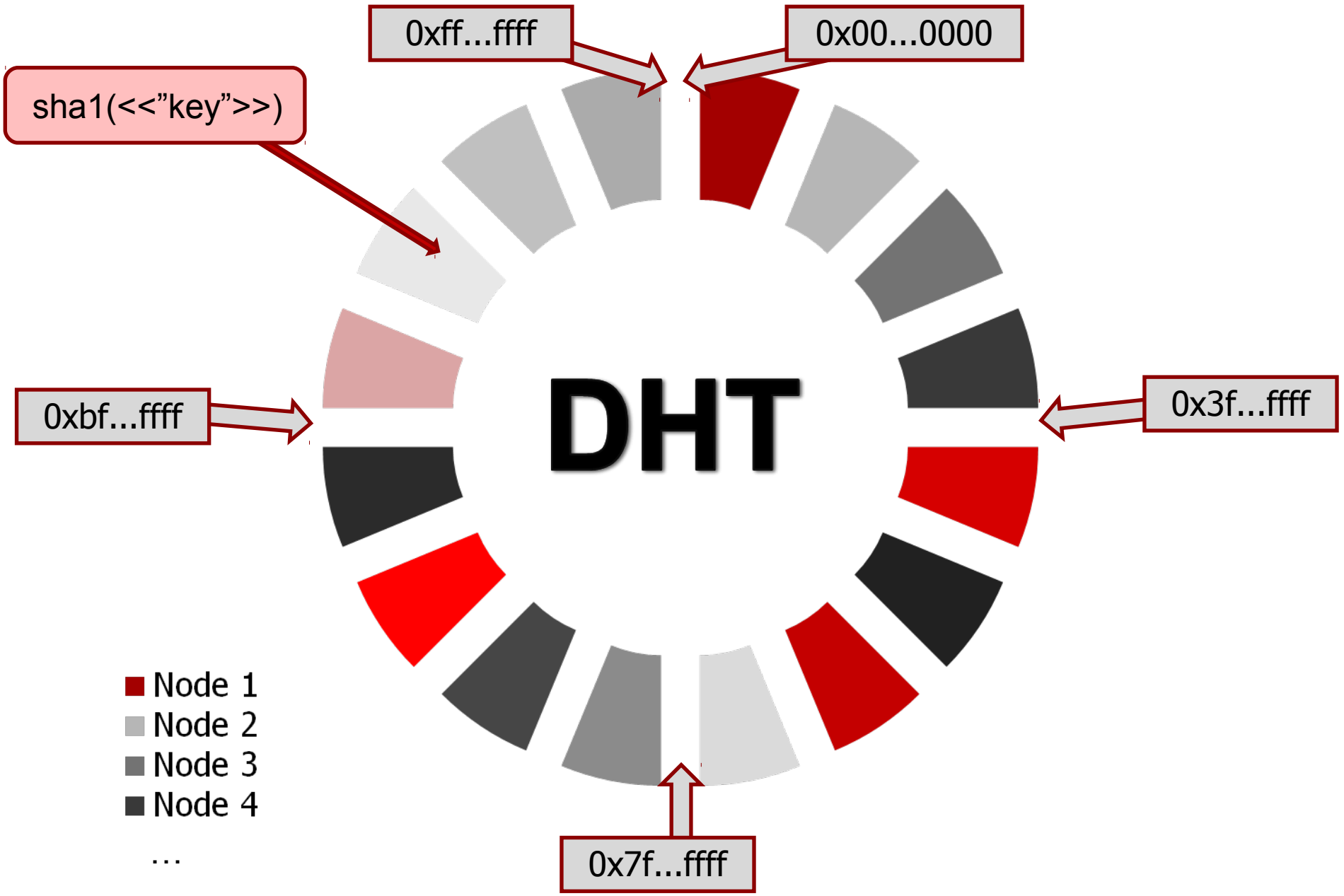


http://basho.com/riak/

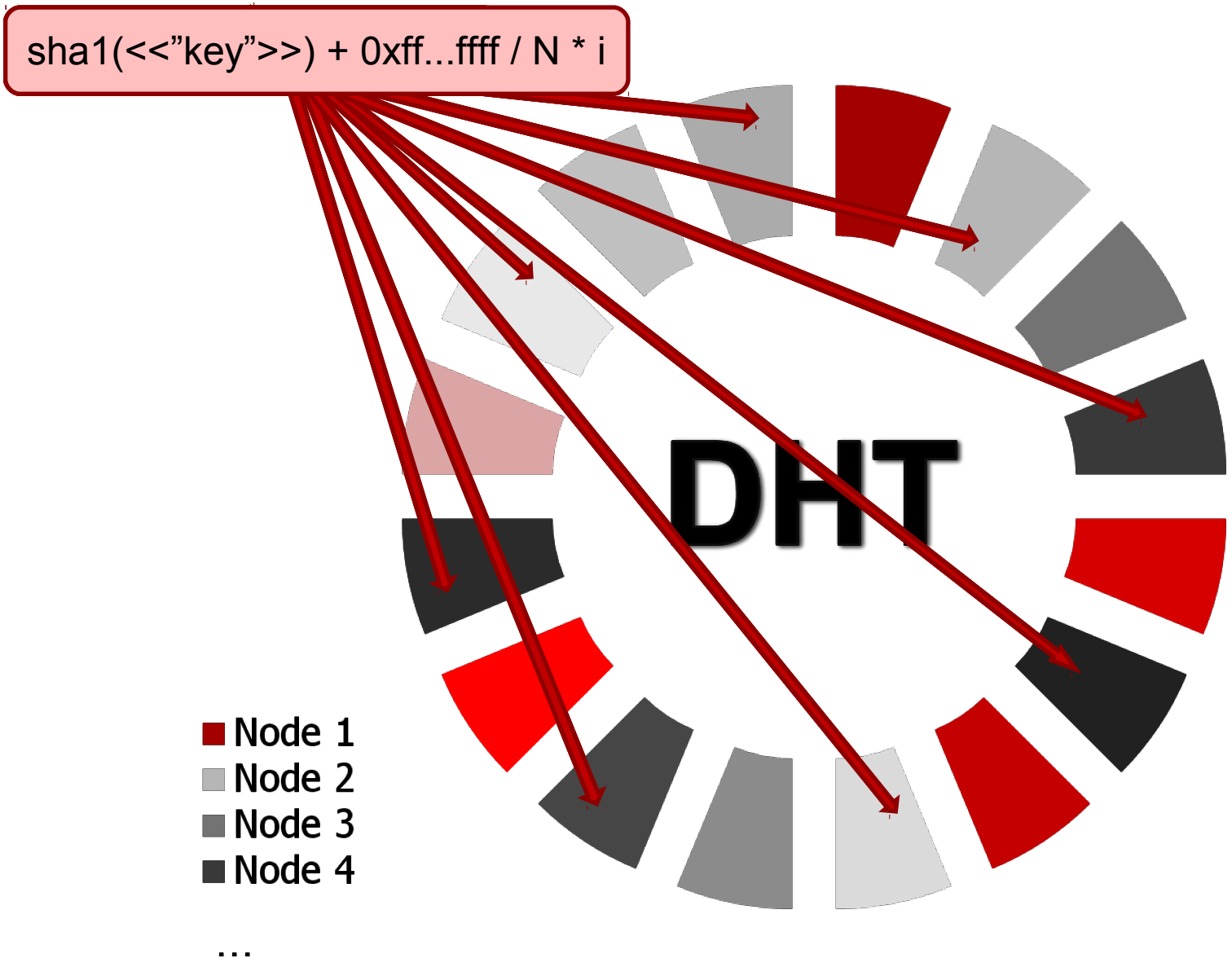
Ключевые особенности

- Изменяемая в run-time избыточность
- Гомогенность
- Отказоустойчивость наследованная от Erlang
- Простота развертывания
- Организация данных в bucket'ы
- Наличие ссылок

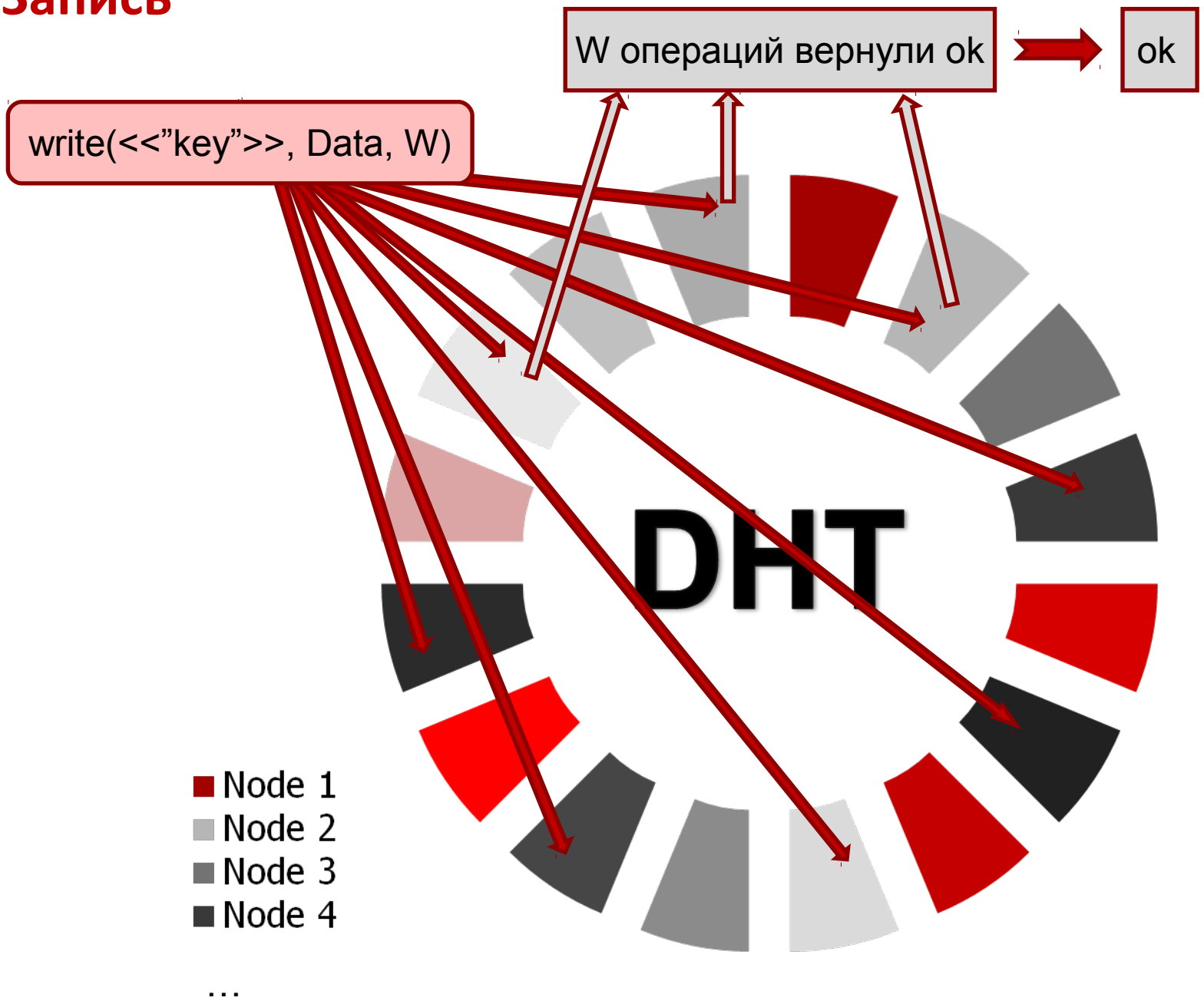
Хеш-кольцо


0xff...ffff
0x00...0000
sha1(<<"key">>)
0xbf...ffff
DHT
0x3f...ffff
Node 1
Node 2
Node 3
Node 4
…
0x7f...ffff

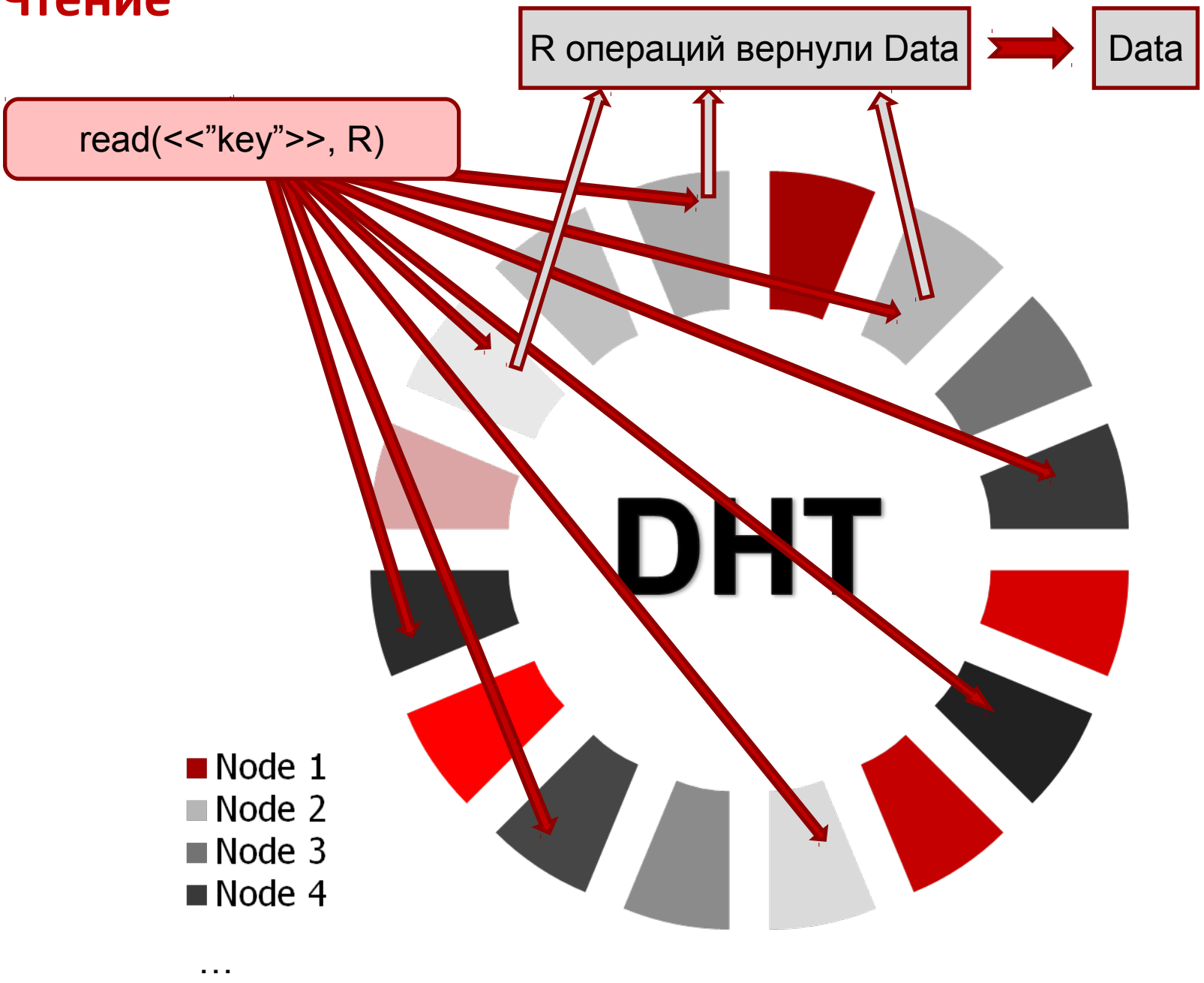
Избыточность



Запись


W операций вернули ok
ok
write(<<”key”>>, Data, W)
DHT
Node 1
Node 2
Node 3
Node 4
…

Чтение

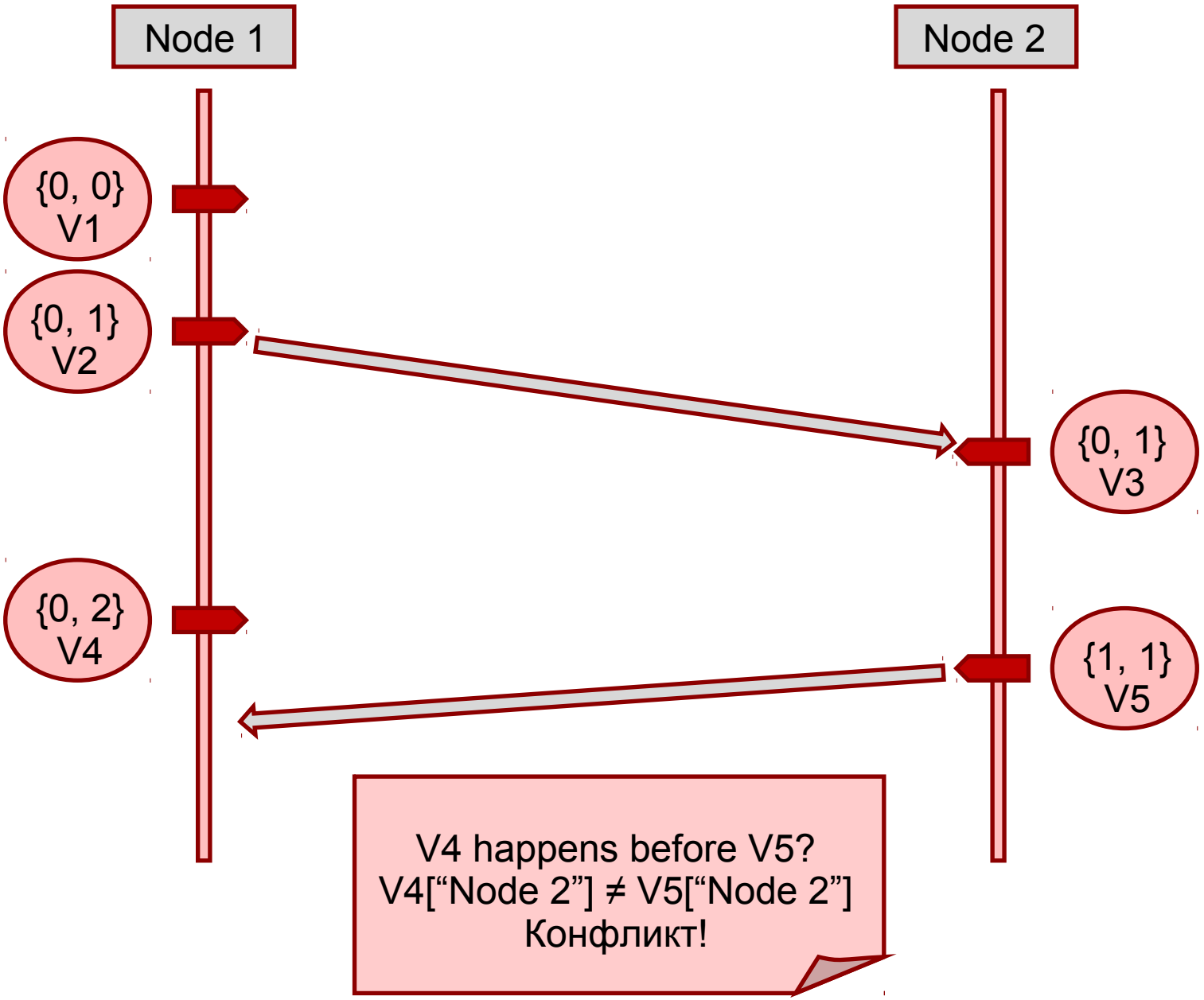

R операций вернули Data
Data
read(<<”key”>>, R)
DHT
Node 1
Node 2
Node 3
Node 4
…

Золотое правило

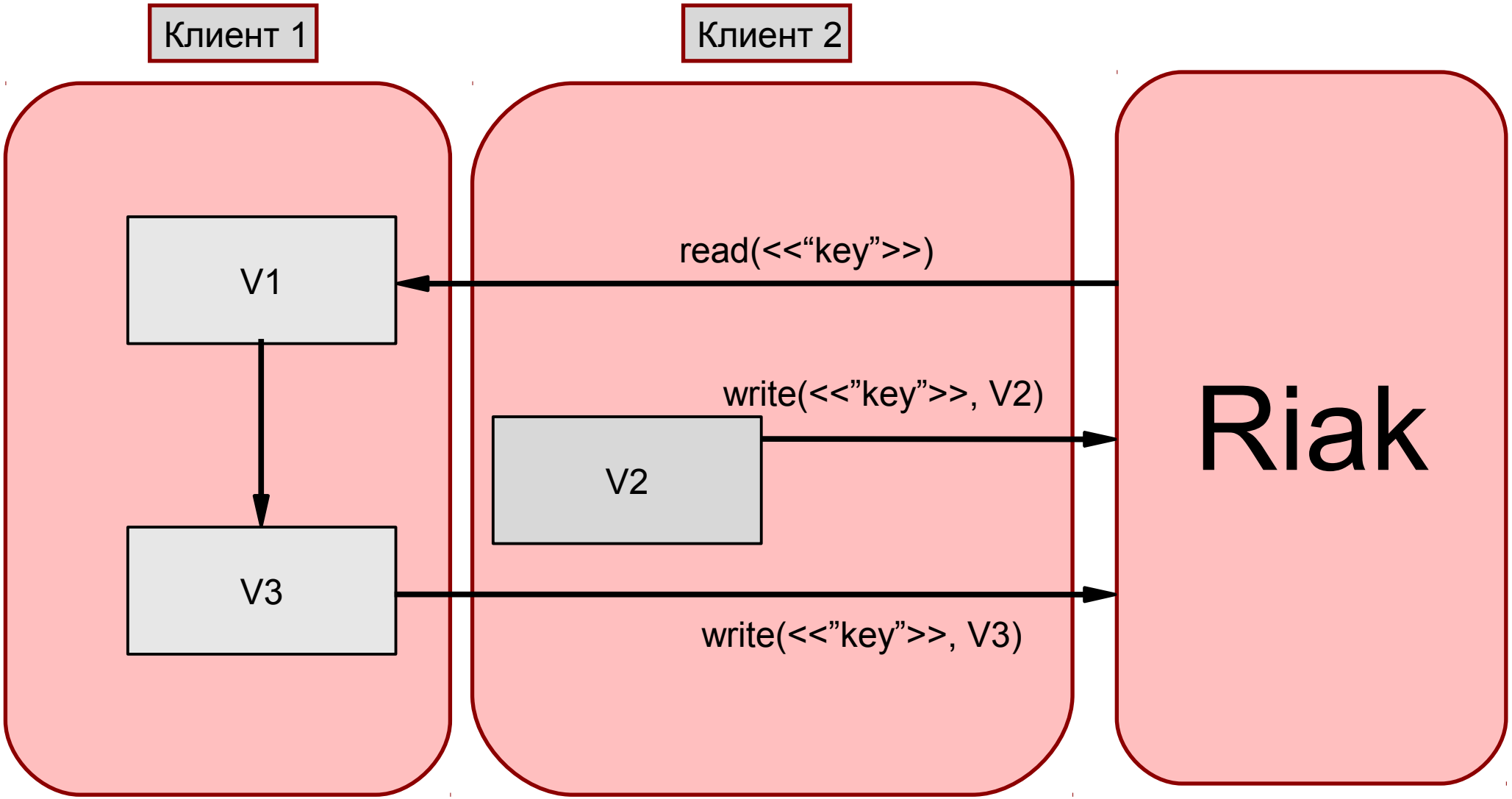
Reksoft

$$R + W > N$$

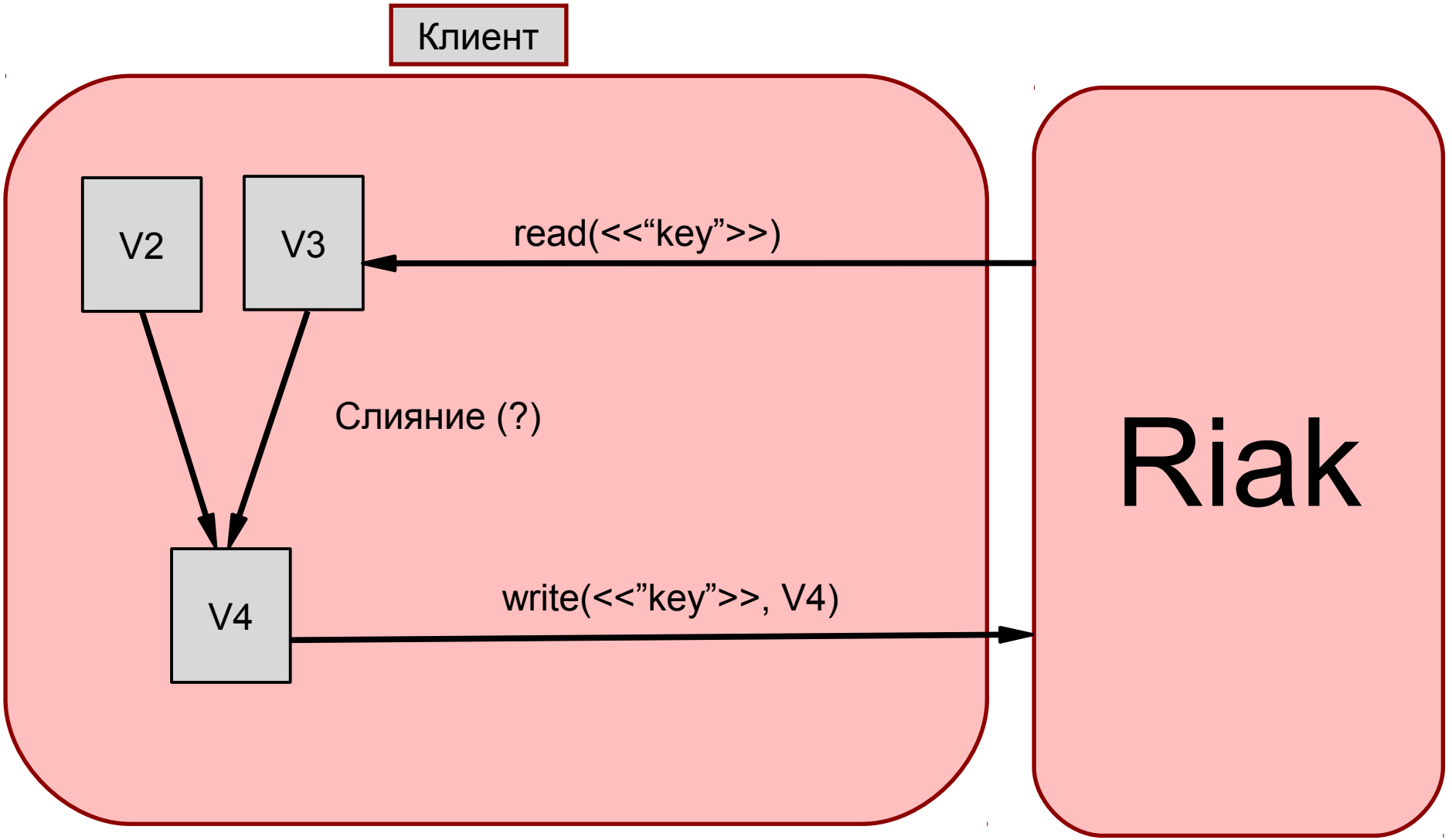
Векторные часы



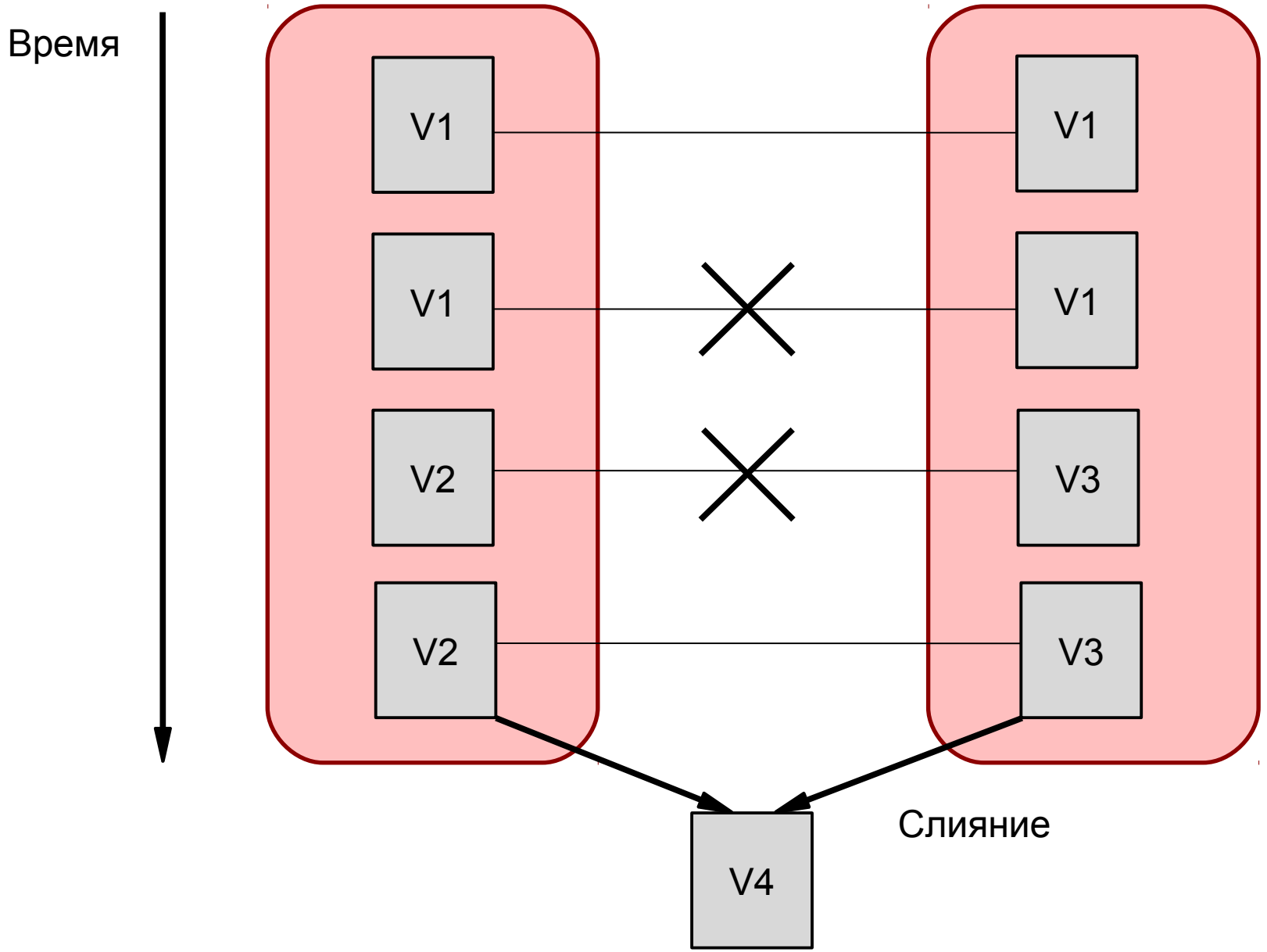
Разрешение конфликтов



Разрешение конфликтов



Устойчивость к разделению



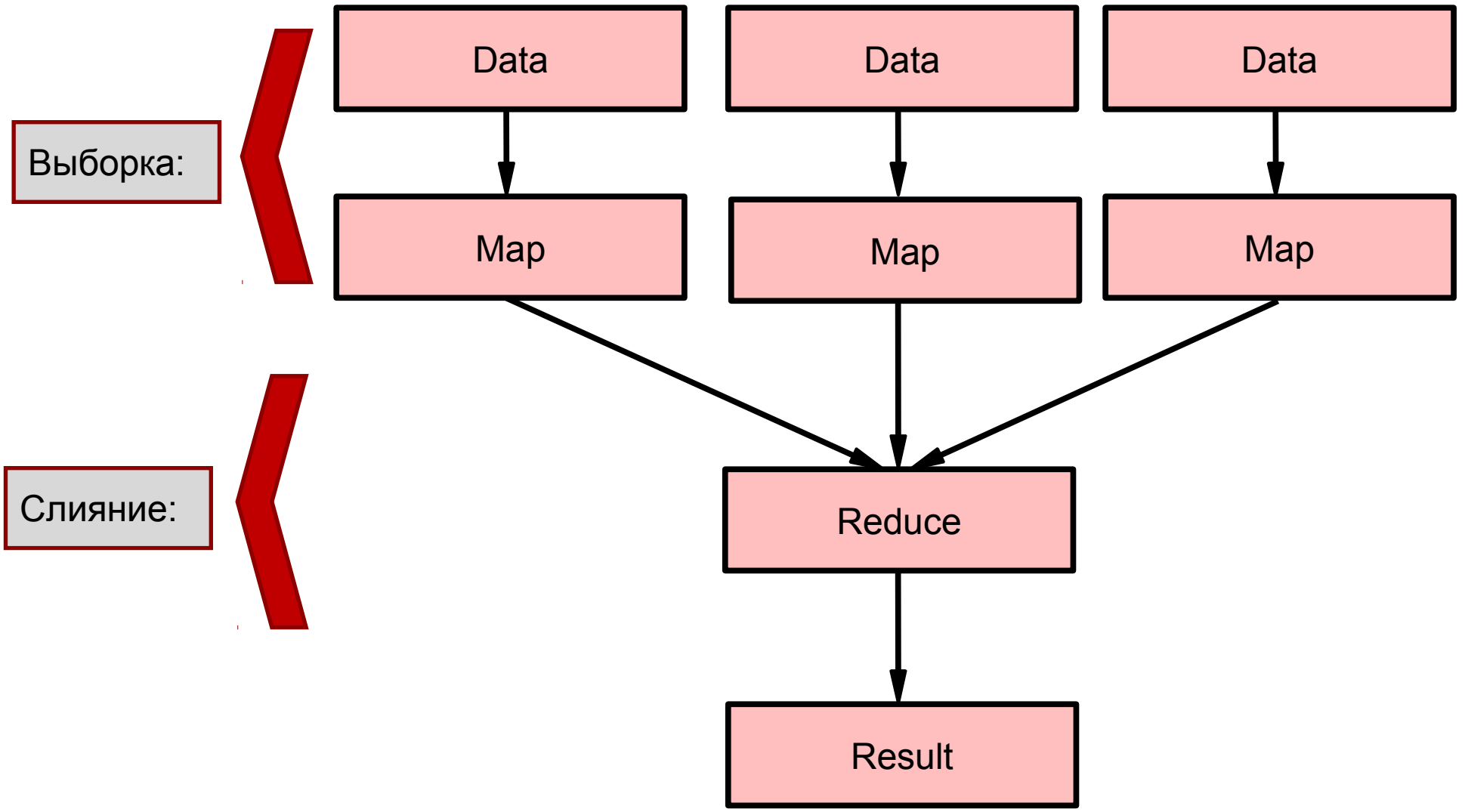
Выборка

- Первичный ключ
- Вторичные индексы
- Riak search engine
- MapReduce

Riak Search Engine

- Полнотекстовый поиск
- Простой язык запросов (Lucene-based)
- Оценка релевантности результатов
- Результаты поиска могут быть входными данными для MapReduce

MapReduce



Клиентская часть

- General HTTP RESTful API
- Google protocol buffers API
- Erlang (native)

Клиентские библиотеки для:
Java, Perl, Python, PHP, .NET, C++ и др.

Клиентская часть: Erlang

```
{ok, Conn} = riak:client_connect('riak@192.168.88.3'),
Obj = riak_object:new(<<"bucket">>, <<"key">>,
                      [{int_data_field, 1},
                       {array_data_field, [1,2,3]}]),
ok = Conn:put(Obj, 2)

{ok, Obj1} = Conn:get(<<"bucket">>, <<"key">>, 2),

Upd = riak_object:update_value(Obj1, [{int_data_field, 42}]),
ok = Conn:put(Upd, 2).
```

Клиентская часть: RESTful

```
curl -X PUT http://127.0.0.1:8098/jiak/bucket \
       -H "Content-Type: application/json" \
       --data "{\"schema\":{\"allowed_fields\":
    [\"int_data_field\", \"array_data_field\"], \"write_mask\":
    [\"int_data_field\", \"array_data_field\"]}}"

curl -X PUT http://127.0.0.1:8098/jiak/bucket/key \
       -H "Content-Type: application/json" \
       --data "{\"bucket\":\"bucket\", \"key\":\"key\",
\"object\":{\"int_data_field\":1,\"array_data_field\":[1,2,3]},
\"links\":[]}"

curl http://127.0.0.1:8098/jiak/bucket/key
```

Эффективность

Basho Bench — утилита для замера производительности кластера Riak

http://docs.basho.com/riak/latest/ops/building/benchmarking/

Эффективность: масштабируемость

Условия тестирования:

Размер записи — 10Kb

Нагрузка — максимум (около 200 операций/узел)

Настройки избыточности — по-умолчанию

	Read	Write	Update
2 узла	6 ms	25 ms	32 ms
10 узлов	8 ms	47 ms	67 ms
20 узлов	7 ms	57 ms	80 ms

Эффективность: client-side

Условия тестирования:

Размер записи — 10Kb

Количество узлов в кластере — 10

Нагрузка — максимум (около 200 операций/узел)

Настройки избыточности — по-умолчанию

	Read	Write	Update
HTTP	97 ms	151 ms	146 ms
Protobuf	8 ms	47 ms	67 ms

Сложные запросы

Дано: новостные данные

Необходимо: посчитать популярность тегов, полученых на основе языкового анализа за указанный период

Сложные запросы: предвыборка

```
{
  “inputs”: {
              “bucket”: “news”,
              “query”: “time: ['2013/10/01 00:00' to
                                '2013/10/31 23:59']”
            }
  “query”: […]
}
```

Сложные запросы: фаза map

```
“query”: [{“map”: {“language”: ”javascript”,
    “source”:
    ”function(v) {
        var tags = doNLProcess(v.values[0].data)
        // tags format: {“tag1”: 0.97, “tag2”: 0.5}
        return [tags]
    }”}},
…]
```

Сложные запросы: фаза reduce

```
"query": [..., {"reduce": {"language": "javascript",
    "source":
    "function(v) {
        var result = {}
        for(var i in v)
          for(var j in v[i])
            result[j] += v[i][j]
        return [result];
    }"}}]
```

Способы разрешения конфликтов

- Необходимое условие целостности: идемпотентность операции изменения
- Основная рекомендация: добавляйте данные, а не изменяйте их!
- Неидемпотентное изменение должно быть проделано только одним клиентом над всеми данными

Автоматическое разрешение конфликтов

Statebox — фреймворк для построения структур данных поверх БД с нежесткой моделью целостности

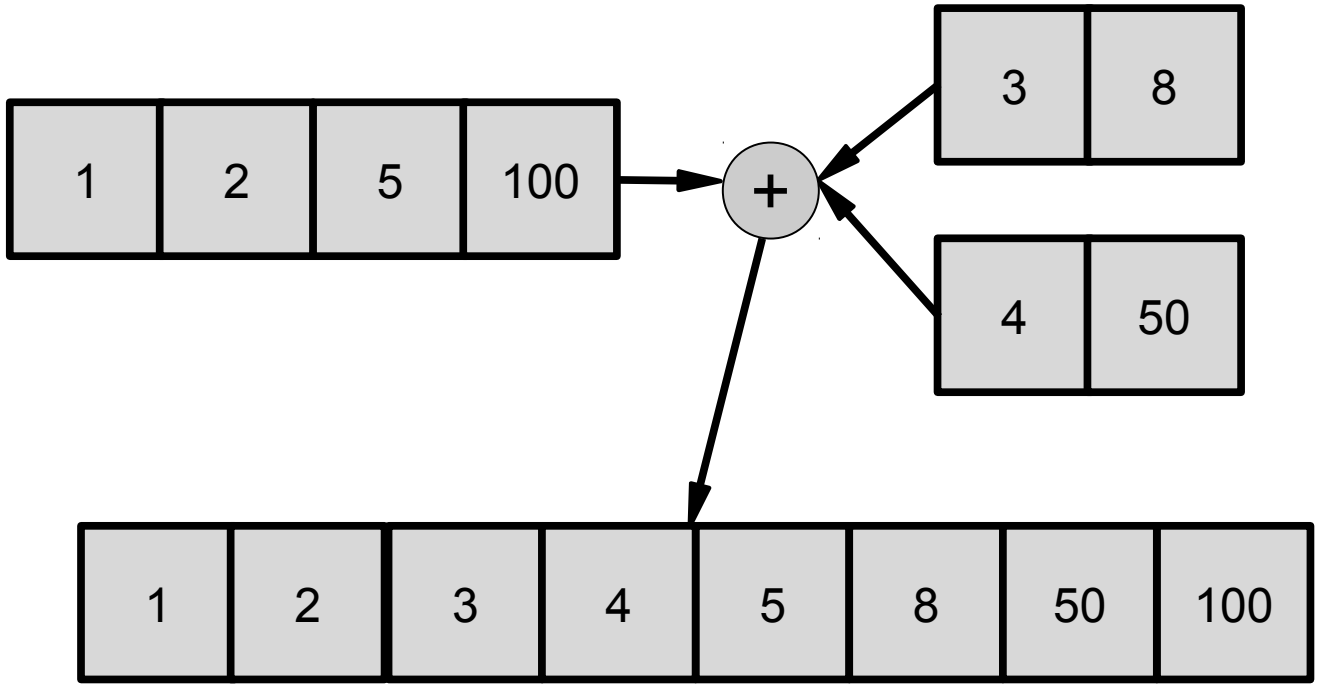
https://github.com/mochi/statebox

Целостные данные в Riak

- Константные значения
- Упорядоченные списки
- Множества
- Упорядоченный enum

Упорядоченные списки/множества

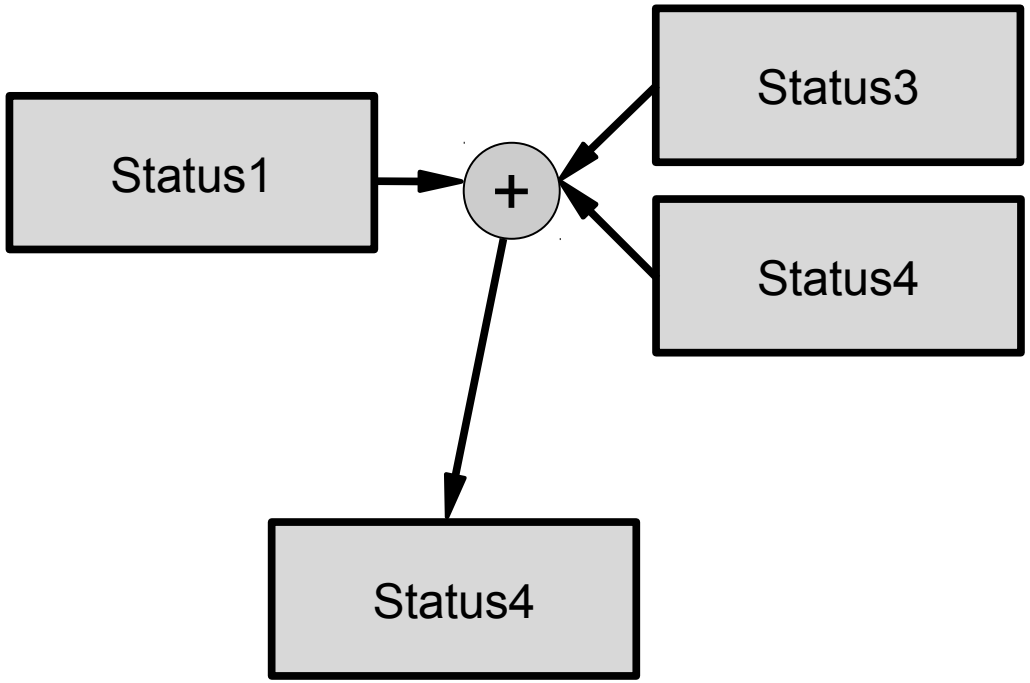
Операция слияния: объединение множеств



Упорядоченный enum

Операция слияния: max

Status1 < Status2 < Status3 < Status4 < Status5



Выводы

Riak стоит использовать, если:

- Много данных
- Модель данных определена на этапе проектирования
- Модель данных проста и не изобилует связями
- Большинство операций — операции добавления, удаления или чтения, но не изменения

Спасибо за внимание!

Андрей Смирнов

asmirnov@reksoft.com

Санкт-Петербург, Россия
Тел.: +7 812 325 2100
www.reksoft.com/ru