

Data structures in Information Retrieval

Max Gubin
mail@maxgubin.com

RuSSIR

Russian Summer School
in Information Retrieval

2008

Information Retrieval History



Legal > Cases - U.S. > Federal Court Cases, Combined

Enter Search Terms
 Terms and Connectors Natural Language Easy Search
trade secret /s/ disclose! /s/ prevent /a/ employee
Support Terms For My Search Search
Check Spelling

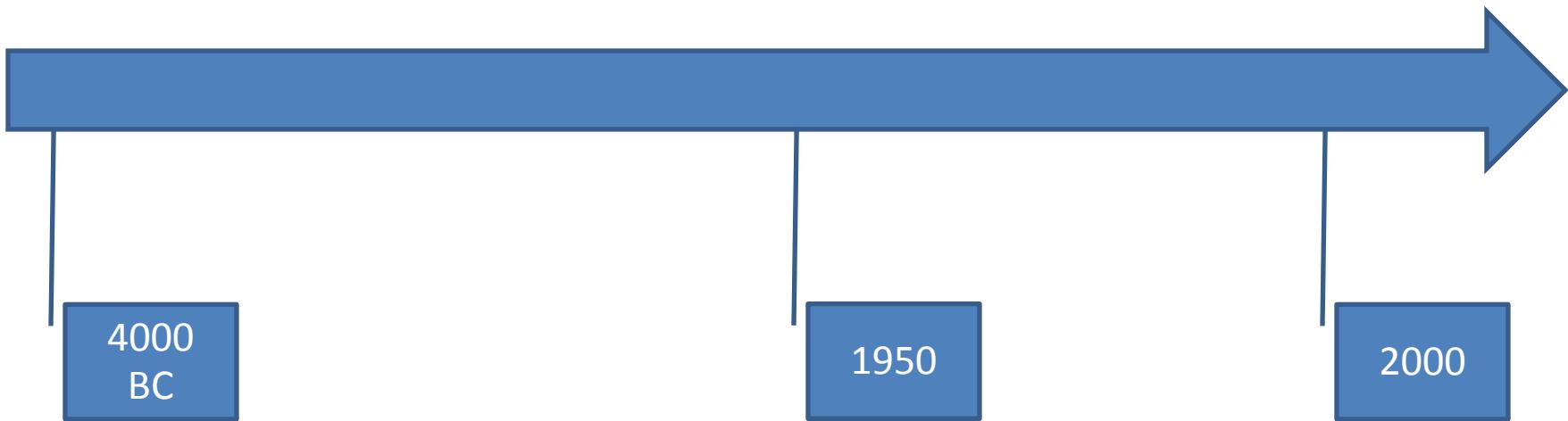
Search Connectors
and and
or or
w/N words
pre-N words
w/in in words
in same paragraph
w/in in same sentence
w/in in same document
and not and not
>More Connectors & Comments...

Restrict by Document Segment:
Select a document segment, enter search terms for the segment, then click Add.
Select a Segment Note: Segment availability differs between sources. Segments may not be applied consistently across sources.

Restrict by Date:
 No Date Restrictions From To Date Formats...

How Do I...?
» Use wildcards as placeholders for one or more characters in a search term
» Restrict by document segment?
» Restrict by date?

View Tutorials



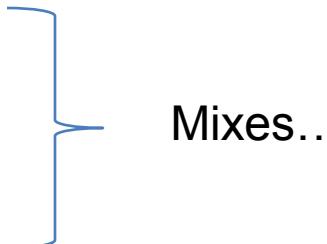
RuSSIR

Russian Summer School
in Information Retrieval

2008

Information Retrieval Tasks

Types of information:

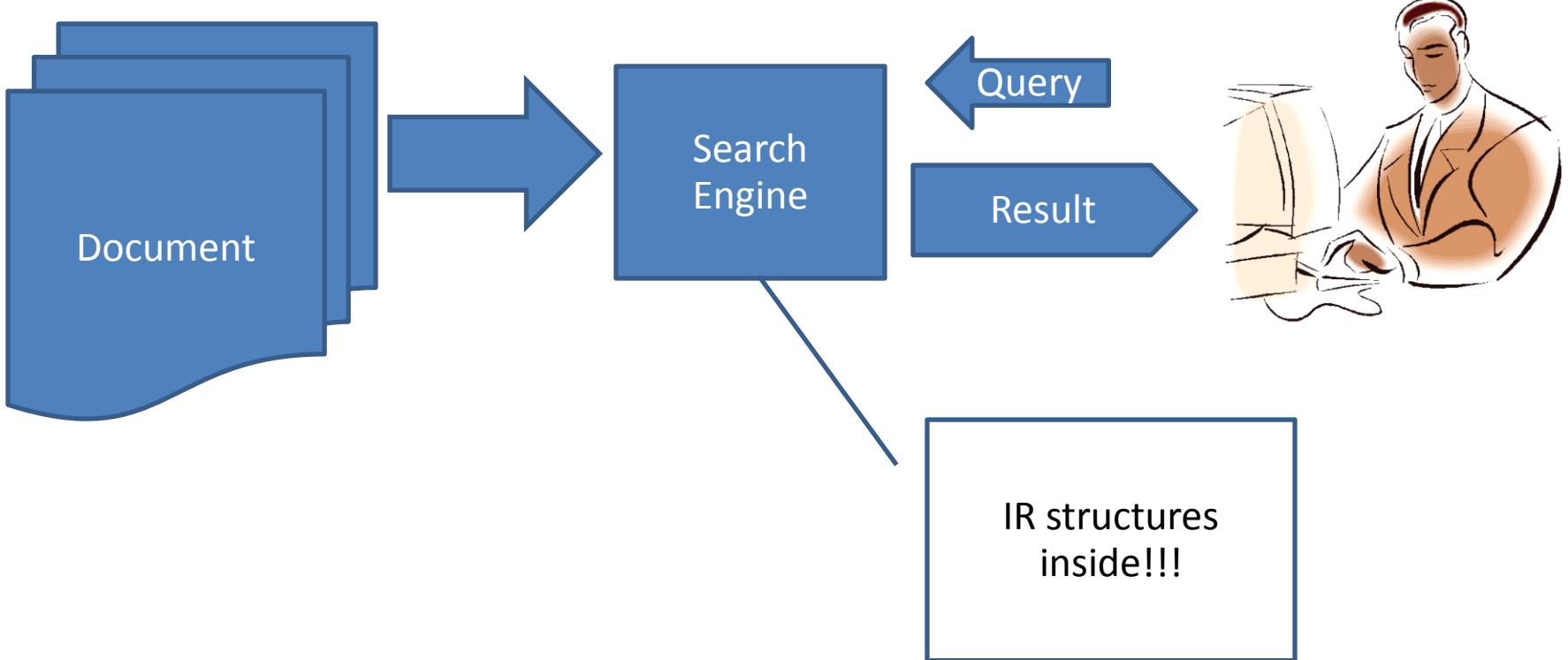
- Text
 - Sound
 - Image
- 
- Mixes...

Types of tasks:

- Search
 - Classification/clustering
 - Extraction/Summarization
- 
- Mixes...

Toy project

Let's create a toy search engine:



Course Outline

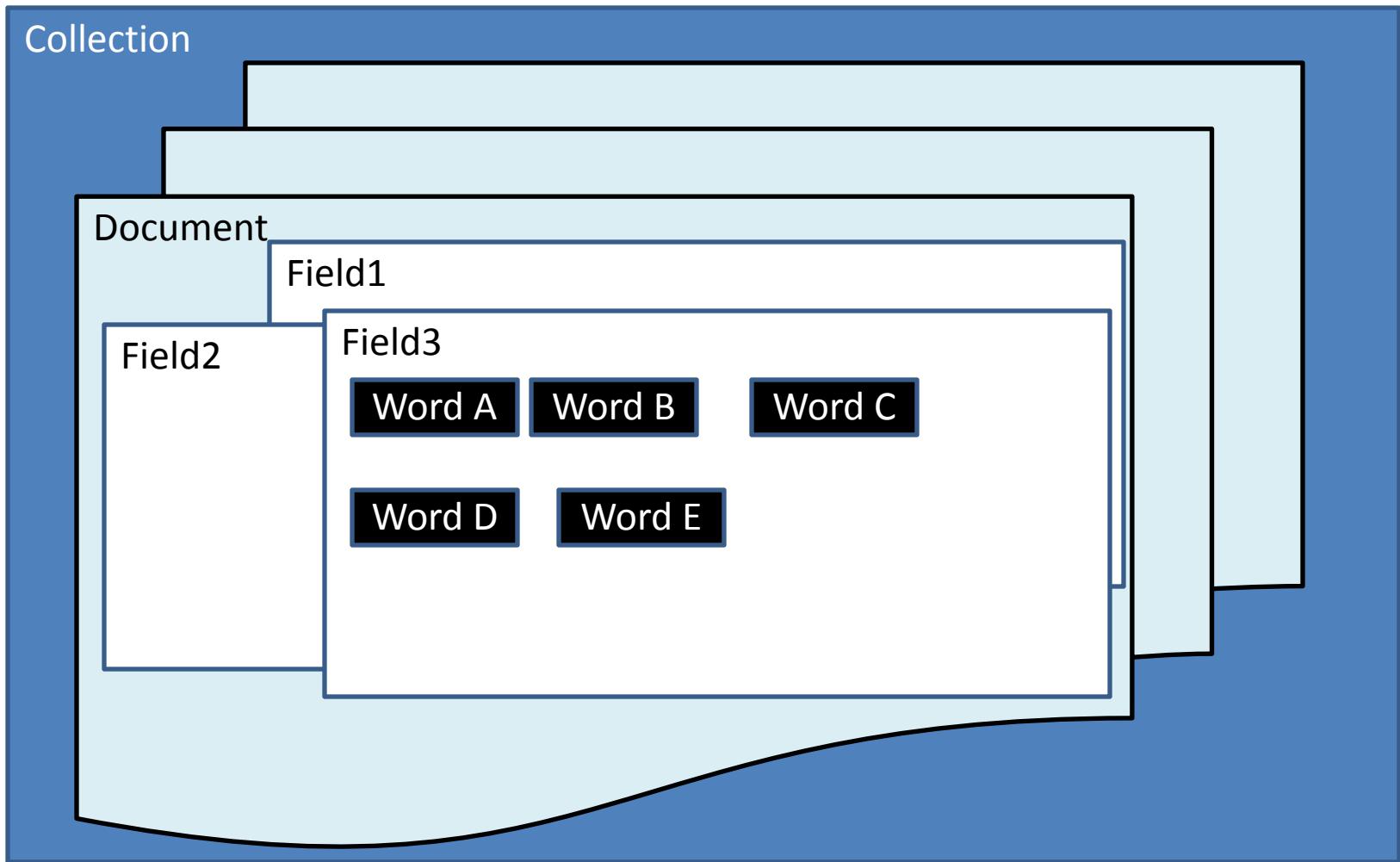
- Introduction (the problem definition)
- Basics (structures and environments)
- Building index
- Search!
- Other data: Language Models and Link Graphs

RuSSIR

Russian Summer School
in Information Retrieval

2008

Hierarchy of data in text IR



Linearization (word extraction)

Влияние морфологического анализа на качество информационного поиска

© М.В. Губин А.Б. Морозов
Консорциум «Кодекс»
max@gubin.spb.ru amoro@kodeks.ru

Аннотация

слова[4]. В литературе данный модуль поисковых систем называют модулем морфологического анализа, нормализатором слов, лемматизатором или стеммером (стеммер). Определив, выплачивающим данный модуль можно представить как отображение:

$$W \rightarrow L$$

где

- W = множество всех терминов;
- L = множество всех тем.

При этом количество темов меньше количества множества всех терминов $|W| > |L|$.

Реализуя запросное преобразование, разработчики поисковой системы пытаются

Слово содержит результат экспериментального исследования в рамках различных подходов к схематизации языка, основанной на критерии информационного поиска. Большинство современных русскоязычных поисковых систем производят нормализацию (лемматизацию) слов, то есть приводят их к единому виду для дальнейшего поисковому признаку. Следует отметить, что это позволяет заметно улучшить качество поиска. Известно множество подходов к нормализации с использованием алгоритмов усечения склонений (стems), алгоритмов морфологического анализа на основе правил и баз словарей. В прошлом исследование было проведено на обширнейшем русскоязычном корпусе общеупотребительных текстов, состоящем из 100 000 страниц. Для сравнения качества поиска использовалась методика ROPMIP.

Введение

В большинстве естественных языках наблюдается такое явление, как морфологическая изменяемость слов[1]. Данные явления сильно выражено в русском языке, который относится к группе флексивных языков со сложной системой флексий[2,3].

Информационно-поисковая система, работающая с русским языком, должна учитывать эту особенность языка, что реализуется обычно с помощью специального модуля системы, называемого модулем морфологического анализа. В данной работе исследуется влияние работы данного модуля на качество информационного

Использование морфологического

В современной поисковой системе модуль морфологического анализа обычно выполняет преобразование множества всех слов языка во множество лемм — нормализованных форм

[4]. В литературе данный модуль поисковой системы называют модулем лексико-семантического языка, нормализатором слов, лемматизатором стемпером (стеммером). Операцию, применяемую данным модулем можно изложить как отображение:

$$W \rightarrow L$$

где

- W – множество всех терминов;
- L – множество всех лемм.

В этом количестве лемм меньше количества всех терминов $|W| > |L|$.

Наиболее лаконичное преобразование, выполняемое поисковой системой, пытается

ищь следующих целей:

1. Увеличение полноты поиска. Так как отбираются документы, которые содержат все формы слова, то в результате поиска попадают не только документы со словом в синтаксической форме, но и другие документы, содержащие различные формы этого слова.

формы ланного слова;

Улучшение точности поиска. При использовании статистических алгоритмов поиска и отбора в результате поиска находятся документы, которые получили наибольший вес, очень важно становится получение частотных характеристик документов. При этом использование вместо частот слов частоты лемм может позволить получить большее вес для релевантных документов и тем самым поместить их во множество отобранных;

3. Упрощение пользовательского интерфейса. Так как для пользователя частой задачей является кэширование «свежих» упоминаний, то в случае отсутствия автоматического

расширение слов его вариантыми пользователь вынужден изучать и использовать функции или операторы отсечения;

4. Уменьшение размера индексной информации и ускорение обработки запроса. Так как количество леммы меньше количества слов, то лемматизация приводит к



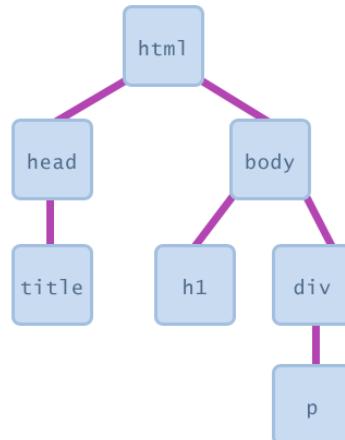
(“To”, 1, Body, Document1)
(“BE”,2, Body,Document1)
(“or”,3,Body,Document1)
(“not”,4, Body,Document1)
(“to”, 5, Body, Document1)
(“be”,6, Body,Document1)

Document formats

- Presentation oriented (PDF, RTF)



- Structure Oriented (SGML, HTML, XML)



RuSSIR

Russian Summer School
in Information Retrieval



Encodings

Official standard Unicode

Latest version 5.10 about 100000 characters:

Character codes (codepoints 0–10FFFF)

Encoding rules (utf-8, utf-16, utf-32)

Algorithms

Words

- Morphology agglutinative, multiroot,
- Abbreviations
- Spelling variants
- Stop-words

How to handle:

1. During document analysis
2. During search

Linearization (complex)

Влияние морфологического анализа на качество информационного поиска

Ф.М. Губин А.Б. Морозов
Консорциум «Кодекс»
maxx@gubin.spb.ru amaxo@codeks.ru

Аннотация

Статья содержит результаты экспериментального исследования влияния различных методов и обработки форм слов на качество информационного поиска. Более всего современные русскоязычные поисковые системы используют нормализацию («нейминг») слов, то есть приведение различных форм слова к одному общему виду. В то же время это не позволяет заметно улучшить качество поиска. Известно несколько подходов к нормализации с использованием лексикона, учётии основной (стеммы), алгоритма морфологического анализа и основании прямой или словарной. В данной статье исследуются различные эксперименты, использующие разные обработки русскоязычных текстов с целью их улучшения в поиске. Для сравнения качества поиска использовалась методика РОДИП.

Введение

В большинстве естественных языков используется такое явление, как морфологическая изменение слов [1]. Данные изменения выражаются в различных формах, которые относятся к группе функциональных единиц со склонкой флексий [2].

Использование таких языковых систем, работосвязанных с русским языком, должно учитывать эту особенность языка, что реализуется обычно с помощью специальных модулей, называемых модулем морфологического анализа. В данной работе исследовано влияние работы данного модуля на качество информационного поиска.

Использование морфологического анализа в поисковой системе

В современной поисковой системе модуль морфологического анализа, как правило, выполняет преобразование множества всех слов языка во множество лемм – нормализованных форм

слов[4]. В литературе леммный модуль поисковой системы называют модулем морфологического анализа, нормализатором слов, лемматизатором или стемматором. Следует отметить, что леммизацию можно представить как обобщение:

для W имеются три варианта:
1. – множества лемм языка

При этом количество лемм меньше количества всех геронимов ($|W|<L$)

Результат лемматизации – преобразование, разработанное поисковой системой, пытающееся достичь следующих целей:

1. Увеличение точности поиска. Так как леммы – это документы, которые содержат все формы слова, то в результате поиска попадают не только документы, содержащие саму лемму с запросом, sondern различные документы, содержащие различные формы слова.

2. Улучшение точности поиска. При использовании статистических методов поиска и отбора в результате получается множество документов, которые получили наибольший ранг отбора, включая документы, не имеющие частично характеристик документов. При этом используется вместо частот слов частота лемм. Для этого необходимо получить большой вак для разнотипных документов и тем самым отобрать документы, не во множестве отображаемых.

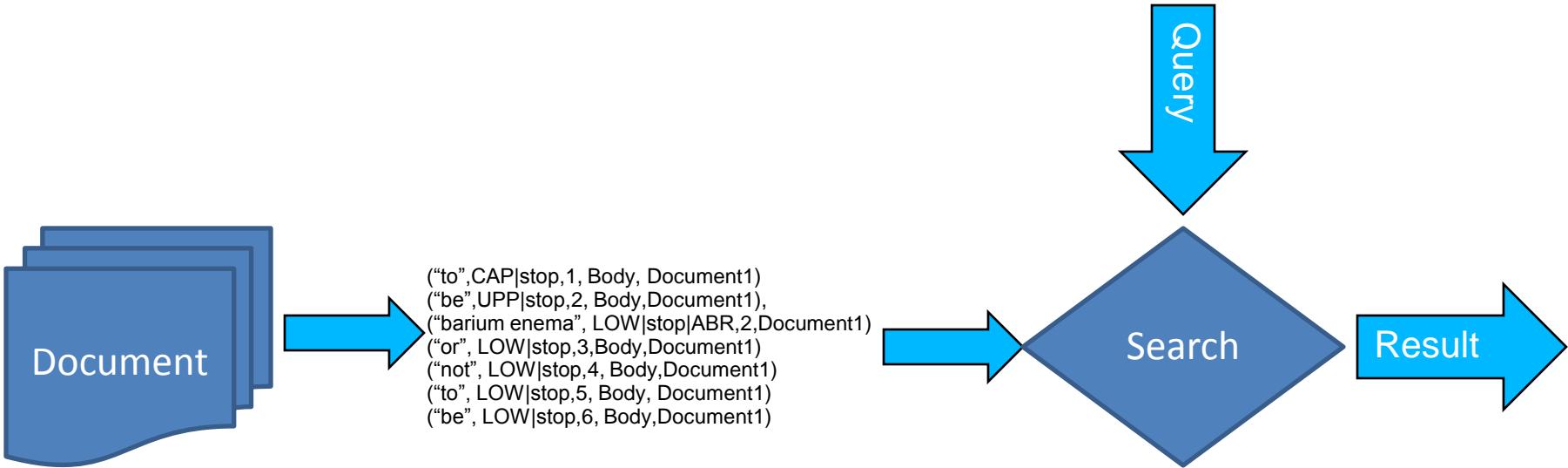
3. Упрощение. Так как для каждого слова существует множество различных форм и множества всех варианты упоминания, то в случае отсутствия автоматического распознавания слов в языке можно использовать вынужденную итерацию и использовать формулы или спиритры для отбора.

4. Уменьшение размера лексической информации и ускорение обработки языка. Так как количество лемм меньше количества слов, то лемматизация приводит к



(“to”,CAP|stop,1, Body, Document1)
(“be”,UPP|stop,2, Body, Document1),
(“barium enema”, LOW|stop|ABR,2, Document1)
(“or”, LOW|stop,3, Body, Document1)
(“not”, LOW|stop,4, Body, Document1)
(“to”, LOW|stop,5, Body, Document1)
(“be”, LOW|stop,6, Body, Document1)

Naïve Scan (grep approach)



- Have the whole context for analysis
- Match current hardware architecture
- Usually can be easily parallelized

Adding index

Two meanings of index:

- Taxonomy that accelerates human search
- **Special data structure** that accelerate data access

Using Standard Database

Dictionary

Word	ID
to	1
be	2
not	4
or	3

Doctable

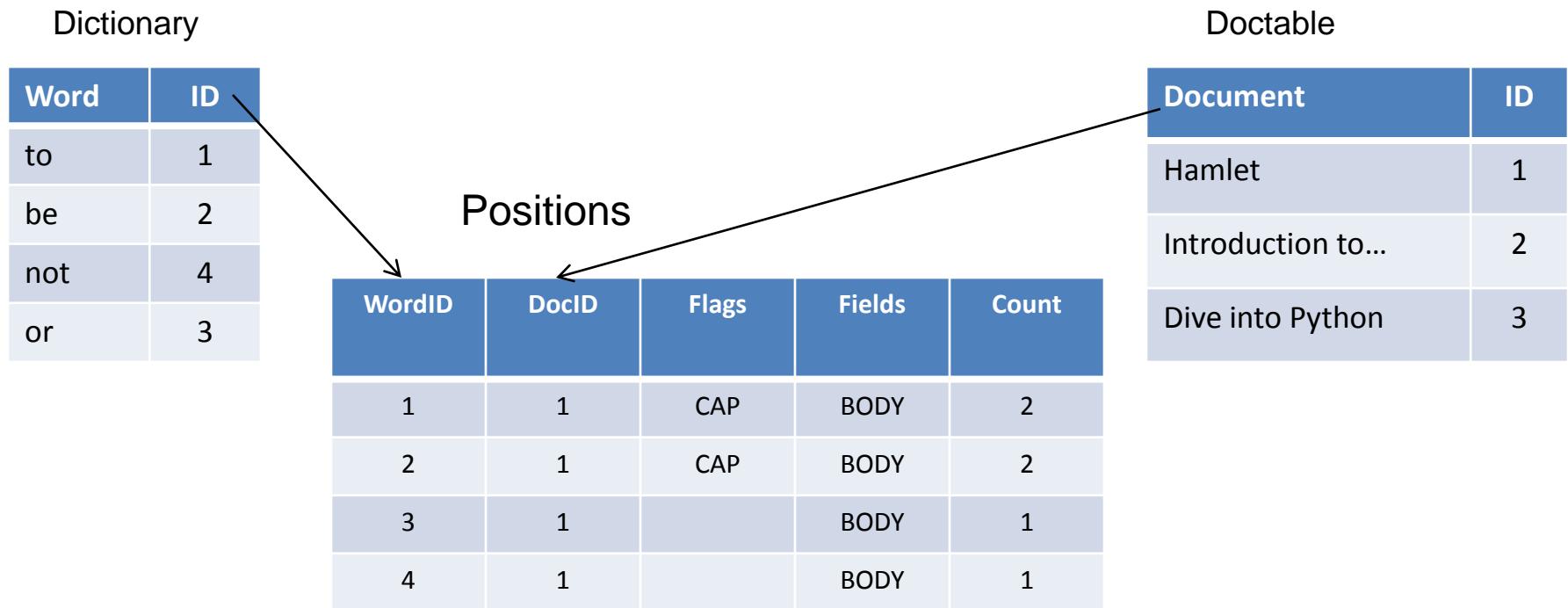
Document	ID
Hamlet	1
Introduction to...	2
Dive into Python	3

Positions

WordID	DocID	Flags	Fields	Pos
1	1	CAP	BODY	1
2	1	CAP	BODY	2
3	1		BODY	3
4	1		BODY	4
1	1		BODY	5
2	1		BODY	6

```
SELECT DocTable.Document FROM Dictionary,Doctable,Positions
WHERE Dictionary.word=? AND Dictionary.ID=Positions.WordID AND
Doctable.ID=Positions.DocID
```

Bag of words



Problems with General Purpose Databases

1. Size
2. Speed build
3. Speed search

This is a tool for another task

RuSSIR

Russian Summer School
in Information Retrieval

2008

Matrix representation

Simple example

1. Dad is reading a book
2. Mom is watching TV
3. Dad and Mom are at home

	1	2	3
a	1	0	0
and	0	0	1
are	0	0	1
at	0	0	1
book	1	0	0
Dad	1	0	1
Mom	0	1	1
is	1	1	0
reading	1	0	0
home	0	0	1
TV	0	1	0

Main IR structure

A sparse n-dimensional matrix in different presentations is
“THE MAIN IR STRUCTURE”

Search – inverted index

Language models – table of probabilities

Link analysis – Adjacency matrix

Sparseness of the matrix

Example:

N - 1 mln documents

D_s - 1000 words/document

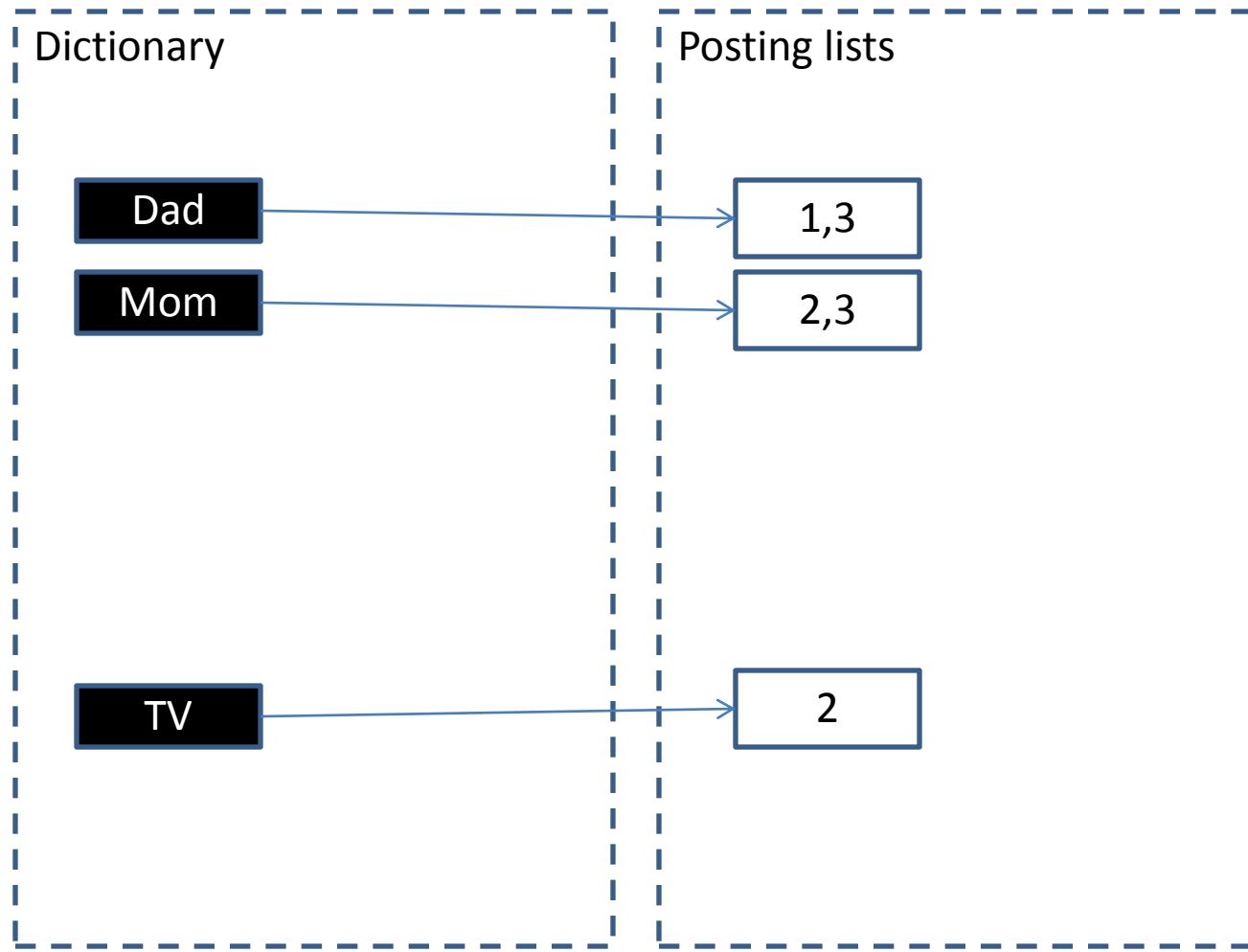
D – 500 000 words in dictionary

|Word/Document matrix| = D*N = 500 bln

Words in collection = 1 mln * 1000 = 1 bln

Only 0.2% elements in the matrix are not 0

Inverted file



Signature file

Signatures for words
(function)

Dad → 00000001

Mom → 00001000

TV → 10000000

watching → 00001000

football → 00001000

Doc Signature = OR words

1	00110001
2	0101 <u>1</u> 000
3	1000 <u>1</u> 001

Signature file (Search)

Query = "Mom Dad"

q_s = 00001001

```
for doc in Document_Signatures:  
    if doc.signature & q_s = q_s:  
        ScanDocument(doc.id)
```

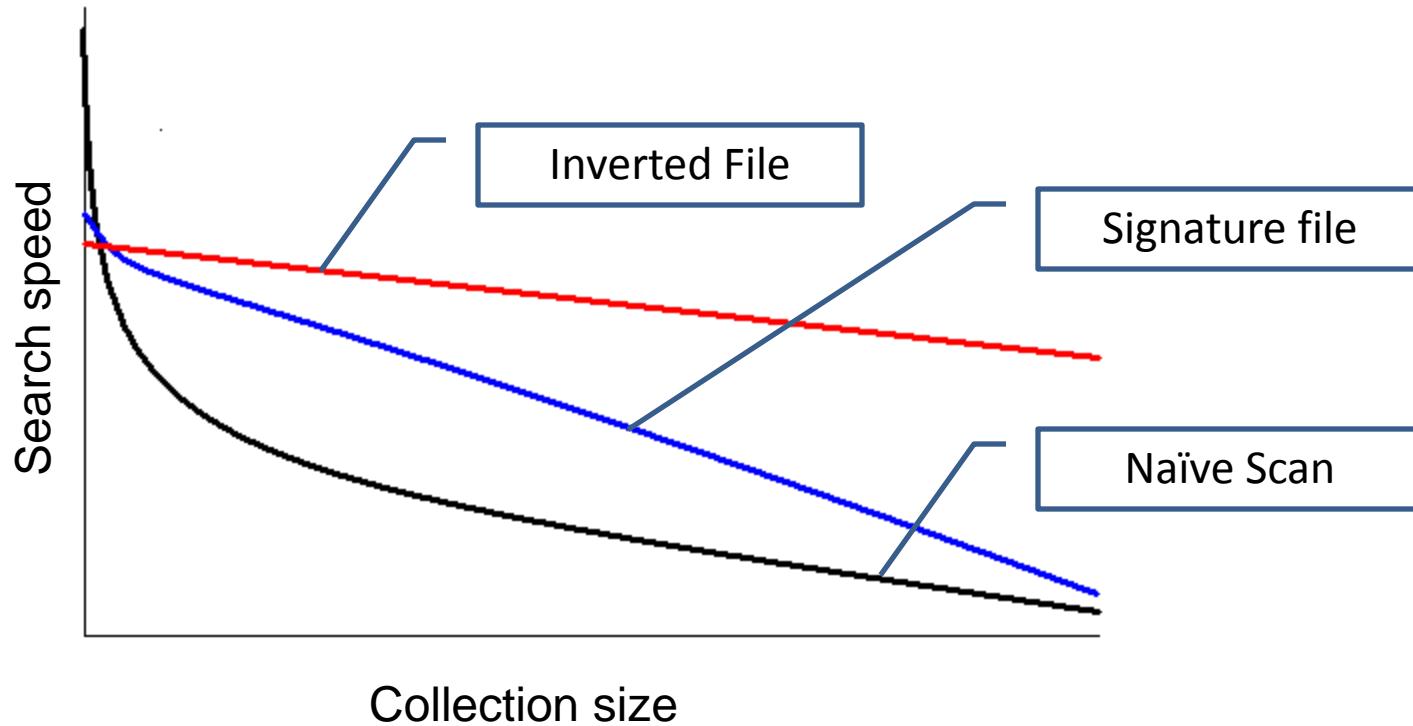
1	001 <u>10001</u>
2	010 <u>11000</u>
3	1000 <u>1001</u>

An old structure = hash + bloom filter + scan

IR Packages

- Lucene (<http://lucene.apache.org/>)
- Terrier (<http://ir.dcs.gla.ac.uk/terrier/>)
- Lemur & Indri (<http://www.lemurproject.org/>)
- Zettair (<http://www.seg.rmit.edu.au/zettair/>)
- Zebra (<http://www.indexdata.dk/zebra/>)

Search speed



Speed (Size) depends on

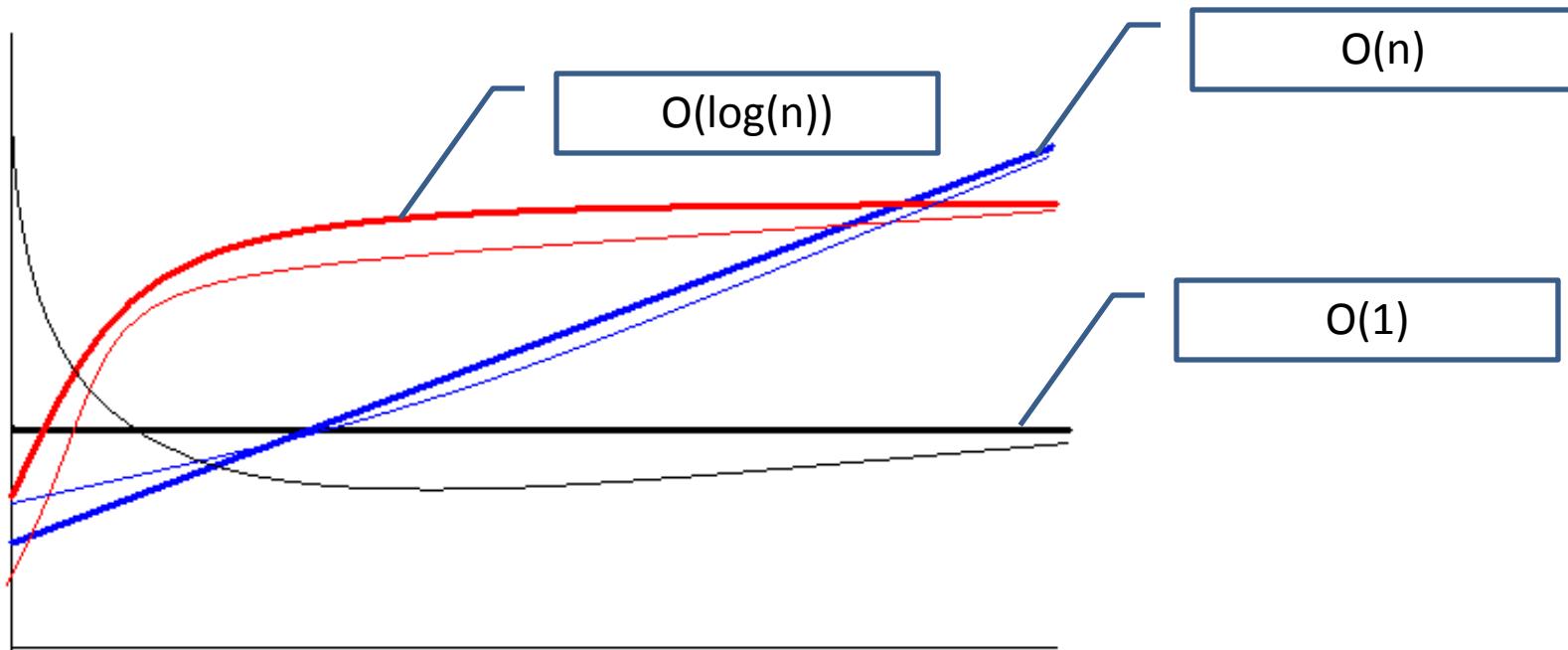
- Algorithm
- Size of data
- Hardware

Algorithm complexity

- Storage complexity (How much memory we need)
- Time complexity (How many operations we need)

$O(f(n))$ notation

$x(n)$ is $O(f(n))$ if $x(n) \leq C * f(n)$, C – const $n \rightarrow \infty$



Structure characteristics

- **Theoretical:** Processing algorithm complexity



=



- **Practical:**
 - Memory access pattern
 - Parallelization

Summary

- IR is old ☺
- Main Structure is sparse matrix
- Index = Inverted file
- Speed & Size

Q&A