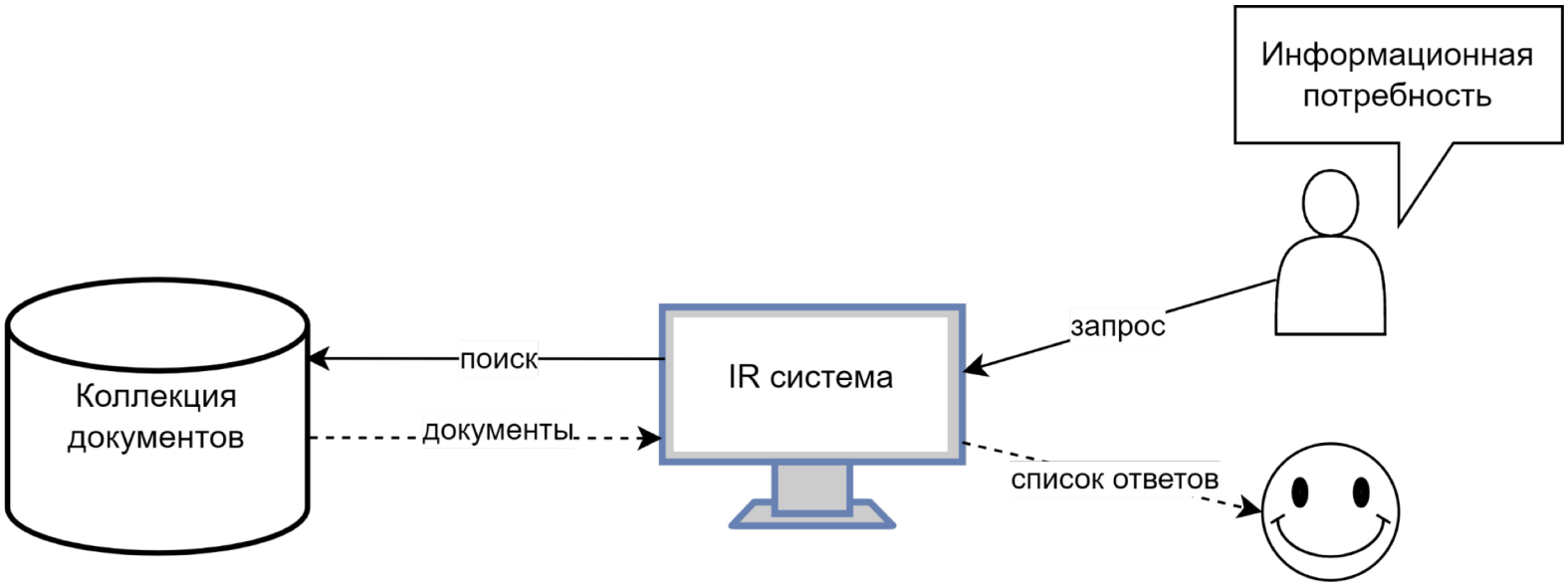
Информационный поиск

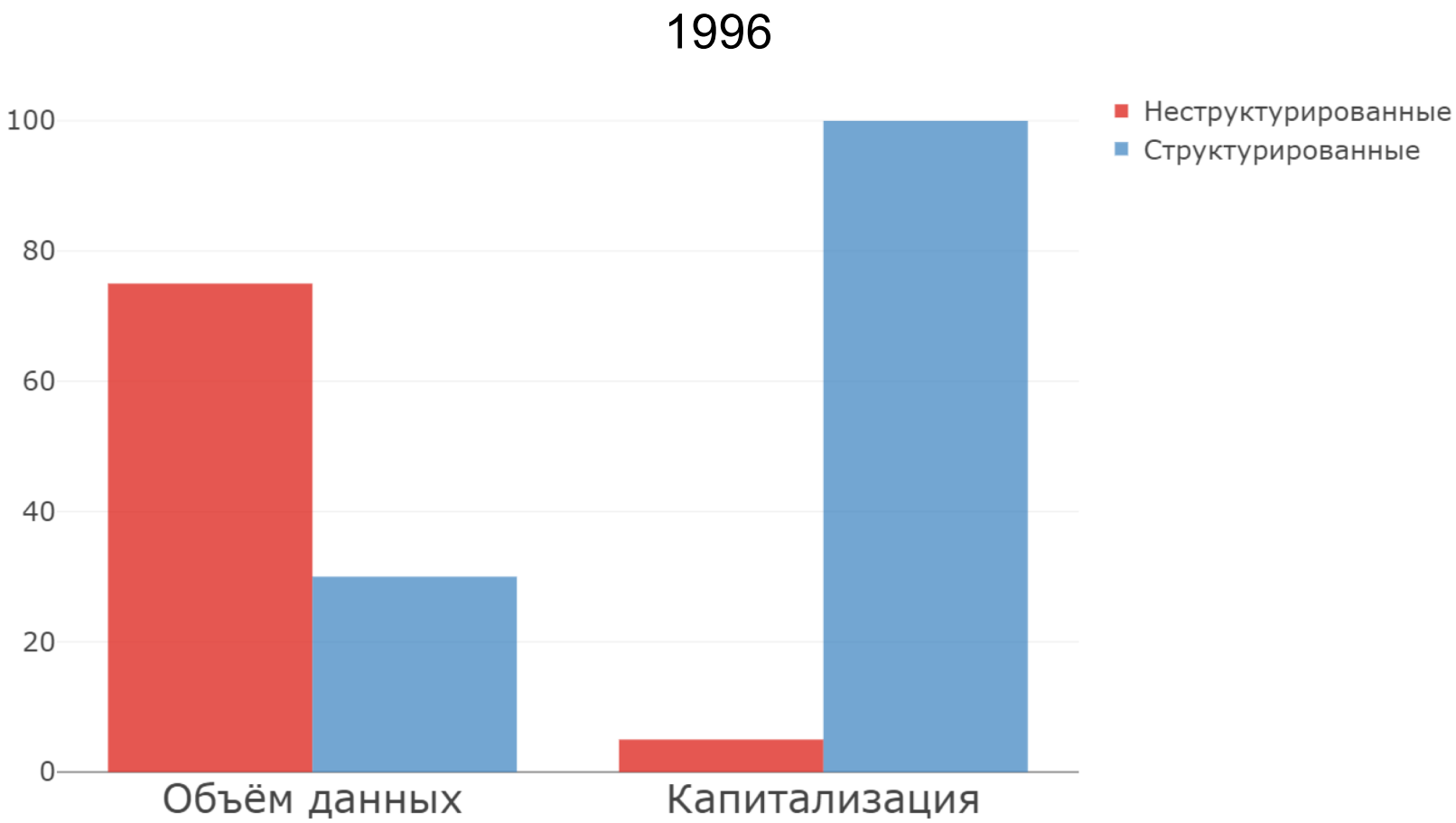
Лекция 11

Определение

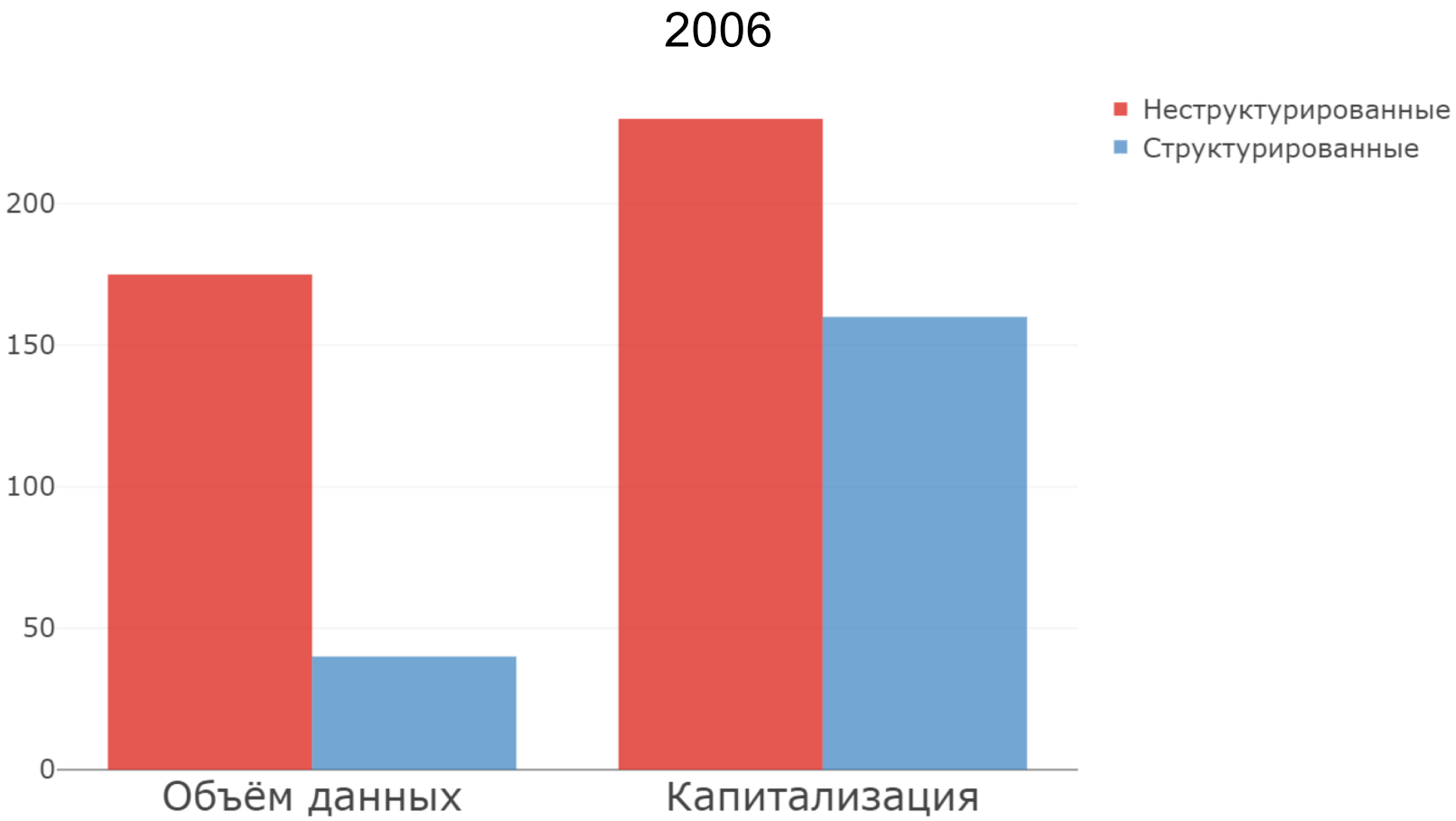
Информационный поиск (Information retrieval – IR) — это процесс поиска в большой коллекции (хранящейся, как правило, в памяти компьютеров) некоего неструктурированного материала, удовлетворяющего информационные потребности.



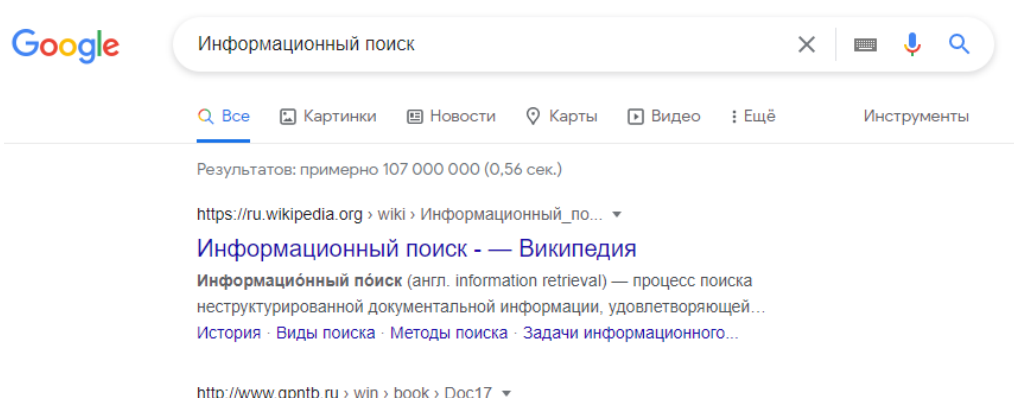
Распространение задачи

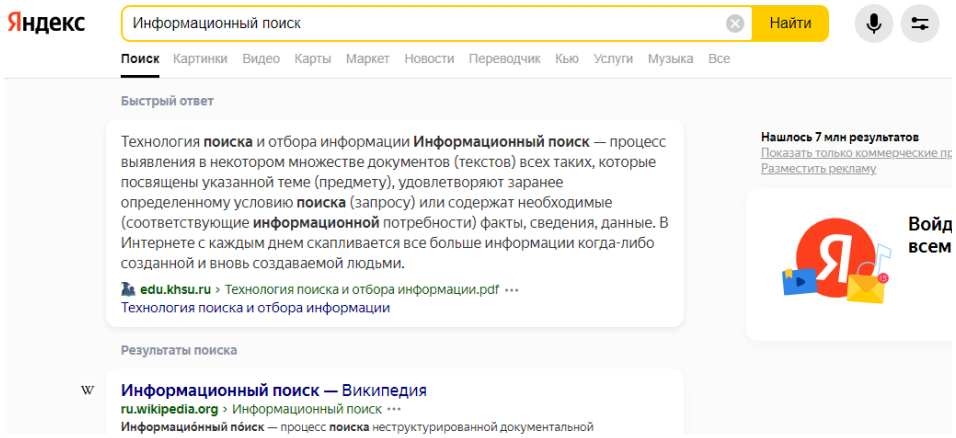


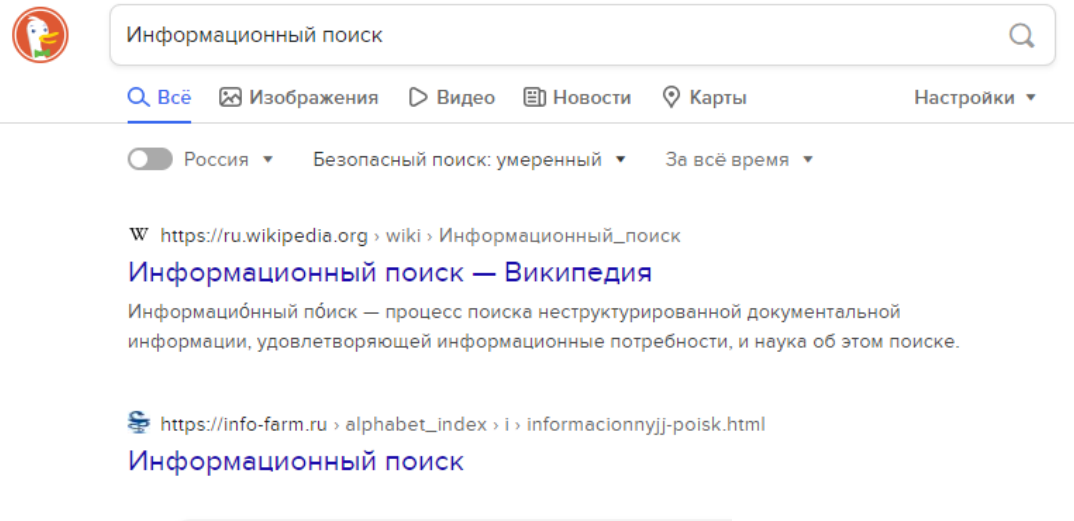
Распространение задачи

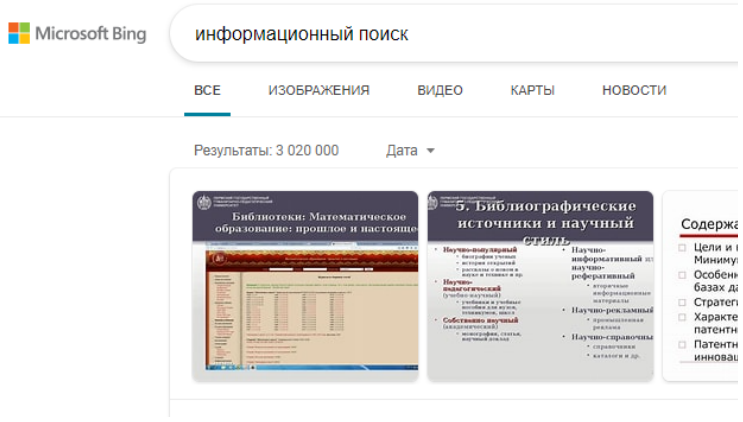


Поисковые системы

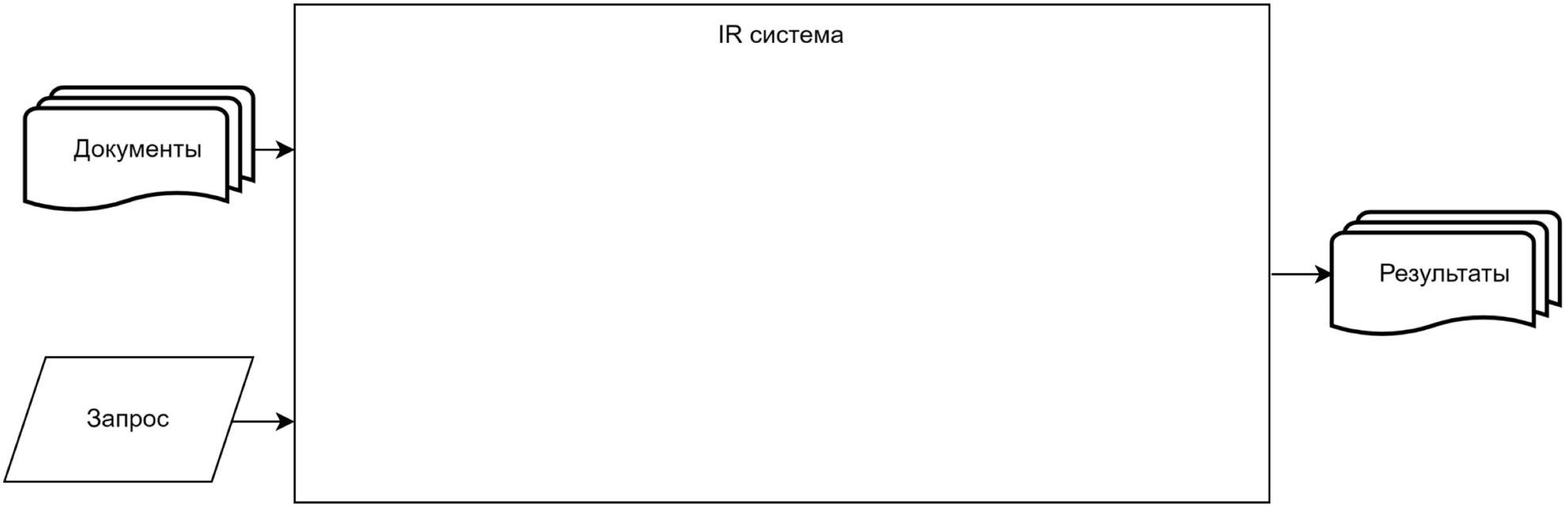






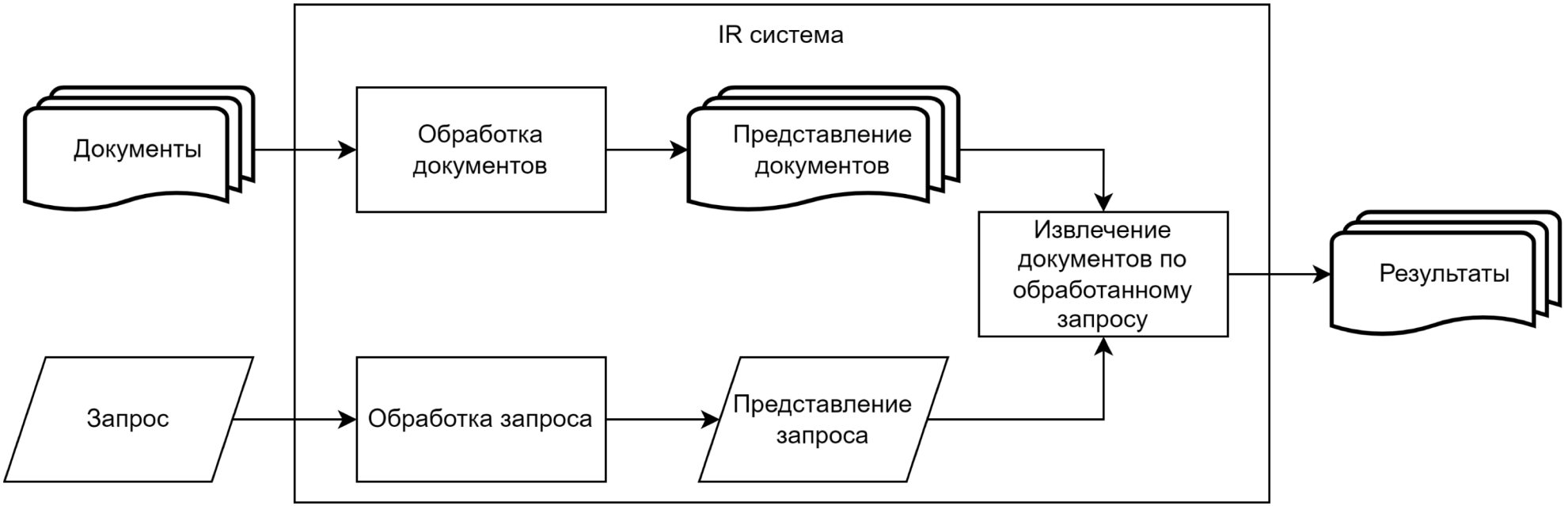


Архитектура IR систем



Что входит в систему?

Архитектура IR систем



Задачи информационного поиска

- Обработка документа
  - В каком виде оптимально хранить информацию о документе. В каком виде оптимально хранить документ в памяти
- Обработка запроса
  - Как представить запрос
- Извлечение документов
  - Какой документ удовлетворяет запросу
- Оценка системы
  - Как узнать, что система работает хорошо

Что такое документ?

Документ 1

This is first document with one sentence.

Документ 2

This is another document

Документ 3

Third document.

Матрица инцидентности

Документ 1

This is first document with one sentence.

Документ 2

This is another document

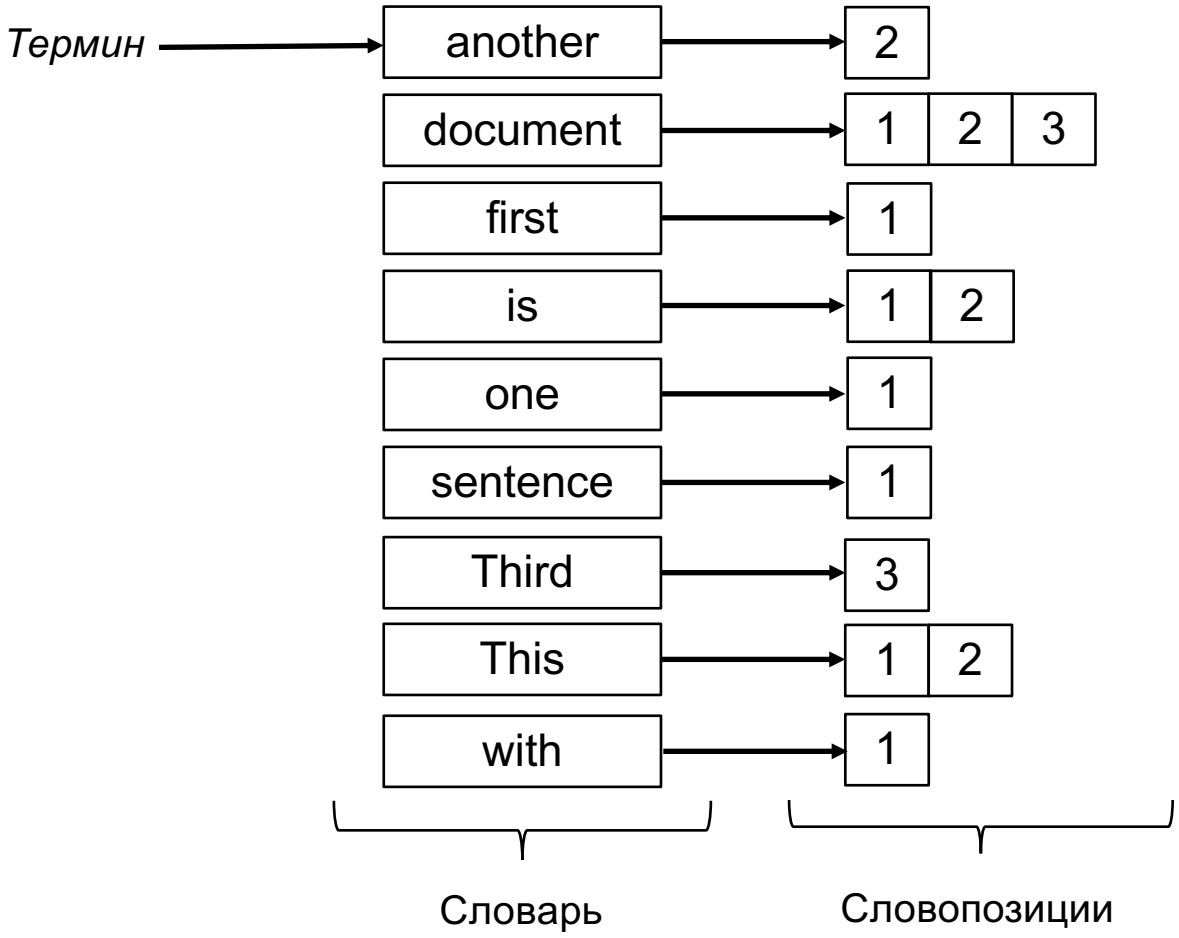
Документ 3

Third document.

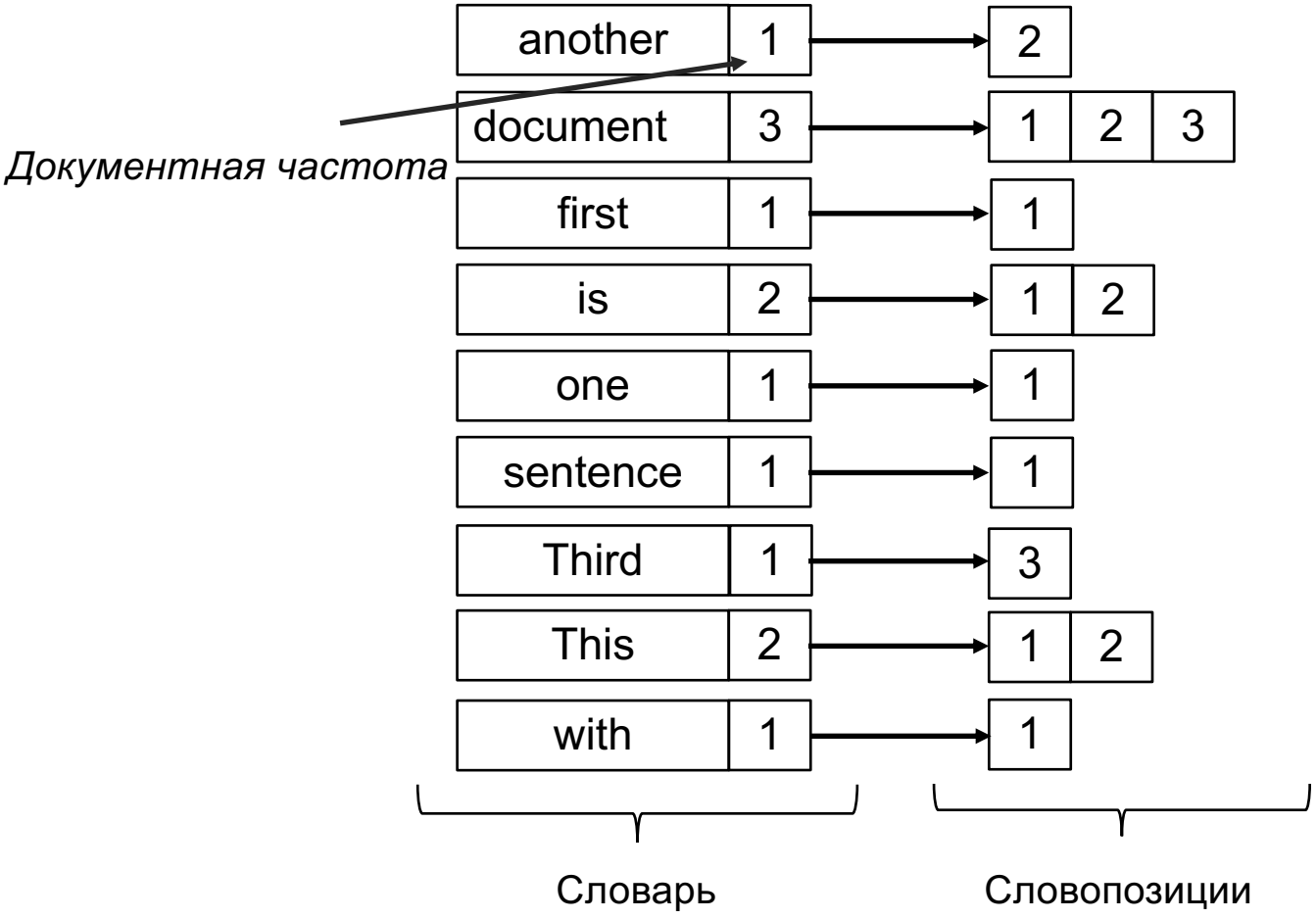
	1	2	3
This	1	1	0
is	1	1	0
first	1	0	0
document	1	1	1
with	1	0	0
one	1	0	0
sentence	1	0	0
another	0	1	0
Third	0	0	1

Чем плоха матрица инцидентности?

Инвертированный индекс



Инвертированный индекс



Предварительная обработка

- Игнорирование распространенных терминов: стоп-слова

- Нормализация
  Ударения и диакритические символы
  Обработка без учета регистра

- Стемминг и лемматизация

Модели информационного поиска

- Документ D = множество взвешенных ключевых слов
- Запрос Q = множество невзвешенных слов

$$R(D,Q) = \sum_i w(t_i, D)$$

t_i – слова запроса

Булева модель

- Документ – конъюнкция слов
- Запрос – булево выражение (AND, OR и NOT)

$$R(D, Q) = Q \rightarrow D$$
$$D = t_1 \wedge t_2 \wedge \ \ldots \wedge t_n$$
$$Q = (t_1 \wedge t_2) \vee (t_3 \wedge t_4)$$
$$Q \rightarrow D \ - \text{то есть } R(D, Q) = 1$$

Булева модель с инцидентной матрицей

$$R(D, Q) = (THIS\ OR\ THIRD)\ AND\ NOT\ WITH$$

	1	2	3
This	1	1	0
is	1	1	0
first	1	0	0
document	1	1	1
with	1	0	0
one	1	0	0
sentence	1	0	0
another	0	1	0
Third	0	0	1

Булева модель с инцидентной матрицей

$$R(D, Q) = (THIS\ OR\ THIRD)\ AND\ NOT\ WITH$$

$$(110\ OR\ 001)\ AND\ NOT\ 100 = 011$$

Результат: 2 и 3 документ

	1	2	3
This	1	1	0
is	1	1	0
first	1	0	0
document	1	1	1
with	1	0	0
one	1	0	0
sentence	1	0	0
another	0	1	0
Third	0	0	1

Найдите

1. $(NOT\ ANOTHER\ OR\ DOCUMENT)\ AND\ (IS\ OR\ THIS)$
2. $(NOT\ THIS\ AND\ WITH)\ OR\ (DOCUMENT\ AND\ THIRD)$
3. $(WITH\ AND\ DOCUMENT\ AND\ ANOTHER\ AND\ FIRST)\ OR\ SENTENCE$

	1	2	3	4	5	6
ANOTHER	0	1	0	0	0	0
DOCUMENT	1	1	1	1	0	1
FIRST	1	0	0	0	0	1
IS	1	1	0	0	0	0
ONE	1	0	0	0	0	0
SENTENCE	1	0	0	0	0	1
THIRD	0	0	1	1	1	0
THIS	1	1	0	1	0	0
WITH	1	0	0	1	0	1

Найдите

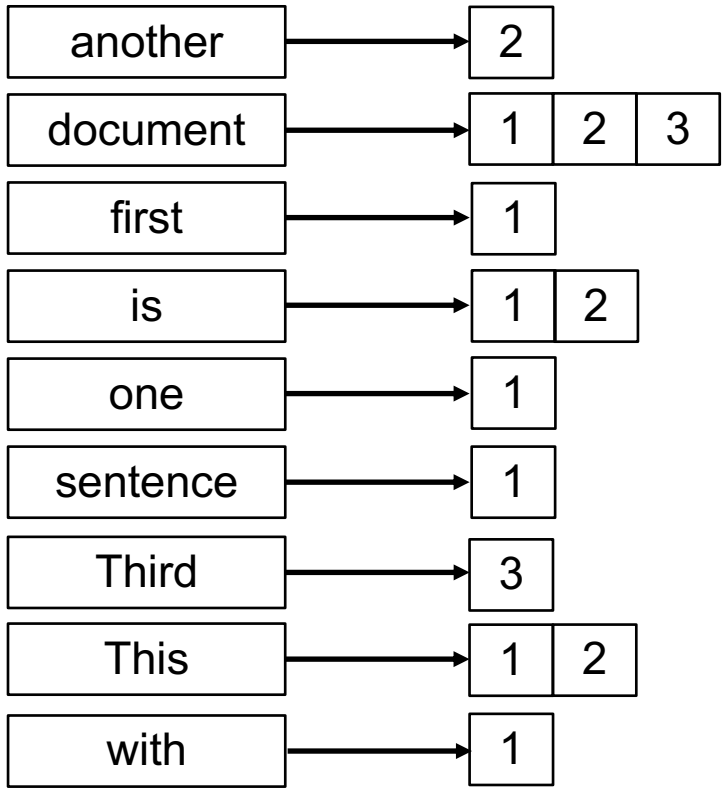
1. $(NOT\ ANOTHER\ OR\ DOCUMENT)\ AND\ (IS\ OR\ THIS)\ -1,2,4$
2. $(NOT\ THIS\ AND\ WITH)\ OR\ (DOCUMENT\ AND\ THIRD)\ -3,4,6$
3. $(WITH\ AND\ DOCUMENT\ AND\ ANOTHER\ AND\ FIRST)\ OR\ SENTENCE\ -1,6$

	1	2	3	4	5	6
ANOTHER	0	1	0	0	0	0
DOCUMENT	1	1	1	1	0	1
FIRST	1	0	0	0	0	1
IS	1	1	0	0	0	0
ONE	1	0	0	0	0	0
SENTENCE	1	0	0	0	0	1
THIRD	0	0	1	1	1	0
THIS	1	1	0	1	0	0
WITH	1	0	0	1	0	1

Булева модель с обратным индексом

$(THIS\ OR\ THIRD)\ AND\ NOT\ WITH$:

1. $THIS\ OR\ THIRD$
2. $NOT\ WITH$
3. $THIS\ OR\ THIRD\ AND\ NOT\ WITH$



Недостатки булева поиска

- $R \in [0,1]$ – неупорядоченные документы

- Обычно результатов на запрос либо слишком много, либо слишком мало

  Запрос 1: “скачать бесплатно” (4 млн. результатов)

  Запрос 2: “скачать бесплатно без регистрации без смс без кидалава без хурмы” (0 результатов)

- Писать булевы запросы сложно

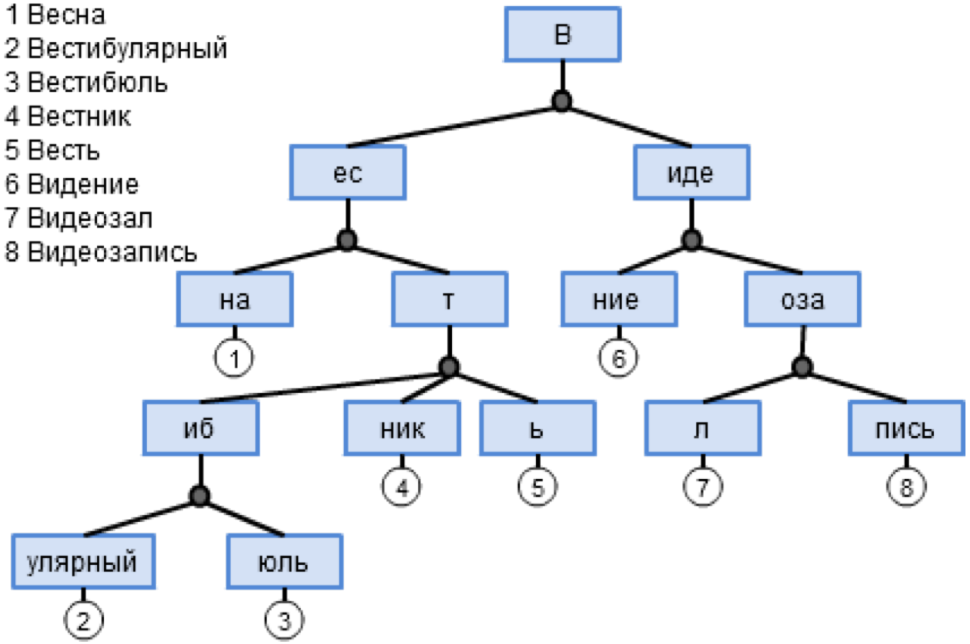
Нечёткий поиск. Запросы с джокерами

Часть запроса содержит джокер или wildcard, обозначенный символом " * "

Используются в следующих ситуациях:
1. пользователь не знает, как правильно пишется слово
2. пользователь знает о существовании нескольких вариантов написания
3. пользователь ищет документы, содержащие вариант термина, унифицированный в результате стемминга
4. пользователь не уверен в правильности воспроизведения иностранного слова или фразы

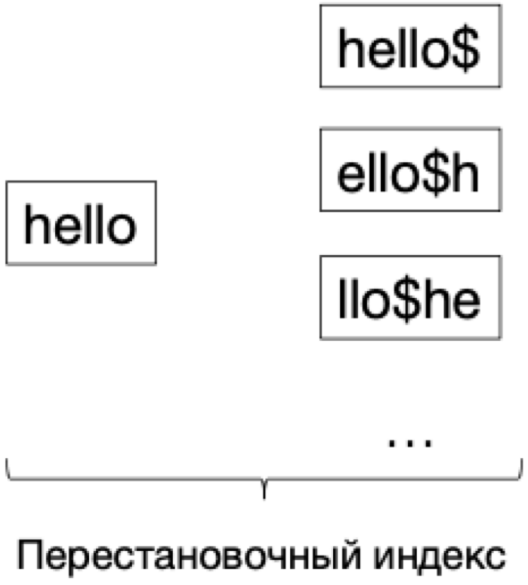
Сжатое префиксное дерево

1 Весна
2 Вестибулярный
3 Вестибюль
4 Вестник
5 Весть
6 Видение
7 Видеозал
8 Видеозапись



Перестановочные индексы

Введём специальный символ $, обозначающий конец термина. Перестановочный индекс терминов получается циклической перестановкой символов исходного термина



Для поиска может потребоваться некоторая модификация запроса:

m*n →n$m*

fi*mo*er → er$fi* → er$mo*

K-граммный индекс

При использовании k-граммного индекса словарь содержит все k-граммы, образованные из всех терминов лексикона. Каждый инвертированный список ставит в соответствие k-грамме все термины лексикона, содержащие данную k-грамму.

3-грамма: castle → $ca – cas – ast – stl – tie – le$

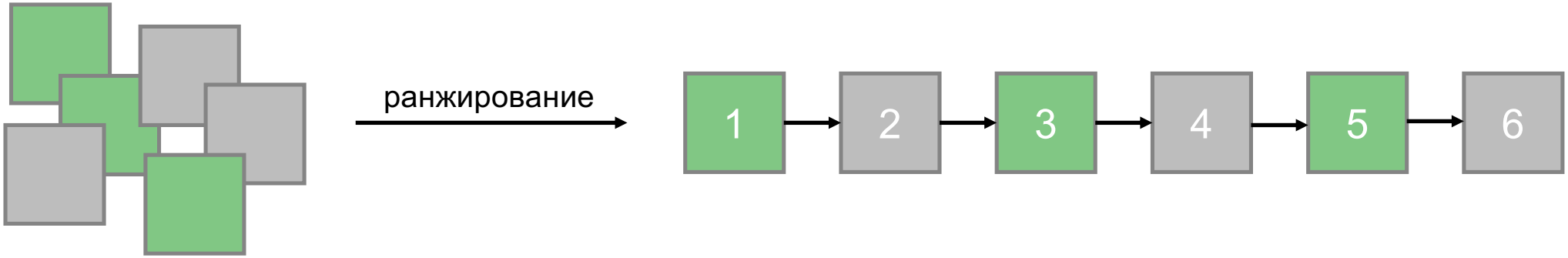
Модификация запроса (для 3-граммного индекса):
re*ve → $re AND ve$

Задача ранжирования

- Для некоторого запроса $q \in Q$ найдено множество документов → имеем выборку пар (d, q)
- Y — упорядоченное множество рейтингов, более высокое значение соответствует более высокой релевантности
- $y: X \to Y$ — оценки, проставленные асессорами
- Правильный порядок определён только на множестве документов, найденных по одному запросу

Ранжирование

- Взвешивание на основе комбинации частоты и обратной документной частоты термина
- Вероятностные методы
- Определение весов на основе машинного обучения



Релевантность документа

Взвешивание слов (терминов)

- tf – term frequency - частота встречаемости термина в документе
- df – document frequency - число документов, содержащих термин
- idf – inverse document frequency - специфичность термина
- $weight(t, D) = tf(t, D) \cdot idf(t)$

Варианты tf-idf

$$idf(t) = \log\left(\frac{N}{n}\right)$$

n – количество документов, содержащих t
N – количество документов в корпусе

- $tf(t, D) = freq(t, D)$
- $tf(t, D) = \log[freq(t, D)]$
- $tf(t, D) = \log[freq(t, D)] + 1$
- $tf(t, D) = \frac{freq(t,d)}{Max[f(t,d)]}$

Пример расчёта tf-idf

Документы:

- маленький котик ест еду
- большой щенок ест еду
- маленький котик большой котик и маленький щенок едят еду

еду	1
ест	1
щенок	0
маленький	1
большой	0
и	0
котик	1
едят	0

Прямой индекс 1

еду	1
ест	1
щенок	1
маленький	0
большой	1
и	0
котик	0
едят	0

Прямой индекс 2

еду	1
ест	0
щенок	1
маленький	2
большой	1
и	1
котик	2
едят	1

Прямой индекс 3

еду	3
ест	2
щенок	2
маленький	2
большой	2
и	1
котик	2
едят	1

Обратный индекс

Пример расчёта tf-idf

Документы:

- маленький котик ест еду
- большой щенок ест еду
- маленький котик большой котик и маленький щенок едят еду

еду	0
ест	0.176
щенок	0
маленький	0
большой	0
и	0
котик	0.176
едят	0

tf-idf 1

еду	0
ест	0.176
щенок	0.176
маленький	0
большой	0.176
и	0
котик	0
едят	0

tf-idf 2

еду	0
ест	0
щенок	0.176
маленький	0.352
большой	0.176
и	0.477
котик	0.352
едят	0.477

tf-idf 3

Векторная модель

- Векторное пространство всех слов

$$\langle t_1, t_2, t_3, \dots, t_n \rangle$$

- Документ

$$D = \langle a_1, a_2, a_3, \dots, a_n \rangle$$

$$a_i - \text{вес } t_i \text{ в документе } D$$

- Запрос

$$Q = \langle b_1, b_2, b_3, \dots, b_n \rangle$$

$$b_i - \text{вес } t_i \text{ в запросе } Q$$

Матричное представление

Пространство документов

Пространство терминов

	t_1	t_2	t_3	…	t_n
D_1	a_{11}	a_{12}	a_{13}	…	a_{1n}
D_2	a_{21}	a_{22}	a_{23}	…	a_{2n}
D_3	a_{31}	a_{32}	a_{33}	…	a_{3n}
…	…	…	…	…	…
D_m	a_{m1}	a_{m2}	a_{m3}	…	a_{mn}
Q	b_1	b_2	b_3	…	b_n

Разреженная матрица (!)

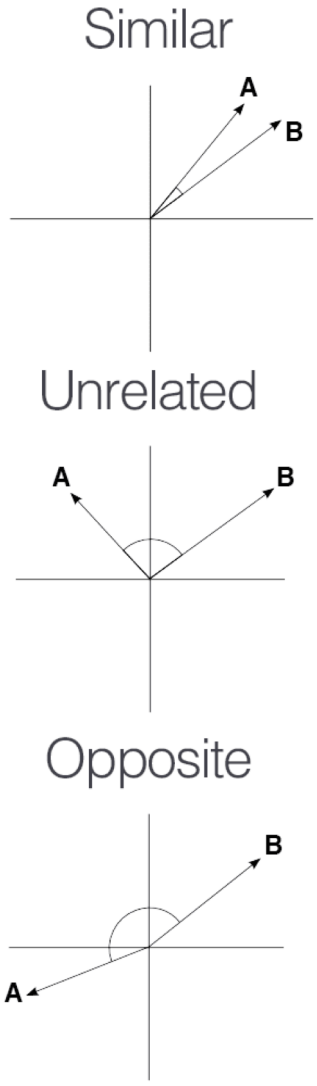
Подсчет близости

Скалярное произведение: $$Sim(D,Q) = \sum_{i=1}^{n} a_i \cdot b_i = \langle a, b \rangle$$

Косинусная мера: $$Sim(D,Q) = \frac{\langle a, b \rangle}{\sqrt{\sum_{i=1}^{n} a_i^2} \cdot \sqrt{\sum_{i=1}^{n} b_i^2}}$$

Мера Дайса: $$Sim(D,Q) = \frac{2\langle a, b \rangle}{\sum_{i=1}^{n} a_i^2 + \sum_{i=1}^{n} b_i^2}$$

Мера Жаккара: $$Sim(D,Q) = \frac{\langle a, b \rangle}{\sum_{i=1}^{n} a_i^2 + \sum_{i=1}^{n} b_i^2 - \langle a, b \rangle}$$



Вероятностная модель

Зная несколько релевантных и нерелевантных документов, можно непосредственно оценить вероятность того, что термин t появится в релевантном документе $P(t|R = 1)$.

Принцип бинарной независимости
Если поисковая система в ответ на запрос пользователя отдает документы в порядке уменьшения вероятности их релевантности запросу, то качество такой поисковой системы будет максимальным (Robertson, 1977)

Модель бинарной независимости

- Документ представляет собой бинарный вектор термов
- Запрос представляет собой бинарный вектор термов
- Предположение о независимости появления термов в документе

$$P(D|R) = \prod_{i=1}^{t} P(D_i|R)$$

Okapi BM25

Классический вероятностный алгоритм ранжирования,
Основанный на модели бинарной независимости

$$RSV_d = \sum_{t \in q} \log \left[\left[\frac{\left(|VR_t| + \frac{1}{2}\right) / \left(|VNR_t| + \frac{1}{2}\right)}{\left(df_t - |VR_t| + \frac{1}{2}\right) / \left(N - df_t - |VR| + |VR_t| + \frac{1}{2}\right)} \right] \right]$$

$$\times \frac{(k_1 + 1) tf_{rd}}{k_1 \left((1 - b) + b(L_d / L_{ave})\right) + tf_{td}} \times \frac{(k_3 + 1) tf_{tq}}{k_3 + tf_{tq}}$$

https://nlp.stanford.edu/IR-book/html/htmledition/okapi-bm25-a-non-binary-model-1.html

ML методы

- Point-wise — поточечный

  Каждой паре (q, d) проставлена некоторая численная оценка, и задача сводится к построению регрессии (или классификации, если оценок всего несколько)

- Pair-wise — попарный

  Для двух документов, соответствующих одному запросу, решается задача бинарной классификации (какой из них релевантнее)

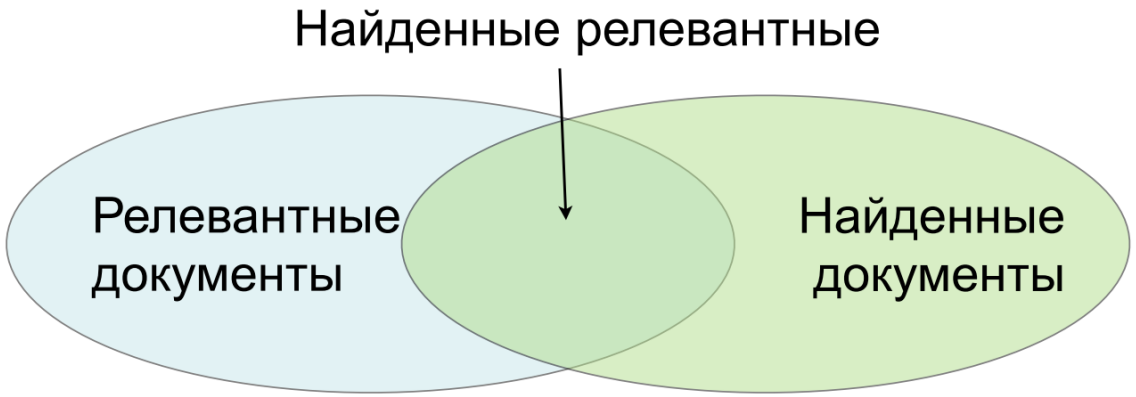
- List-wise — списочный

  На вход поступает список всех документов, на выходе — их перестановка. Модель напрямую оптимизирует одну из описанных выше метрик (точнее, её гладкую аппроксимацию)

Оценка качества

$$\text{Точность } (P) = \frac{\text{количество найденных релевантных}}{\text{количество найденных}}$$

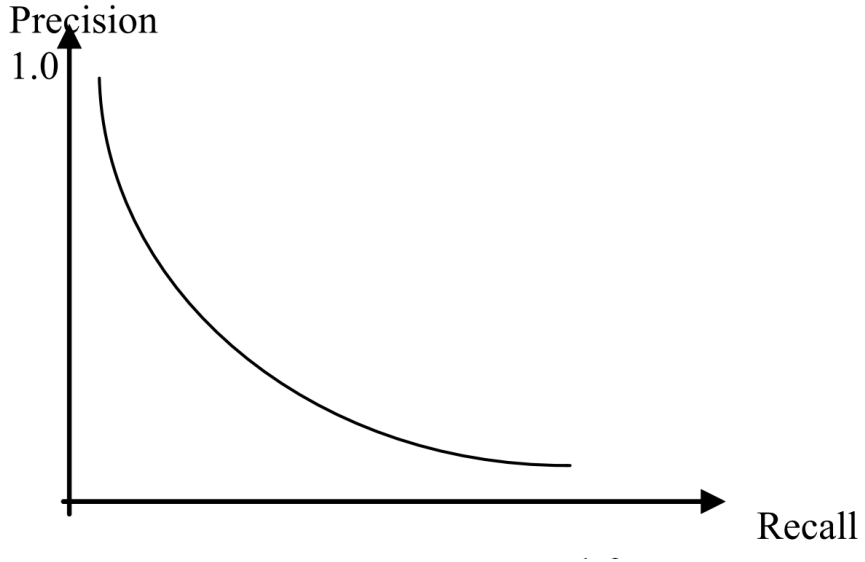
$$\text{Полнота } (R) = \frac{\text{количество найденных релевантных}}{\text{количество релевантных}}$$



Точность и полнота

Общая форма зависимости

- Точность и полнота зависимы
- Системы нельзя сравнивать в одной точке
- Вычисляют среднюю точность (например, в 11 точках с полнотой: 0.0, 0.1, ..., 1.0)



$$AveP = \int_0^1 pr(r)dr$$

$$AveP = \sum_{k=1}^{n} Pr(k)\Delta r(k)$$

$$AveP = \frac{\sum_{k=1}^{n}(Pr(k)\times rel(k))}{\text{число релевантных документов}}$$

$rel(k) \in \{0,1\} = 1$, если k-ый документ релевантен запросу

MAP

Mean Average Precision

$$MAP = \frac{1}{n}\sum_{Q_i}\frac{1}{|R_i|}\sum_{D_j \in R_i}\frac{j}{r_{ij}}$$

- r_{ij} – ранг j-го релевантного документа для Q_i
- $|R_i|$ – число релевантных документов для Q_i
- n – количество тестовых запросов

Ранг		
1	4	1-ый релевантный документ
5	8	2-ый релевантный документ
10		3-ый релевантный документ

$$MAP = \frac{1}{2}\left[\frac{1}{3}\left(\frac{1}{1}+\frac{2}{5}+\frac{3}{10}\right)+\frac{1}{2}\left(\frac{1}{4}+\frac{2}{8}\right)\right] \approx 0.408$$

Обратная связь

- Явная оценка пользователей
- Неявные показатели релевантности:
  - Оценка времени просмотра документа
  - Клики по документам
  - Другие метаданные

Обратная связь

Проблемы обратной связи

- Пользователи часто отказываются от участия в сборе обратной связи
- Неявная оценка менее надежна
- Поисковая потребность может не соответствовать запросу

Web-поиск

Особенности:

- Постоянный рост документов
- Поиск не только текста, но и мультимедиа
- Спам
- Реклама
- Наличие полуструктуры (теги, микроразметка)

Какие задачи решаются в web-поиске?

- Переиндексация (поисковые роботы)
- Хранение
- Генерация сниппетов
- Работа с дубликатами
- Фасеты
- Фильтрация нежелательного контента

Ранжирование web-страниц – Page Rank (1998-2016)

Page Rank (PR) – алгоритм ссылочного ранжирования, основанный на вычислении веса страницы путем подсчета важности ссылающихся на нее документов. Выражается числом.

Являлся одним из показателей авторитетности сайта для поисковой системы Google.

http://www.ams.org/publicoutreach/feature-column/fcarc-pagerank

Page Rank

На странице P_j есть l_j ссылок. Если одна из этих ссылок ведет на страницу P_i, то P_j передаст $\frac{1}{l_j}$ своей важности P_i:

$$I(P_i) = \sum_{P_J \in B_i} \frac{I(P_j)}{l_j}$$

Матрица гиперссылок:

$$H_{ij} = \begin{cases} \frac{1}{l_j}, P_j \in B_i \\ 0, \text{иначе} \end{cases}$$

Page Rank

Вектор значений Page Rank - рейтинг важности всех страниц:

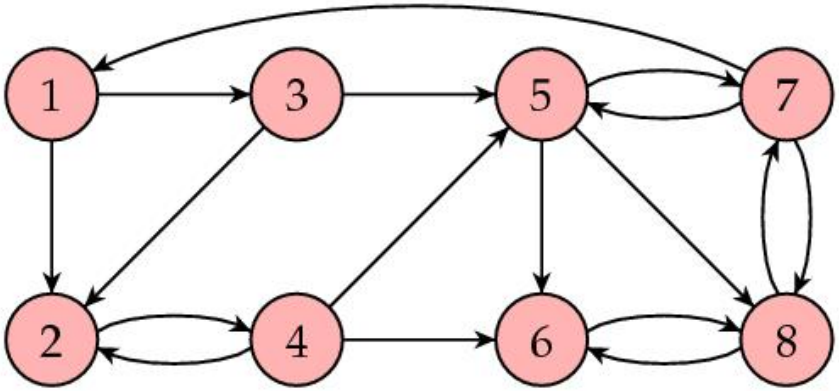
$$I = [I(P_i)]$$

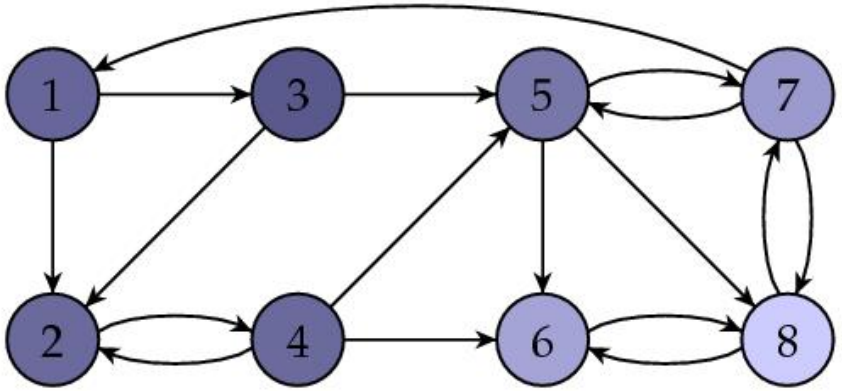
Условие определения PageRank может быть выражено как

$$I = HI$$

Вектор I является собственным вектором матрицы H с собственным значением 1.

Page Rank





$$H = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 0 \\ 1/2 & 0 & 1/2 & 1/3 & 0 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 1/3 & 0 & 0 & 1/3 & 0 \\ 0 & 0 & 0 & 1/3 & 1/3 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/3 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/3 & 1 & 1/3 & 0 \end{bmatrix} \quad \mathrm{I} = \begin{bmatrix} 0.06 \\ 0.0675 \\ 0.03 \\ 0.0675 \\ 0.0975 \\ 0.2025 \\ 0.18 \\ 0.295 \end{bmatrix}$$

Page Rank

$$I^{k+1} = HI^k$$

I^0	I^1	I^2	I^3	I^4	...	I^{60}	I^{61}
1	0	0	0	0.0278	...	0.06	0.06
0	0.5	0.25	0.1667	0.0833	...	0.0675	0.0675
0	0.5	0	0	0	...	0.03	0.03
0	0	0.5	0.25	0.1667	...	0.0675	0.0675
0	0	0.25	0.1667	0.1111	...	0.0975	0.0975
0	0	0	0.25	0.1806	...	0.2025	0.2025
0	0	0	0.0833	0.0972	...	0.18	0.18
0	0	0	0.0833	0.3333	...	0.295	0.295

Случайное блуждание (Random walk)

Page Rank. А оно работает вообще?

- Всегда ли последовательность I^k сходится?
- Зависит ли конечный вектор от начального приближения I_0?
- Содержат ли рейтинги важности необходимую информацию?

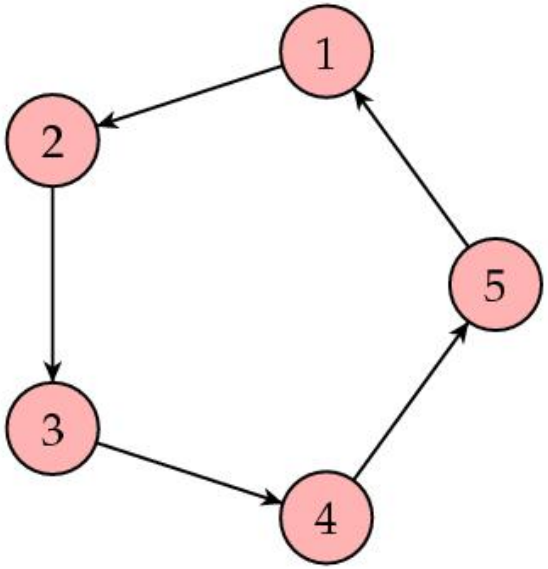
Описанный выше метод даёт отрицательные ответы на все вопросы. Можно модифицировать метод для удовлетворения всех требований.

Page Rank. Переход к вероятностной модели

$$H = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \qquad I = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$S = \begin{bmatrix} 0 & 1/2 \\ 1 & 1/2 \end{bmatrix} \qquad I = \begin{bmatrix} 1/3 \\ 2/3 \end{bmatrix}$$

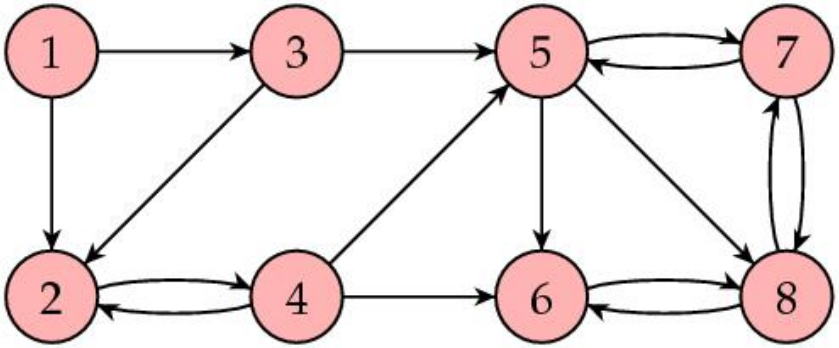
Page Rank. Плохие ситуации

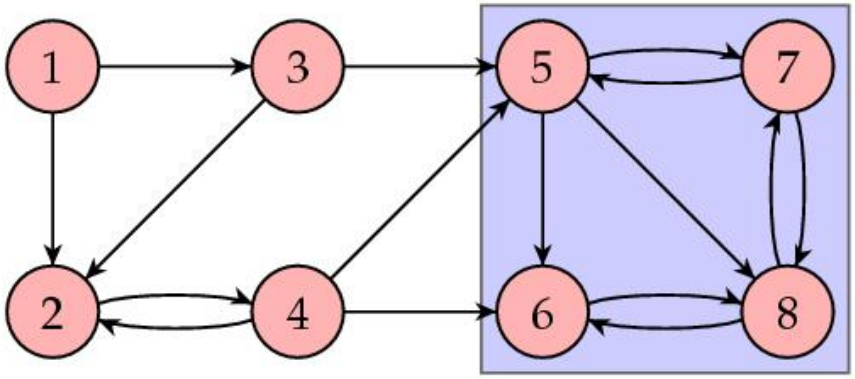


$$S = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

I^0	I^1	I^2	I^3	I^4	I^5
1	0	0	0	0	1
0	1	0	0	0	0
0	0	1	0	0	0
0	0	0	1	0	0
0	0	0	0	1	0

Page Rank. Плохие ситуации





$$S = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1/2 & 0 & 1/2 & 1/3 & 0 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 1/3 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 1/3 & 1/3 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/3 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/3 & 1 & 1/2 & 0 \end{bmatrix} \quad \mathrm{I} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0.12 \\ 0.24 \\ 0.24 \\ 0.4 \end{bmatrix}$$

Page Rank.

$$G = \alpha S + (1 - \alpha)\frac{1}{\mathrm{n}}\Theta$$

Θ – матрица из единиц $n \times n$

α – вероятность использования матрицы S

$Google$ использовал $\alpha = 0.85$

Вопросы?