



ТЕХНОСФЕРА

Спайдер. Сегментация сайта.

Воронов Филипп
программист отдела ранжирования

Поиск@Mail.Ru

Поисковый спайдер



Проблема:

Сайтов много

Страниц еще больше

Времени мало

Спайдер

1. Постановка задачи
2. Выкачка
3. Обновление
4. Хранение

Требования к спайдеру

1. Politeness
2. Freshness
3. Actuality
4. Производительность
5. Масштабируемость

URL

RFC: <https://www.ietf.org/rfc/rfc1738.txt>

<http://site.ru/path?page=10>

http - схема

site.ru - хост

path - путь

page=10 - query

IP

Уникальный адрес сетевого узла

```
$ host go.mail.ru
```

```
$ host ru.wikipedia.org
```

DNS

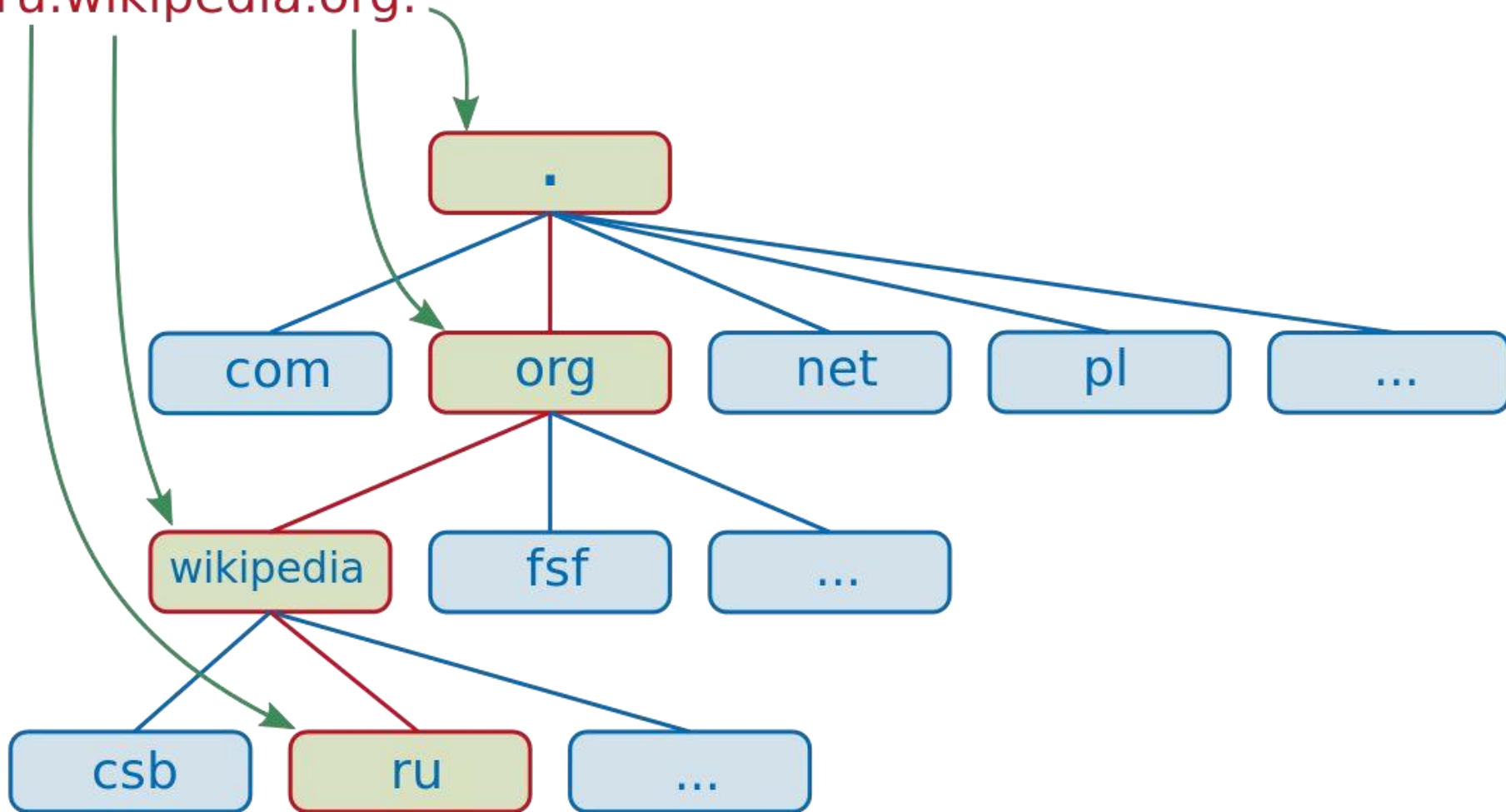
DNS - сервис для получения информации о доменах. Нам нужна информация об IP.

url -> ip

информация предоставляется иерархической системой серверов - может быть долго

DNS

ru.wikipedia.org.



Сколько ip-адресов у сайта?

1. 1-1:

```
$ host -v -t A zonova.xyz
```

2. 1-n: снижение нагрузки (для высоконагруженных систем)

```
$ host -v -t A go.mail.ru
```

3. m-1: снижение стоимости

```
$ host -v -t A catalogr.ru
```

```
$ host -v -t A redbook73.ru
```

Robots.txt

```
User-agent: *  
Crawl-delay: 50  
Disallow: /admin  
Allow: /article
```

Хорошие роботсы:

<http://lenta.ru/robots.txt>

Плохие роботсы:

<https://money.yandex.ru/robots.txt>

Robots.txt

```
User-agent: *  
Crawl-delay: 50  
Disallow: /admin  
Allow: /article
```

Какие из этих документов
можно качать?

<http://site.ru/>

<http://site.ru/admin>

<http://site.ru/admin/article>

<http://site.ru/article/admin>

<http://site.ru/post>

Robots.txt

```
User-agent: *  
Crawl-delay: 50  
Disallow: /admin  
Allow: /article
```

Какие из этих документов
можно качать?

<http://site.ru/>

<http://site.ru/admin>

<http://site.ru/admin/article>

<http://site.ru/article/admin>

<http://site.ru/post>

Спайдер

1. Постановка задачи
2. Выкачка
3. Обновление
4. Хранение

Алгоритм

1. "Точка входа" - seed-урлы
2. Скачали
3. Распарсили, извлекли урлы, отправили урлы в очередь на обкачку
4. goto #2

Seed-урлы

КАТАЛОГ@mail.ru®

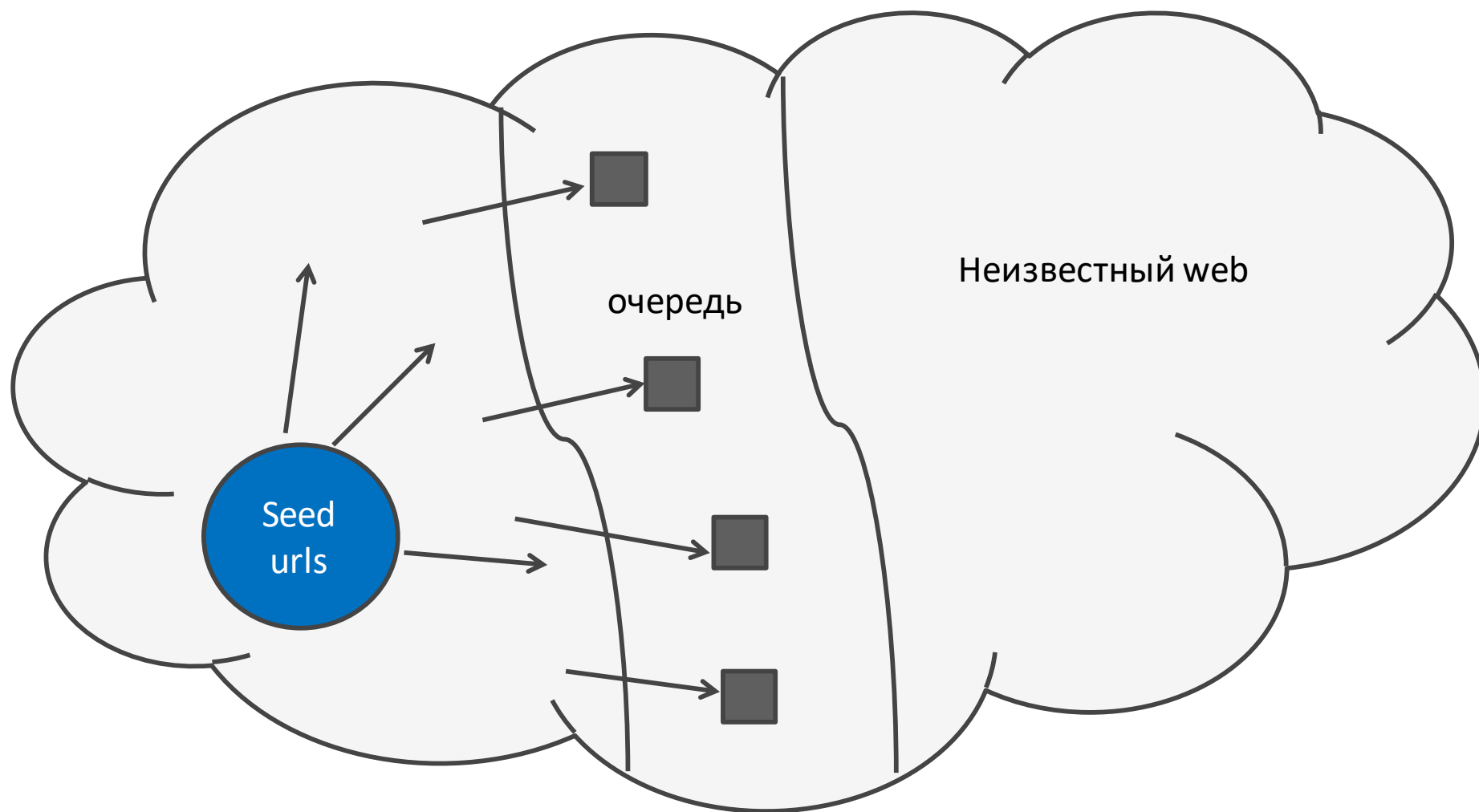
Яндекс
каталог

РЕЙТИНГ@mail.ru



Википедия
Свободная энциклопедия

Выкачка



Ответы сервера

Какие бывают?

2xx - успешно

3xx - перенаправление

4xx - ошибка клиента

5xx - ошибка сервера

Особенности контента

1. Тип контента (ајах?)
2. Кодировка

Тип контента

html, jpeg, pdf, xml, mp3 и т.д.

Как определить:

1. Content-Type: text/html
2. По первым символам контента

```
1 <!DOCTYPE html>  
2 <html>  
3 <head>
```

Не всё так просто:

<http://kiev-ehudi.org.ua/>

Какая кодировка?

Не надо быть умнее браузера.

1. Content-type: charset в http-head
2. Meta-charset

Извлечение ссылок (discovering)

``

Помним о politeness:

`<meta name="robots" content="nofollow" />`

`Войти`

Извлечение ссылок (discovering)

Ссылки бывают:

1. Внутренние и внешние
2. Абсолютные и относительные
3. Валидные и невалидные

Абсолютные и относительные ссылки

<http://site.ru/page/1>

`` --> <http://site.ru/page/2>

`` --> <http://site.ru/2>

`` --> <http://site.ru/d3>

`` --> <http://site.com/page>

`` --> <http://abc.org/g>

Нельзя брать все ссылки

1. Robots.txt
2. Некоторые документы мы уже качали
3. Внутренний blacklist:
 1. Правильные ограничения: <http://go.mail.ru/robots.txt>
 2. <https://www.iconfinder.com/search/?q=search>

А еще сайты могут быть "бесконечными":

<http://www.calend.ru/day/1-2-2050/>

Что брать и сколько?

Решает внешняя задача - scheduler

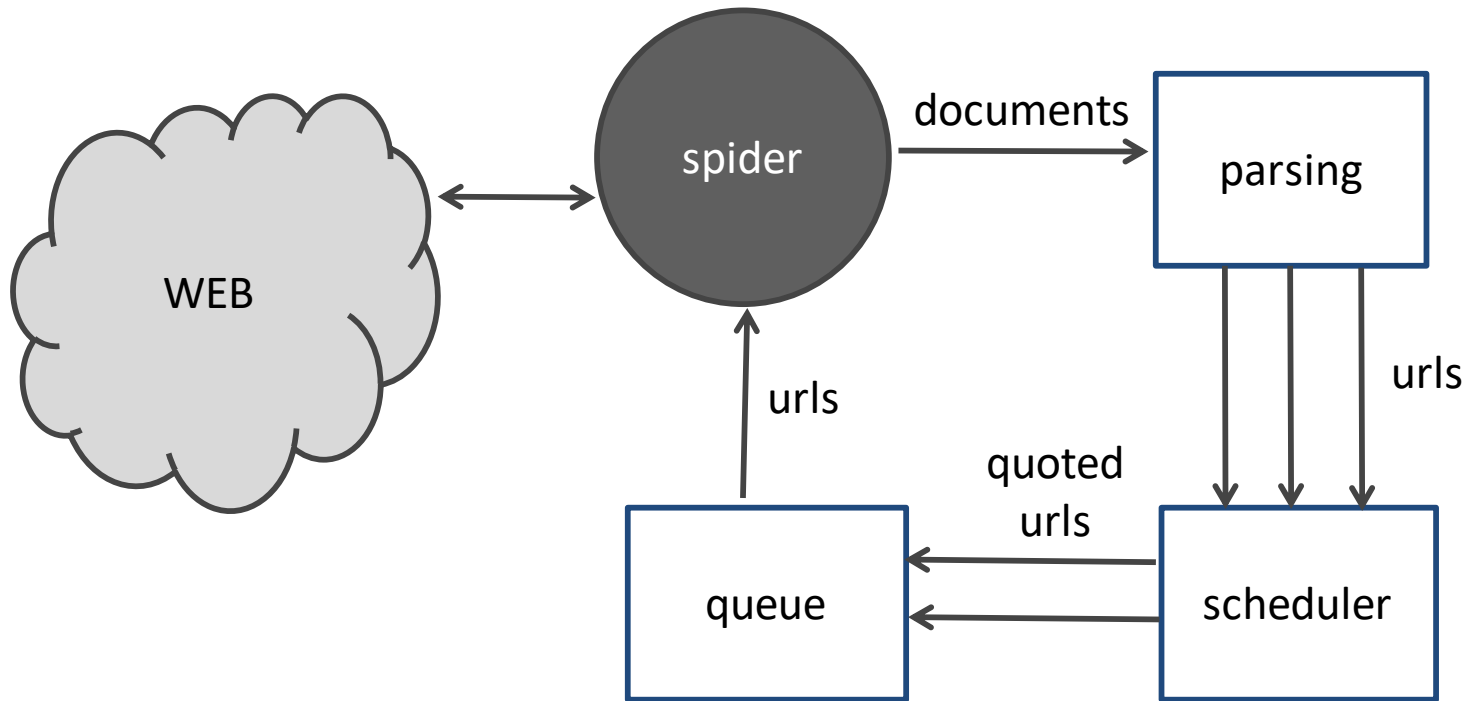
Учитывает:

1. Количество уже скачанных документов с сайта (успешно и нет)
2. Свойства скачанных документов (тип / язык)
3. Свойства самого сайта (посещаемость, CTR и т.д.)

Формируется квота.

Лекции про ранжирование

Spider & utils



Спайдер

1. Постановка задачи
2. Выкачка
3. Обновление
4. Хранение

Зачем перекачивать страницы?

1. Обновилось содержимое
2. Появились ссылки на новые страницы

Пример: главная страница сайта

Как часто перекачивать?

Простой подход:

если страница изменилась - $T = T/2$

если страница не изменилась - $T = T*2$

Усложнение:

- История выкачки
- Ранк сайта

Что важнее?

Выкачка новых страниц или перекачка старых?



Как понять, что страница изменилась?

<http://lenta.ru/>

<http://wellclix.net/>

<https://www.adme.ru/>

Как понять, что страница изменилась?

1. Брать только "чистый" контент
2. Удаление обвязки

Как понять, что страница изменилась?

Вэбмастера в одной лодке с нами

Http-response:

eTag

Last-Modified

В основном - для статического контента

Как понять, что страница изменилась?

```
$ HEAD http://s.imgur.com/images/loaders/ddddd1_181817/24.gif  
200 OK
```

```
ETag: "57a25124-14f9"
```

```
Last-Modified: Wed, 03 Aug 2016 20:16:36 GMT
```

```
$ HEAD -H 'If-None-Match: "57a25124-14f9"'  
http://s.imgur.com/images/loaders/ddddd1_181817/24.gif  
304 Not Modified
```

```
$ HEAD -H 'If-None-Match: "57a25124-14f8"'  
http://s.imgur.com/images/loaders/ddddd1_181817/24.gif  
200 OK
```

```
$ HEAD -H 'If-Modified-Since: Wed, 03 Aug 2016 20:16:36 GMT'  
http://s.imgur.com/images/loaders/ddddd1_181817/24.gif  
304 Not Modified
```

Дополнительные источники информации

<http://simonscat.tumblr.com/rss>

```
<?xml version="1.0" encoding="UTF-8"?>
<rss xmlns:dc="http://purl.org/dc/elements/1.1/" version="2.0">
<channel>
  <description>Channel description</description>
  <title>Simon's Cat</title>
  <item>
    <title>Simon's Cat refusing to face Monday! </title>
    <description>post description</description>
    <link>http://simonscat.tumblr.com/post/150306700829</link>
    <pubDate>Mon, 12 Sep 2016 12:33:35 +0100</pubDate>
  </item>
  ...
</channel>
```

Дополнительные источники информации

<http://all-t-shirts.ru/sitemap.xml?start=0>

```
<urlset>
```

```
  <url>
```

```
    <loc>http://all-t-shirts.ru/</loc>
```

```
    <lastmod>2016-03-28T00:03:15+03:00</lastmod>
```

```
    <changefreq>daily</changefreq>
```

```
  </url>
```

```
  ...
```

```
</urlset>
```

Спайдер

1. Постановка задачи
2. Выкачка
3. Обновление
4. Хранение

Хранение скачанных документов

Документ <--> урл

Ключ - f(url)

Практика. Есть разные способы записать один URL

<https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BD%D0%B8>

<https://ru.wikipedia.org/wiki/Пони>

<https://ru.wikipedia.org/wiki/%CF%EE%ED%E8>

http://kikolani.com/blog-post-promotion-ultimate-guide?utm_source=kikolani&utm_medium=320banner&utm_campaign=bpp

<http://kikolani.com/blog-post-promotion-ultimate-guide>

<http://scifi.stackexchange.com/questions?page=4&sort=newest>

<http://scifi.stackexchange.com/questions?sort=newest&page=4>

<https://music.yandex.ru/album/3575649/track/29692077>

<http://music.yandex.ru/album/3575649/track/29692077/>

<https://www.music.yandex.ru/album/3575649/track/29692077>

http://opennet.ru/docs/RUS/inet_book/4/45/retr4514.html

http://www.opennet.ru/docs/RUS/inet_book/4/45/retr4514.html

<http://домены.рф/>

<http://xn--d1acufc5f.xn--p1ai/>

<http://domeny.rf/>

Хранение документов

Нормализация урла

RFC: <https://www.ietf.org/rfc/rfc1738.txt>

Хранение документов

И проверка на валидность

<http://domeny.rf/> - .rf не существует

Хранение документов

Нормализованный URL - всегда в ASCII

Percent-encoding для query и пути

```
$ python -c "import urllib, sys; print urllib.quote(sys.argv[1])" Пони  
%D0%9F%D0%BE%D0%BD%D0%B8
```

Punycode для имени домена:

```
$ python -c "import urllib, sys; print sys.argv[1].decode('utf-8').encode('idna')"  
домены.рф  
xn--d1acufc5f.xn--p1ai  
$ python -c "import urllib, sys; print sys.argv[1].decode('idna')" xn--d1acufc5f.xn--  
p1ai  
домены.рф
```

Хранение документов

Нормализованный URL - всегда в ASCII

<https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BD%D0%B8>

<https://ru.wikipedia.org/wiki/Пони>

<https://ru.wikipedia.org/wiki/%CF%EE%ED%E8>

<http://домены.рф/>

<http://xn--d1acufc5f.xn--p1ai/>

Хранение документов

utm-метки для маркировки траффика

Параметры, которые игнорируются сервером, но учитываются в статистике

Позволяют оценить успешность рекламных кампаний (источники переходов)

Хранение документов

utm-метки для маркировки траффика

http://kikolani.com/blog-post-promotion-ultimate-guide?utm_source=kikolani&utm_medium=320banner&utm_campaign=bpp

<http://kikolani.com/blog-post-promotion-ultimate-guide>

Хранение документов

www. - наследие старого мира

Большинство - редиректят на нужную версию

Есть исключения:

www.music.yandex.ru - редиректит на корневик

[http://www.opennet.ru/](http://www.opennet.ru) и [http://opennet.ru/](http://opennet.ru) - обе отдают контент (одинаковый)

Хранение документов

Зеркало - сайт (до 80%) дублирующий контент оригинала

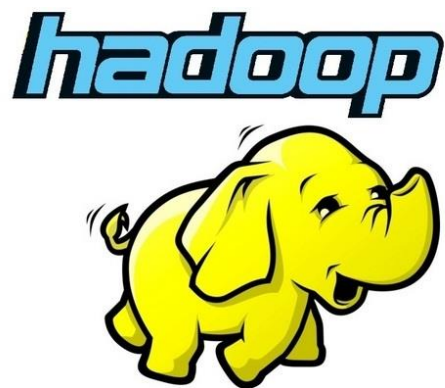
1. Защита от падения
2. ... и от блокировок (lurkmore.to, lurklurk.com, lurkmirror.ml)
3. Дорогой внешний трафик - локальное зеркало

Как бороться? Искать дубликаты

Хранение документов

> 40 Pb

> 100 млрд. документов

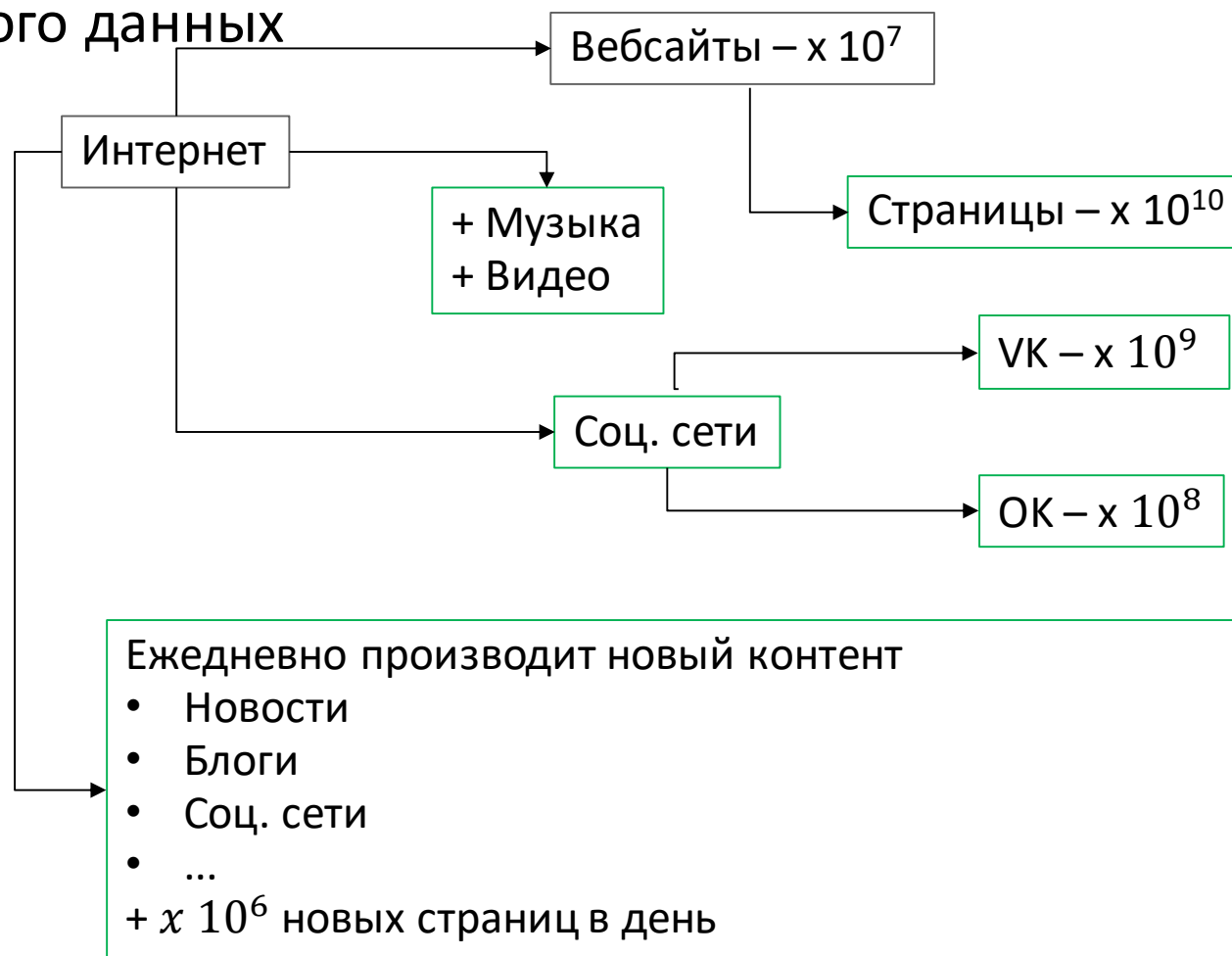


Хранение документов



Почему мы не можем скачать всё?

Слишком много данных



Почему мы не можем скачать всё?

Слишком много данных

При этом:

Качество поиска напрямую зависит от количества или качества страниц в индексе

Дадим спайдеру мозги

Бездушный wget качает всё.

Наш идеальный спайдер будет качать только те страницы, которые нужны пользователям.

Задача

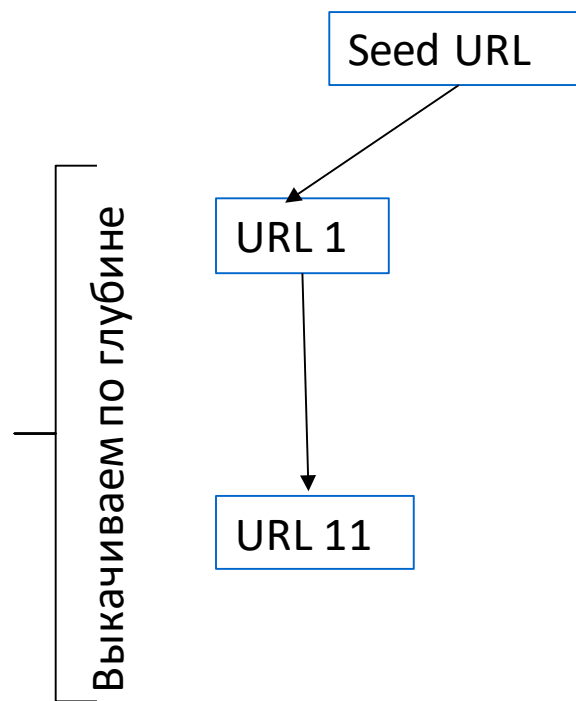
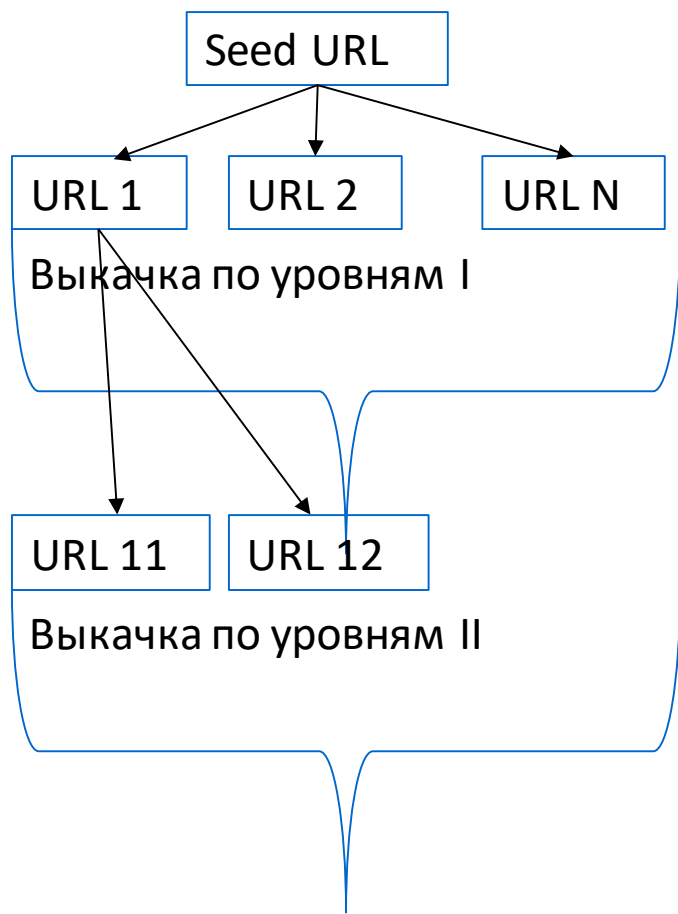
«Running a web crawler is a challenging task»

Sergey Brin and Lawrence Page, 1998

Алгоритмы обкачки

Обход сайта

Алгоритмы обкачки. Поиск в ширину/ глубину



Без приоритета!!!

Алгоритмы обкачки.

- Использует предположение, о взаимной релевантности страниц.
 - $A \rightarrow B$ – значит темы A и B одинаковые
- Вес B оценивается через вес A по отношению к теме T.
- Вес оценивается через схему TF-IDF, через близость к некоторой теме.

Алгоритмы обкачки. Fish Search

- «игра в жизнь» с «рыбками» в главной роли
- Есть точка входа. d – минимальная глубина обкачки
- Скачиваем страницу. Запускаем агентов по каждой исходящей ссылке и пересчитываем d для каждой страницы:
 - Родительский документ и текущая страница релевантны:
 $d(\text{child}) = d$
 - Нерелевантны: $d(\text{child}) = d - 1$
- В итоге «рыбки» умирают

Алгоритмы обкачки. PageRank/HITS.

Авторитетность источников.

- PageRank – индекс цитируемости страницы. Чем больше цитируют, тем больше вес ссылки
- HITS – индекс цитируемости с учетом авторитетности источника.

Фокусировка поискового робота

Сводится к построению очереди на обкачку и ее регулярной балансировке

Фокусировка поискового робота

Фокусируем робота через:

- Приоритет (priority-based)
- Структура (structure-based)
- Контекст (context-based)
- Поведение пользователей (behavioral-based)
- Обучение (learning-based)

Фокусировка: priority-based

Дано:

- Скачанная страница
- Метрика релевантности текста (например, тематические словари)

Фокусировка: priority-based

Алгоритм:

1. Скачали страницу
2. Оценили ее релевантность – число
3. Все ссылки с нее – с этим приоритетом
4. Очередь ссылок упорядочивается по их приоритетам

Фокусировка: structure-based

Учитываем структуру страницы:

- Заголовок важнее обычного текста
- Приоритет: не просто вхождения тематических слов, но и куда именно они входят

Фокусировка: structure-based

Division Score:

- У нас есть тематические словари. Тематика – Division. У каждой тематики – свой коэффициент.
- Каждая ссылка принадлежит определенной тематике (например, по родительской странице).
- Division score – отношение пересечения слов ссылки и всей тематики к полному словарю темы

Фокусировка: context-based

У ссылок тоже есть текст.

Текст ссылки определяет ее релевантность.

+ PageRank/HITS

Фокусировка: behavioral-based

Предыдущие подходы – релевантность каким-то тематикам.

Поиск – для пользователей!

Давайте учтем их интересы.

Фокусировка: behavioral-based

Давайте учтем интересы пользователей:

- Какие запросы задают
- Какие сайты посещают

Фокусировка: learning-based

До сих пор работали с уже обкачанной страницей.

Можно научиться предсказывать приоритет страницы.

Фокусировка: learning-based

Например, ML

Features:

- Релевантность родительской страницы
- Количество входящих ссылок (и их тексты)
- Форма урла (?)

Очередь на обкачку

- У каждого урла есть приоритет
- Приоритеты сравнимы между собой
- Очередь упорядочена

Считаем, что так мы будем в первую очередь скачивать страницы, которые **интересны** пользователю.

Какие страницы интересны?

Что мы можем ранжировать

- С заранее определёнными темами
 - Знаем запросы пользователей, и качаем страницы релевантные этим запросам, взвешивание по релевантности
- С большим индексом цитируемости
 - Знаем, на что ссылаются и взвешиваем их по количеству входящих ссылок
- С хорошей пользовательской посещаемостью
 - Раз ходят пользователи, значит это нужные страницы. Можно взвешивать на количество посещений

Какие страницы интересны?

С заранее определёнными темами:

- Хорошо для популярных запросов
- Плохо для низкочастотных запросов
- Много ресурсов: скачать, оценить, принять решение

Какие страницы интересны?

С большим индексом цитируемости:

- Хорошо для популярных ресурсов
- Плохо для новых сайтов (с нерелевантными входящими ссылками)
- Линкофермы(!)

Какие страницы интересны?

С большим индексом цитируемости:

Как ИЦ коррелирует с популярностью страниц у реальных пользователей?

Какие страницы интересны?

С хорошей пользовательской посещаемостью:

- Страницы точно нравятся пользователям
- Мало данных – проще работать

Какие страницы интересны?

С хорошей пользовательской посещаемостью:

- Страницы точно нравятся пользователям
- Мало данных – проще работать

QLink – query-link

Ранк – количество переходов. Нормируем по всем переходам на сайт.

Полнота покрытия

У Qlink малая полнота покрытия (~5-10%)

Полнота покрытия

У Qlink малая полнота покрытия (~5-10%)

Экстраполируем эти данные:

1. Для всего сайта
2. По ссылкам ($A \rightarrow B$: $ql(B) = coef * ql(A)$)
3. Для сегмента сайта

Что такое сегмент

Например:

Host: aldebaran.ru

Path: /kid/krapiv/krapiv[0-9]*\$

Query: *

Что такое сегмент

Предполагаем, что урлы, похожие по форме, имеют схожее содержимое.

В контексте QLink:

чем больше QLink в сегменте, тем выше вероятность, что и неизвестные еще урлы из сегмента понравятся пользователям

URL

RFC 1738, RFC 3986

Нас интересует схема `http – address`:

- `http://<host>:<port>/<path>?<query>#<fragment>`
- `<host>:<port>` - одинаковы для всего сайта
- `fragment` выбрасываем (анкеры в ajax)
- Нас интересует `<path>` и `<query>`
 - `path = segment *["/" segment]`
 - `segment = *[uchar | ";" | ":" | "@" | "&" | "="]`
- `<query>` состоит из пар `name=value` разделенных `&`
- Порядок следования пар в `<query>` не важен (на нормальных сайтах)

Кластеризация урлов

Сколько урлов надо взять, чтобы разбить на сегменты с выраженными особенностями?

Слишком мало: неточные и крупные сегменты

Слишком много (и слишком строгие условия): много мелких сегментов, распыление QLink

Сколько урлов брать?

Сегмент == тематика

α – вероятность встретить урл из тематики

N – размер сэмпла

Какова вероятность найти менее k урлов из тематики?

$$p_{N,k}(\alpha) = \sum_{i=1}^k \binom{i}{N} \alpha^i (1 - \alpha)^{(N-i)}$$

$$P_{1000,10}(0.01) \approx 0.58$$

$$P_{1000,10}(0.02) \approx 0.01$$

$$P_{1000,10}(0.03) \approx 2 \times 10^{-5}$$

石庭(sekitei)



石庭(sekitei)

1. Отбираем N случайных урлов с сайта

石庭(sekitei)

1. Отбираем N случайных урлов с сайта
2. Создаем признаки для урлов
 - Длина урла
 - Количество query-параметров
 - Сегменты пути
 - Query-параметры
 - Регулярки O_O

石庭(sekitei)

1. Отбираем N случайных урлов с сайта
2. Создаем признаки для урлов
 - Длина урла
 - Количество query-параметров
 - Сегменты пути
 - Query-параметры
 - Регулярки O_O
3. Отбираем признаки по частотности: αN
4. Кластеризуем:
 - Jaccard distance measure
 - Clustering

$$K(a, b) = \frac{|A \cap B|}{|A \cup B|}$$

Отбираем N случайных урлов

Насколько случайных?

Отбираем N случайных урлов

1. Сколько урлов? Примерно 1к

$$p_{N,k}(\alpha) = \sum_{i=1}^k \binom{i}{N} \alpha^i (1 - \alpha)^{(N-i)}$$

$$P_{1000,10}(0.01) \approx 0.58$$

$$P_{1000,10}(0.02) \approx 0.01$$

$$P_{1000,10}(0.03) \approx 2 \times 10^{-5}$$

2. Насколько случайные? Известные:неизвестные – 1:1

Признаки урлов

`somesite.com/path/to/url.html?a=1&b=2`

1. Количество сегментов:
 - `/path/to/url.html` – 3
2. Query:
 1. Количество параметров: 2
 2. Список параметров: `a+b`
 3. Наличие пары параметр-значение: `a=1`
3. Конкретные сегменты в пути:
 1. На 1-ой позиции `path`
 2. На 2-ой позиции `to`
 3. ...

Конкретные сегменты в пути

Давайте использовать регулярки

/path/12345/day_576/image.jpg

path -> "path", "[^/]+"

12345 -> "12345", "[0-9]+"

day_576 -> "day_576", "[^/]_576", "day_[0-9]+", "[^/]+_[0-9]+"

Image.jpg -> "image.jpg", "[^/]+.jpg", "[^/]+"

Пример

Создаем признаки для каждого адреса (пример):

http://www.sports.ru/tags/1365242.html?p=57&type=photo

↑ ↑ ↗
сегмент путь запрос

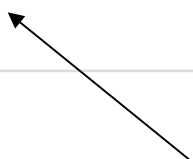
№	Название признак
1	2 Сегмента
2	Запрос состоит из двух параметров
3	0-й сегмент пути: tags
4	1-й сегмент пути: 1365242\.html
5	1-й сегмент пути; [0-9]+\.
6	1-й сегмент пути: [^/]+\.
7	В запросе есть параметр p=57
8	В запросе есть параметр type=photo

Пример

Отсекаем признаки по частотности αN

- Отбираем признаки для sport.ru
- $\alpha = 0.1; N = 1000$

	Н (частота)	Признак
1	759	Пустой запрос
2	379	В пути ровно два сегмента
3	328	0-й сегмент: fantasy
4	321	1-й сегмент пути: [^/]+\..html
5	315	1-й сегмент пути: [0-9]+\..html
6	266	1-й сегмент пути: football
7	249	В пути ровно 4 сегмента



Не берем признаки с частотой
меньше: $0.1 * 1000 = 100$

Кластеризация

- Используем любой алгоритм, который позволяет нам найти кластера по выделенным признакам
 - Принадлежность сегменту определяем через пространство признаков
- Формируем регулярные выражения в формате PCRE для найденных кластеров.
 - Принадлежность сегменту определяем по регулярным выражениям описывающим кластер.

Что делать с остатком?

Кластеризация

- Используем любой алгоритм, который позволяет нам найти кластера по выделенным признакам
 - Принадлежность сегменту определяем через пространство признаков
- Формируем регулярные выражения в формате PCRE для найденных кластеров.
 - Принадлежность сегменту определяем по регулярным выражениям описывающим кластер.
- Урлы вне сегментов – тоже сегмент.

Пример. Случай регулярных выражений

1. **`^/wiki/File:[^/]+\.`**`jpg$`

Регулярное выражение,
описывающее кластер

`/wiki/File:Spongilla_lacustris.jpg`

2. **`^/wiki/[^\.]+\.`**`jpg$`

`/wiki/Image:Deve.jpg`

3. **`^/wiki/Category:[^/]+$`**

`/wiki/Category:Roman-era_historians`

4. **`^/wiki/Talk:[^/]+$`**

`/wiki/Talk:North_Light`

...

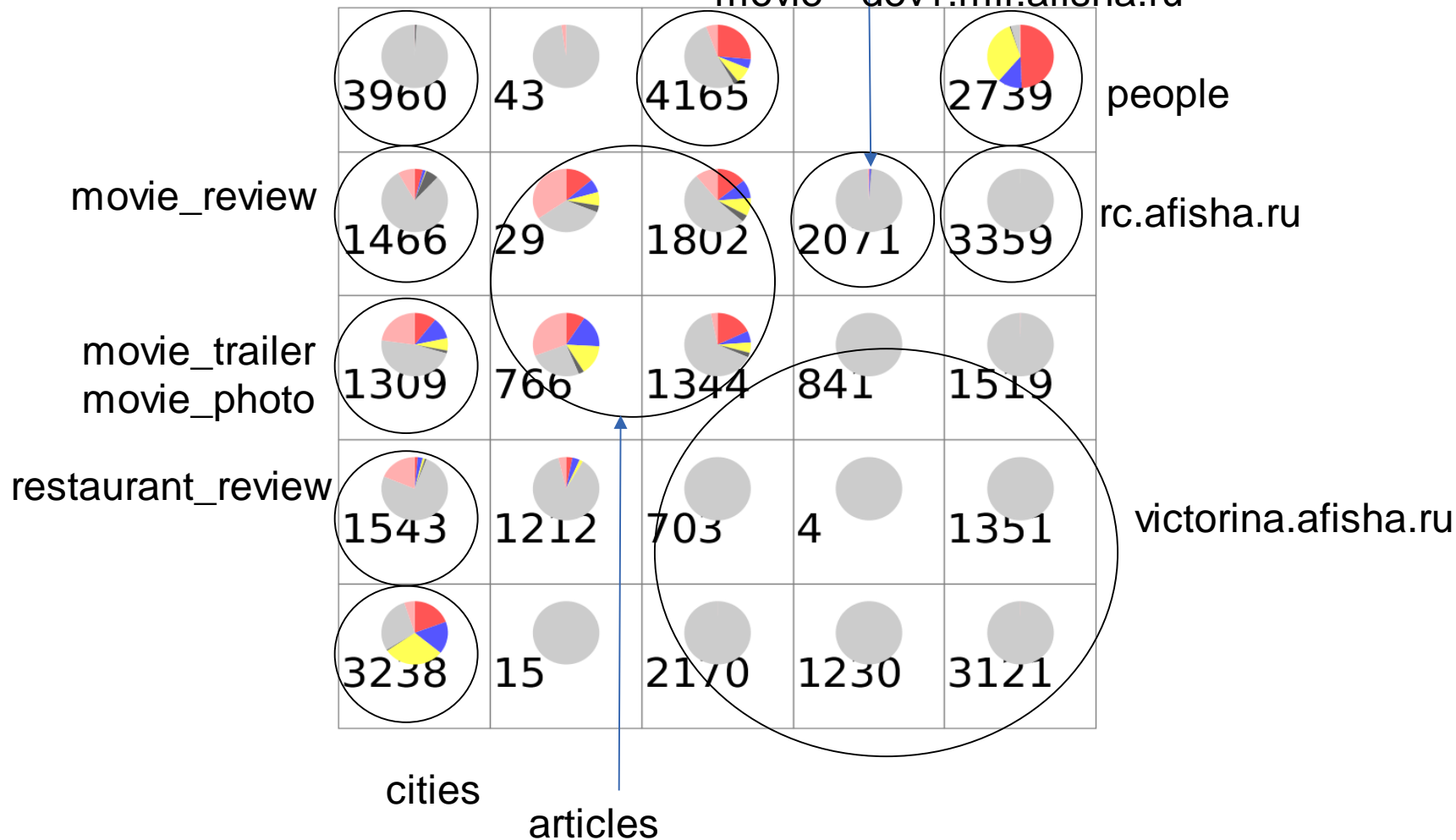


Кластеризация

afisha.ru

personalpage/review

movie dev1.mir.afisha.ru



Что делать с сегментами?

- QLink'и позволяют оценить сегмент
- Хороший сегмент качаем активнее
- Используем эту информацию в построении индекса

Что делать с сегментами?

Давайте качать и показывать людям только сегменты с QLink'ами!

Нельзя – вырожденные случаи + есть то, что мы не знаем

Статический ранк (Static Rank)

Ранк – число.

Получаем из:

1. Sekitei
2. Антиспам
3. Ссылочные (Indegree, PR и т.д.)

Строим модель: gradient boosting decision trees

Предугадываем, сколько QLink получим.

Модели:

- Индивидуальные для больших сайтов
- Общая для всех остальных

Что дальше?

Мы научились оценивать урлы и упорядочивать их по значимости.

В очереди на выкачку лежат урлы с разных сайтов.

Как соотносятся оценки между сайтами?

Надо ли качать весь сайт?

Квотирование

Мы ограничены в возможностях хранения и индексации

Размер квоты:

- Всем поровну
- По посещаемости
- По сегментам

Как оценить качество?

Как оценить качество?

Мы утверждаем, что умеем предсказывать появление QLink в результатах выкачки (не ниже определенного уровня)

Оцениваем, сколько в итоге оказалось

Как оценить качество?

Цель: собрать индекс фиксированного размера

Сайты:

- «хостинги» – домены 2 уровня, чьи поддомены – посещаемые и крупные, часто независимые друг от друга, сайты. Пример: livejournal.com
- «большие сайты» – поддомены не-хостингов, которые по характеру могут быть выделены в отдельный сайт. Пример: mail.ru – не «хостинг», но можно выделить tu.mail.ru
- все остальные

Алгоритм Секитей	Жадный алгоритм - посещаемость
MIN_QUOTA ~ 100	MIN_QUOTA ~ 100
QUOTA = #PagesWithQlinks * MIN_QUOTA	QUOTA = F(#Visits) * MIN_QUOTA
Квота по камням	Квота для сайта

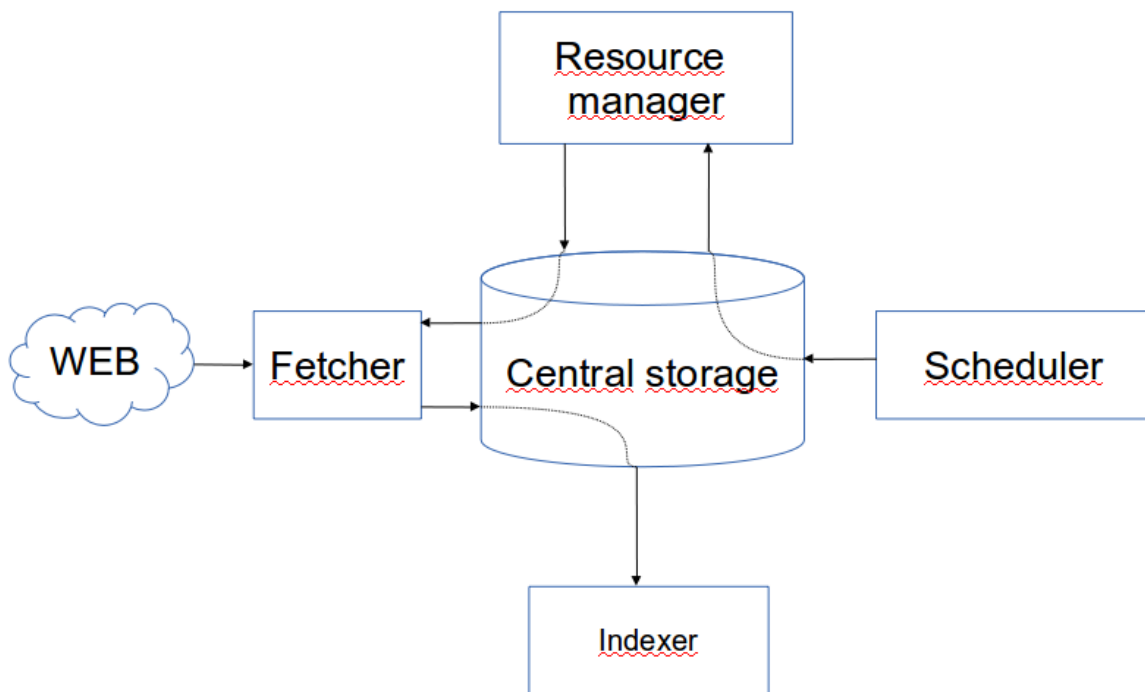
Как всё работает

Fetcher – просто качает (чуть сложнее wget'а)

Resource manager – доп.данные

Scheduler – квотирование, очереди на обкачку – батчи

Indexer – индексация урлов, построение поискового индекса



Вопросы?