Bitmap Index: ускоряем поиск

- Некрасов Дмитрий
- C# разработчик в ЦИАН
  (cian.ru - база недвижимости)

- telegram: @Aberkromb
- Github: github.com/aberkromb

HLCup2018

- 2 GB RAM
- Архив с данными в разжатом виде ~ 1.3 GB
- Нужно быстро совершать поиск по коллекции из 1300000 пользователей и сопутствующих данных

HLCup2018

- Простой поиск по аккаунтам - O(N)
  (место жительства, пол, возраст и т.д.)

- Поиск по множествам среди аккаунтов - O(N^2)
  (лайки, интересы)

Идеи?

Нужны индексы в памяти!

Нужны индексы в памяти!

- Словари?

```
var indexes = new Dictionary<string, int[]>();
indexes.Add("women", new int[100]);
indexes.Add("blue_eyes", new int[200]);
indexes.Add("women_with_blue_eyes", new int [/*???*/]);
```

Нужны индексы в памяти!

- Словари?

```
var indexes = new Dictionary<string, int[]>();
indexes.Add("women", new int[100]);
indexes.Add("blue_eyes", new int[200]);
indexes.Add("women_with_blue_eyes", new int [/*???*/]);
```

  - Неудобно перестраивать.
  - Дублирование информации в массивах
  - Сложно объединить 2 признака в 1
  - Занимают много места

Нужны индексы в памяти!

- bool[] ?
  + почти то что надо
  + можно быстро и просто объединять признаки
  - заняли места больше чем словари

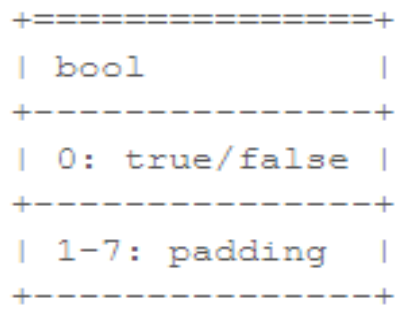
Нужны индексы в памяти!

- bool[] ?
  + почти то что надо
  + можно быстро и просто объединять признаки
  - заняли места больше чем словари

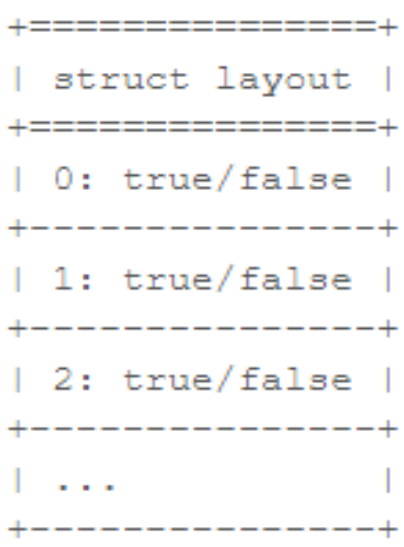
- ЧЯДНТ?

Нужны индексы в памяти!

bool

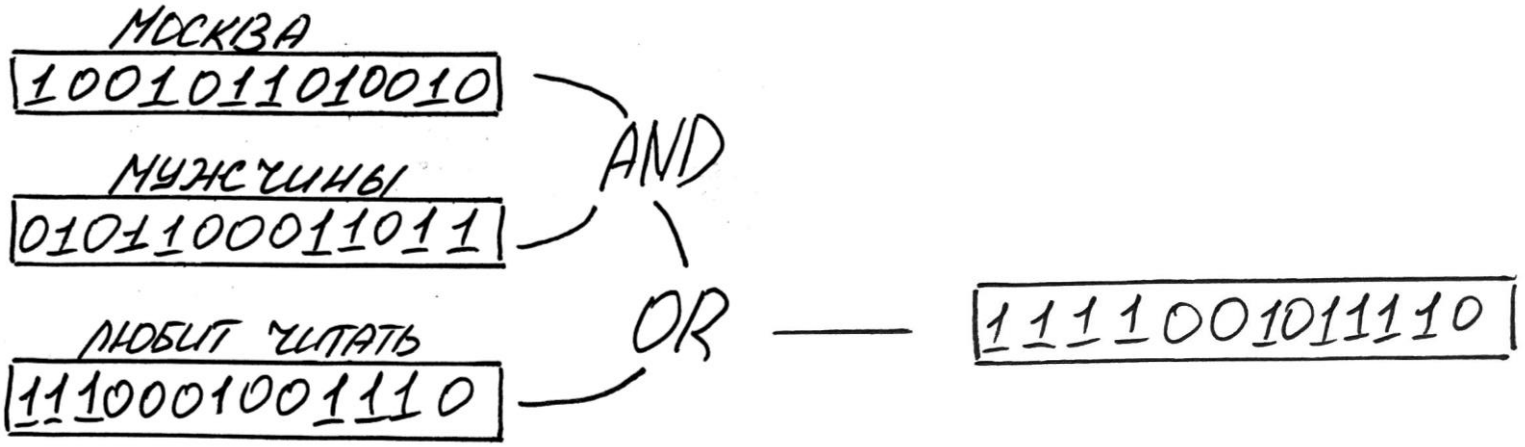



```
[StructLayout(LayoutKind.Explicit)]
public struct EightBits
{
    [FieldOffset(0)]
    public bool First;
    [FieldOffset(1)]
    public bool Second;
    [FieldOffset(3)]
    public bool Third;
    /*...*/
}
```



Что же такое Bitmap index?

- Компактная структура данных
- Одна битовая карта - один признак



Где встречаются?

Где встречаются?

- Oracle DB
- Tarantool
- Postgres (под капотом)
- Elasticsearch использует Roaring Bitmap (прокаченные Bitmap от Daniel Lemire - есть блог)
- Pilosa (новая БД на Go полностью на bitmap)

Реализация

Реализация

- Int32/int64 - самые эффективные массивы
- Int64[] - массив массивов
- Звучит просто... но!

Реализация

- Чтобы выставить бит по индексу

```
public void Set(int index, bool value)
{
    var bitmapIndex = index / BitsPerLong;
    var bit = index % BitsPerLong;

    var newValue = _mapArray[bitmapIndex];

    if (value)
        newValue |= 1L << bit;
    else
        newValue &= ~(1L << bit);

    _mapArray[bitmapIndex] = newValue;

    _bitsCount = -1;
}
```

- Посчитать длину массива

```
public static int GetArrayLength(int n) =>
    (int) ((uint) (n - 1 + (1L << BitShiftPerInt64)) >> BitShiftPerInt64);
```

Реализация

- Что по памяти?
  - Один индекс занимает (N/64) * 8 (bytes)

Для 1000 элементов получится 16 * 8 = 128 bytes

Реализация

- Модный Fluent интерфейс

```
_defaultBitMapIndex = new BitmapBuilder<MessageData>()
    .IndexFor("IsPersistent", msg => msg.Persistent)
    .IndexFor("ServerIsFrederik", msg => msg.Server.Equals("Frederik", StringComparison.Ordinal))
    .ForData(_data)
    .Build();
```

- Запрос

```
var query = _defaultBitMapIndex.NewQuery
    .Where("IsPersistent")
    .And("ServerIsVickie")
    .And("ApplicationIsTrantow");

var result = query.Execute();
```

Реализация

- А как быстро это работает? (Для O(n))

Method	Mean	Error	StdDev	Gen 0	Gen 1	Gen 2	Allocated
Iterate	5,891.07 us	72.3925 us	67.7160 us	-	-	-	32 B
DefaultBitMap	18.09 us	0.1953 us	0.1631 us	5.9814	-	-	12560 B

Реализация

- А быстрее можно?

Реализация

- А быстрее можешь?
  - SIMD? (single instruction, multiple data)
  - Нужно закреплять массивы (fixed())
  - В Core 3 из коробки

Реализация (SIMD)

```
public unsafe IBitmap Xor(IBitmap other)
{
    var valueArray = other.AsSpan();

    fixed (long* v = &valueArray[0])
    fixed (long* l = &_mapArray[0])
    {
        for (var i = 0; i <= _mapArray.Length; i += VecSize)
        {
            var left = Avx.LoadVector256(l + i);
            var right = Avx.LoadVector256(v + i);
            var resultVector = Avx2.Xor(left, right);

            Avx.Store(l + i, resultVector);
        }
    }

    return this;
}
```

Реализация (SIMD)

- А как быстро это работает? (для O(n))

Method	Mean	Error	StdDev	Gen 0	Gen 1	Gen 2	Allocated
Iterate	5,891.07 us	72.3925 us	67.7160 us	-	-	-	32 B
DefaultBitMap	18.09 us	0.1953 us	0.1631 us	5.9814	-	-	12560 B
Avx2BitMap	11.82 us	0.0801 us	0.0710 us	5.9814	-	-	12560 B

  - p.s. можно использовать AVX512

Какие проблемы?

- Высокая кардинальность
  - Кардинальность - это мощность множества значений
  - можно группировать значения
- Проблема обновлений индекса
  - Перестроить в фоновой задаче и подменить
  - Разбить на блоки/корзины/группы и делать lock на отдельные

Ресурсы

- Моя реализация на C#
  https://github.com/aberkromb/Crude.BitmapIndex
- Пример в corefx
  https://egorbo.com/llvm-range-checks.html
- Доклад с Go конференции
  https://habr.com/ru/company/badoo/blog/451938/
- Расширение темы
  https://habr.com/ru/company/mailru/blog/479822/

О чем я говорил

- Индексы не только в БД, но и в памяти
- Могут помочь ускорить некоторые задачи
- Существует такая компактная СД - bitmap
- Ее можно легко ускорить за счет SIMD
  - Без SIMD тоже быстро работает
- Битовые карты используются в разных БД уже давно

Q/A

Некрасов Дмитрий

- telegram: @Aberkromb
- Github: github.com/aberkromb