Основы обработки текстов

лекция 1

О курсе

- Лектор: **Турдаков Денис Юрьевич**
- Лекции каждую **среду в 10.30 ауд. 523**
  - предполагаются минимальные знания
    - линейной алгебры,
    - теории вероятности и математической статистики
    - программирования
  - не все имеют одинаковые знания
    - предполагается, что студенты могут быстро учиться

План на сегодня

- Подробнее о курсе и практикуме
- Язык программирования Python
- Проблемы обработки текстов

Обработка текстов

Часть 1

О курсе

- Курс состоит из
  - –лекций,
  - –практической части и
  - –итогового экзамена
- Язык программирования Python
- Вся информация: http://tpc.at.ispras.ru

Практическая часть

- Одна из открытых задач обработки текстов
  - В этом году: определение оскорблений участников дискуссий в Интернете
- Веб-интерфейс для проверки и задание будут доступны через две недели

Обработка текстов

Часть 2

Python

- Значимые пробелы

```
if x==1:
    print ‘x is 1’
    print ‘внутри блока’
print ‘вне блока’
```

Python

- Конструкторы списков и словарей

```
number_list=[1,2,3,4]
string_list=['a','b','c','d']
mixed_list=['a',2,'c',7]
```

- Словарь (ключ/значение)

```
ages={'John':34, 'Sarah':20, 'Max':24}
```

- Доступ к элементам []

```
string_list[3]  # 'c'
ages['Sarah'] # '20'
```

Python

- Трансформация списков
  - [выражение for переменная in список]
  - [выражение for переменная in список if условие]

```
l1=[1,2,3,4,5,6,7,8,9]
print [v*10 for v in l1 if v>4]
 > [50, 60, 70, 80, 90]
```

- Функции:
  - –map, filter, zip

```
print filter(lambda x: x > 1, [0,1,2,3])
 > [2,3]
print zip([1,2],[3,4])
 > [(1, 3), (2, 4)]
```

Python и обработка текстов

- NLTK
- http://www.nltk.org
- NLTK book

```
import nltk
text = "Hello world!"
tokens = nltk.word_tokenize(text)
print tokens

> ['Hello', 'world', '!']
```

Python и машинное обучение

- scikit-learn
- http://scikit-learn.org

```
from sklearn.naive_bayes import GaussianNB
x = [[0,0],[1,1]]
y = [0,1]

classifier = GaussianNB()
trained_classifier = classifier.fit(x,y)
predicted_value = trained_classifier.predict([0.6,0.6])

  > [1]
```

Обработка текстов

Часть 3

Классические задачи обработки текстов

- Информационный поиск (IR)
- Извлечение информации (IE)
- Вопросно-ответные системы (QA)
- Классификация и кластеризация
- Автоматическое аннотирование и реферирование
- Диалоговые системы
- Машинный перевод

Приложения обработки текстов

Что нужно знать о тексте?

- Рассмотрим приложение
  - **Siri**: интеллектуальный ассистент на iPhone

Уровни обработки текстов

- Морфологический
  - I’m - I am
  - кошка-кошки, дно-?
- Синтаксический
  - Мне один черный кофе и один сладкий булка...
- Семантический
  - Сколько китайского шелка было экспортировано в Западную Европу в конце 18 века?
  - лексическая и композиционная семантика
- Прагматический (дискурс)
  - Сколько тогда было штатов в США?
  - установление кореферентности (coreference resolution)

Многозначность

- Ключевая проблема обработки текстов
- Я траву **косил косой**,

  Дождик вдруг пошел **косой**.

  Бросил я тогда **косить**

  И на Стешу стал **косить**.

  Ну а Стеша,ох,краса,

  Как огонь её **коса**!

Многозначность

- Морфологическая
  - –часть речи
  - –мой (-- нос, -- руки)
  - –look (look at me, have a look)
- Синтаксическая
  - –мужу изменять нельзя
  - –мать любит дочь
  - –Flying planes can be dangerous

Алгоритмы определения частей речи (part of speech tagging)

Синтаксический разбор (parsing)

Многозначность

- Лексическая (семантическая)
  - –Омонимия (ключ)
  - –полисемия (платформа)
  - –семантическая многозначность (лиса)
- Прагматическая
  - –Огонь! (в армии или в комнате)
  - –You have a green light

разрешение лексической многозначности (word sense disambiguation)

Многозначность и перевод

- **Help для Windows 95**

  ... Мышь может неадекватно реагировать на щелчок по почкам. Но не спешите! Это могут быть физические проблемы, а не клоп Окон 95.

  Почистите вашу мышь.
  Отсоедините ее поводок от компьютера, вытащите гениталий и промойте его и ролики внутренностей спиртом.
  Снова зашейте мышь.
  Проверьте на переломы поводка.
  Подсоедините мышь к компьютеру.
  Приглядитесь к вашей прокладке (подушке) - она не должна быть источником мусора и пыли в гениталии и роликах.
  Поверхность прокладки не должна стеснять движения мыши.

  ...

Сложность языка

- Естественный язык:
  - –многозначен на всех уровнях
  - –сложное, едва уловимое использование контекста для передачи значения
  - –включает знания и рассуждения о мире
- Но обработка естественного языка может быть иногда очень простой
  - –использование грубых признаков часто позволяют достичь очень хороших результатов

Понимание

- Тест Тьюринга
- Китайская комната
- ELIZA (1966)
  - „I am unhappy.“
  - „How long have you been unhappy?“
  - „Six months. Can you help me?“
  - „What makes you think I help you?“
  - „You remind me of my sister.“
  - „Can you tell me more about your sister?“
  - „I like teasing my sister.“
  - „Does anyone else in your family like teasing your sister?“
  - „No. Only me.“
  - „Please go on.“

Текущее состояние

- Разговорные агенты используются некоторыми авиакомпаниями
- Можно отдавать голосовые команды устройствам (телефон, в автомобиле...)
- Многоязыковой информационный поиск Google
- Перевод страниц Google
- Компании занимающиеся анализом текстов позволяют анализировать мнения и предпочтения людей

Новые взгляд на старые проблемы

- Информационный взрыв и масштабируемость (big data)
- Обработка сообщений в социальных сетях и Интернете в целом
- Автоматическое извлечение знаний из текста

Резюме

- Хороший способ понять проблемы обработки текстов - сделать систему машинного перевода, вопросно-ответную систему или разговорного агента
- Обработка текста основана на формальных моделях
- Основы обработки текста лежат в компьютерных науках, математике, лингвистике, электротехнике и психологии
- Сейчас - удивительное время, когда революционные разработки используются повсеместно

Дополнительные ресурсы

- Конференции: ACL, EACL, COLING, CoNLL, EMNLP, Диалог
- Журналы: Computational Linguistics, Natural Language Engineering, Speech & Language Processing
- http://www.aclweb.org/anthology-new/
- Книги:
  - D. Jurafsky, J.H. Martin. Speech and Language processing.
  - C. Manning, H. Schutze. Foundations of Statistical Natural Language Processing
- Курс Stanford NLP: http://see.stanford.edu/

Следующая лекция

- регулярные выражения
- конечные автоматы

Обработка текстов

Лекция 2

Регулярные выражения

Базовые задачи обработки текстов

Мотивация

- Обновить цену товара в прайс-листе:

  для конкретного товара за 1000р. сделать 999.99р.
- Заменить все вхождения одного слова в тексте на другое
  - для части слова (Википедия -> Энциклопедия)
  - с учетом контекста
- Найти сообщения о терроризме
- Фильтрация нецензурных высказываний на форумах

Регулярные выражения

- Regular Expressions (RegExp)
- Языки программирования (Python, Perl, Ruby, Java, .Net)
- Текстовые редакторы (Vim, EmEdit)
- Утилиты (grep, sed)

Регулярные выражения

- Регулярные выражения - алгебраическая нотация для записи множества строк
- Функции Python

```
import re
re.search("в","пиво").group(0) # в
re.sub("о","ко","пиво") # пивко
re.findall("cd","abcdcde") # [“cd”,”cd”]
```

Регулярные выражения

- Последовательность букв: *abcd*
- Чувствительны к регистру: *“Пиво”≠ “пиво”*
- Дизъюнкция: *[П|п]иво,[abc],[1234567890]*
- Интервал: *[A-Z], [0-9], [A-Za-z]*

```
for letter in re.findall("[а-о]","пиво"):
    print letter,
> и в о
```

- Знак ^: *[^a]* - все кроме *“a”*
- “.” - любой символ, кроме \n

Регулярные выражения

- ? - условие для 0 или 1 вхожденя символа

```
re.findall("пивк?о", "пиво или пивко")
> [“пиво”,”пивко”]
```

- Как найти “Gooooogle”?
- Счетчики
  - –Gooo*gle
  - –Goo+gle

```
print re.sub("Goo+gle","Google","Gooooogle")
> Google
```

Регулярные выражения

- Якоря

  ^ - начало строки

  $ - конец строки

```
re.search("^cat(1|2)","cat1 cat2").group(0)
> cat1
re.search("cat(1|2)$","cat1 cat2").group(0)
> cat 2
```

Регулярные выражения

- Память (Memory)

```
text="A conditional random field (CRF) is a ..."
print re.search("\([^\)]+\)",text).group(0)
> (CRF)
print re.search("\(([^\)]+)\)",text).group(1)
> CRF
```

```
text1 = "the faster they ran, the faster we ran"
text2 = "the faster they ran, the faster we ate"
re.search("the (.*)er they (.*), the \\1er we \\2", text1) # Match
re.search("the (.*)er they (.*), the \\1er we \\2", text2) # Not match
```

- Приоритет операций

Круглые скобки	()
Счетчики	* + ? {}
Последовательности и якоря	the ^my end$
Дизъюнкция	|

Практические примеры

- Найти все вхождения слова в тексте
  - слово the в тексте 6 раз

Практические примеры

- ELIZA

User: Men are all alike.
ELIZA: IN WHAT WAY?
User: They’re always bugging us about something other
ELIZA: CAN YOU THINK OF A SPECIFIC EXAMPLE?
User: My boyfriend made me come here
ELIZA: YOUR BOYFRIEND MADE YOU COME HERE
User: He says I’m depressed much of the time
ELIZA: I AM SORRY TO HEAR YOU ARE DEPRESSED

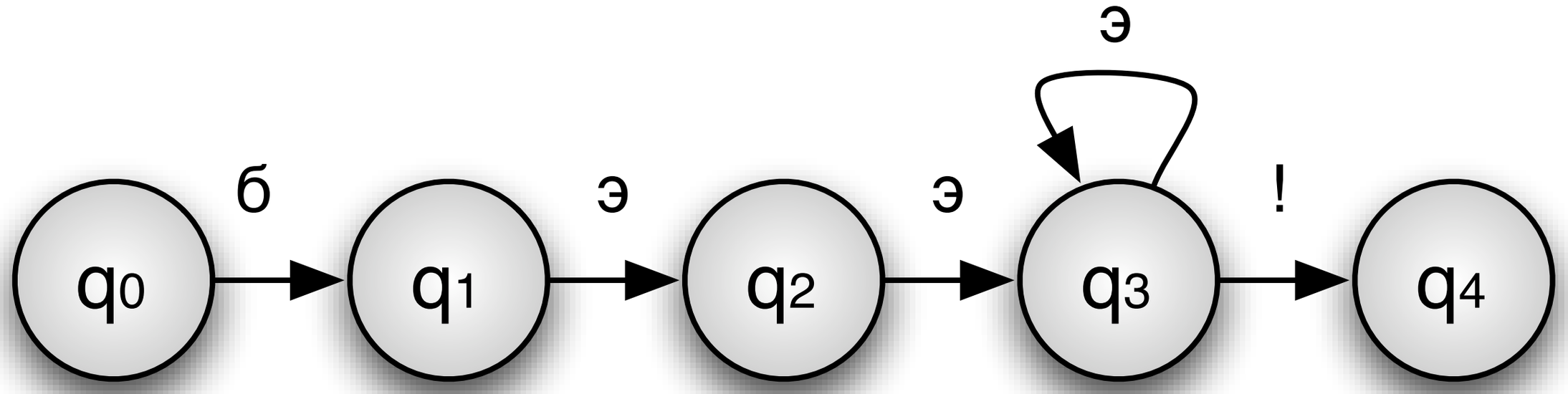
```
sentence=re.sub("I’m","YOU ARE", sentence)
...
sentence=re.sub(".* YOU ARE (depressed|sad) .*","I'M SORRY TO HEAR YOU ARE \\1",sentence)
sentence=re.sub(".* all .*","IN WHAT WAY?",sentence)
sentence=re.sub(".* always .*","CAN YOU THINK OF A SPECIFIC EXAMPLE?",sentence)
```

Конечные автоматы

- Finite-state automation (FSA)
- Один из важнейших инструментов для обработки текстов
- Могут быть использованы для реализации регулярных выражений

Использование КА для распознавания языка

- Научимся говорить с овцами
  - –бээ!
  - –бээээ!
  - –бээээээ!
  - –бээээээээ!
  - –...
- RE: “бээ+!”



Представление автоматов

- Текст: лента с ячейками

- Таблица переходов между состояниями

Состояние	Вход		
	б	э	!
0	1	∅	∅
1	∅	2	∅
2	∅	3	∅
3	∅	3	4
4	∅	∅	∅

Формальное определение

$Q = q_0 q_1 q_2 \ldots q_{N-1}$	конечное множество из N **состояний**
Σ	конечный **входной алфавит**
q_0	**начальное состояние**
F	множество **конечных состояний**
$\delta(q, i) : Q \times \Sigma \rightarrow Q$	**функция перехода** или матрица перехода между состояниями

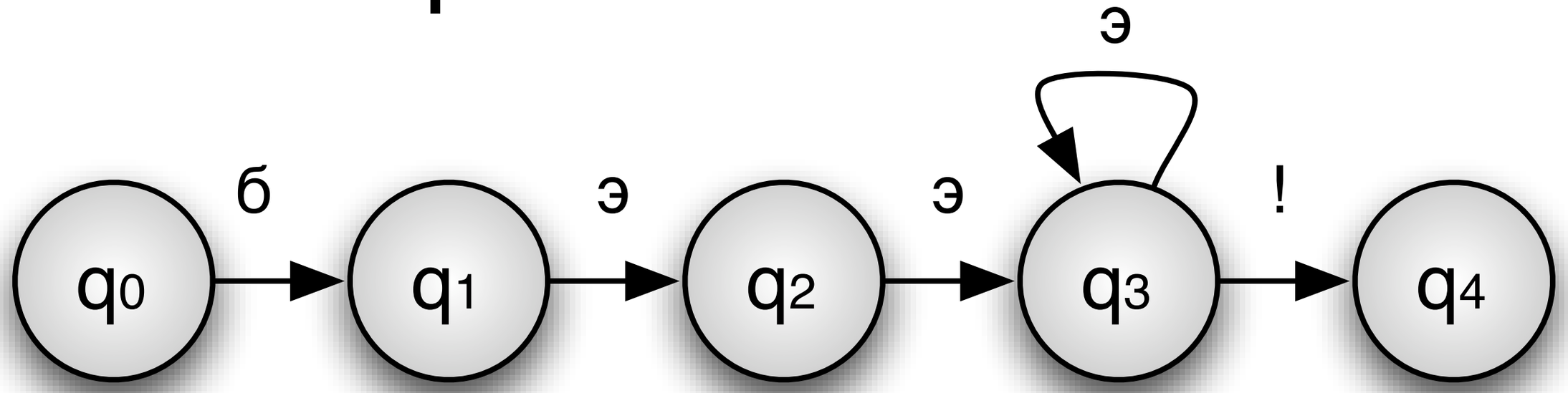
Алгоритм распознавания для детерминированного КА

```
#encoding=CP1251
def recognize(tape, machine, acceptStates):
    index = 0 # Beginning of tape
    currentState = 0 # Initial state of machine
    while True:
        if index == len(tape):
            if currentState in acceptStates:
                return True
            else:
                return False
        elif machine[currentState].has_key(tape[index]):
            currentState = machine[currentState][tape[index]]
            index+=1
        else:
            return False

machineSheep = {0:{"б":1}, 1:{"э":2}, 2:{"э":3}, 3:{"э":3,"!":4}, 4:{}}
print(recognize("бэээээээ!",machineSheep, [4]))
```

	Вход		
	б	э	!
0	1	∅	∅
1	∅	2	∅
2	∅	3	∅
3	∅	3	4
4	∅	∅	∅

Формальные языки



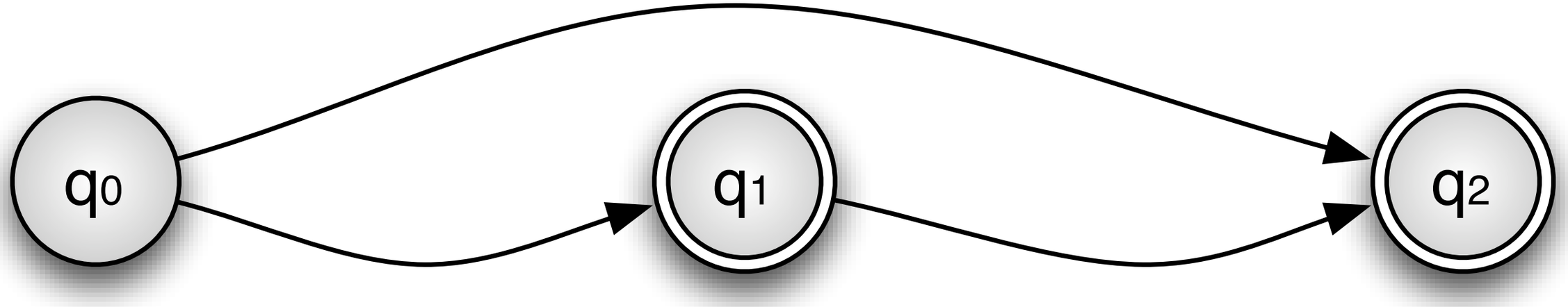
- **формальный язык** — это множество конечных слов (строк, цепочек) над конечным алфавитом

$$\Sigma = \{a, b, !\}$$
$$L(m) = \{baa!, baaa!, baaaa!, \ldots\}$$

Пример формального языка

один	шесть
два	семь
три	восемь
четыре	девять
пять	десять

одиннадцать	пятнадцать
двенадцать	шестнадцать
тринадцать	семнадцать
четырнадцать	восемнадцать
	девятнадцать



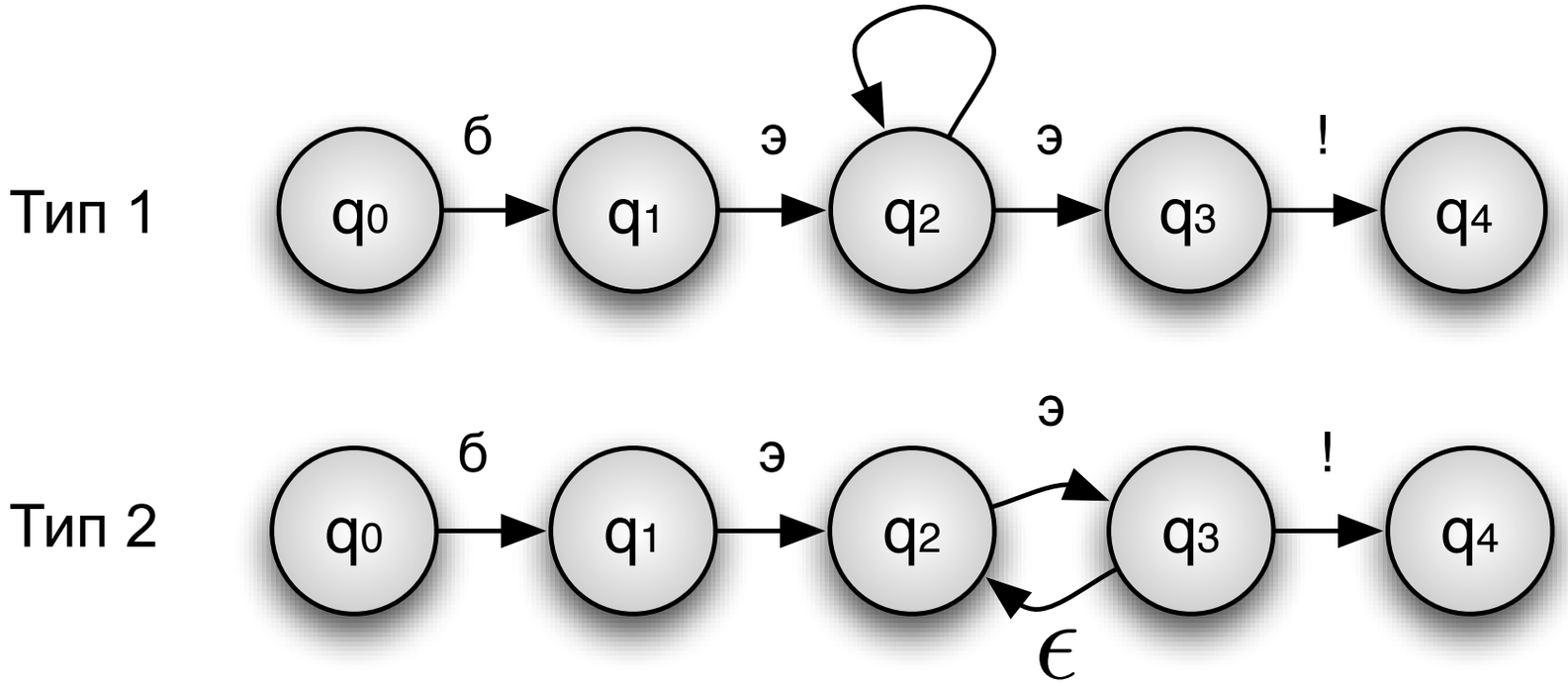
двадцать	шестьдесят
тридцать	семьдесят
сорок	восемьдесят
пятьдесят	девяносто

один	шесть
два	семь
три	восемь
четыре	девять
пять	

Недетерминированные КА

- Обобщение ДКА
- Недетерминизм двух типов

э



Распознование для НКА

- Подходы к решению проблемы недетерминизма
  - –Сохранение состояний (backup)
    - поиск в глубину и ширину
  - –Просмотр будующих состояний (look-ahead)
  - –Параллелизм

Состояние	Вход			
	б	э	!	ϵ
0	1	∅	∅	∅
1	∅	2	∅	∅
2	∅	2,3	∅	∅
3	∅	∅	4	∅
4	∅	∅	∅	∅

ДКА и НКА

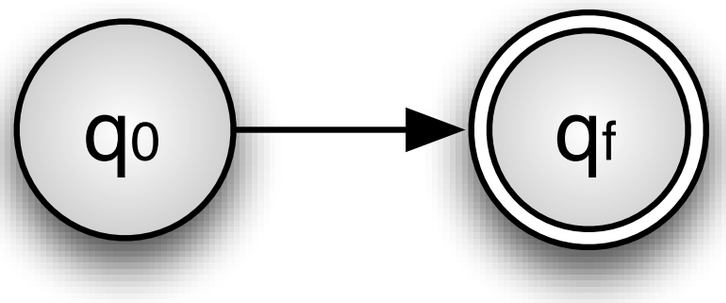
- ДКА и НКА эквивалентны
- Существует простой алгоритм для преобразования НКА в ДКА
- Идея:
  - взять все параллельные ветки НКА
  - в них взять все состояния, в которых одновременно может находиться НКА
  - объединить их в новое состояние ДКА
- В худшем случае НКА с N состояниями преобразуется в ДКА с 2^N состояниями

Регулярные языки и ДКА

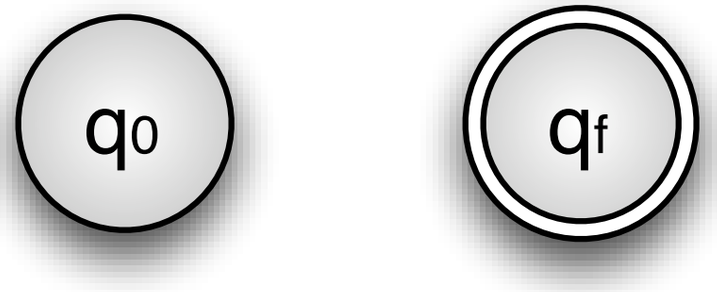
1. $\varnothing$ - регулярный язык
2. $\forall a \in \Sigma \cup \epsilon, \{a\}$ - регулярный язык
3. Для любых регулярных языков L_1 и L_2, такими также являются:
   (a) $L_1 \cdot L_2 = \{xy \mid x \in L_1, y \in L_2\}$, соединение L_1 и L_2
   (b) $L_1 \cup L_2$, объединение или дизъюнкция L_1 и L_2
   (c) L_1^*, замыкание (Клини) языка L_1

- регулярные языки также замкнуты относительно операций
  - пересечения
  - разности
  - дополнения
  - инверсии

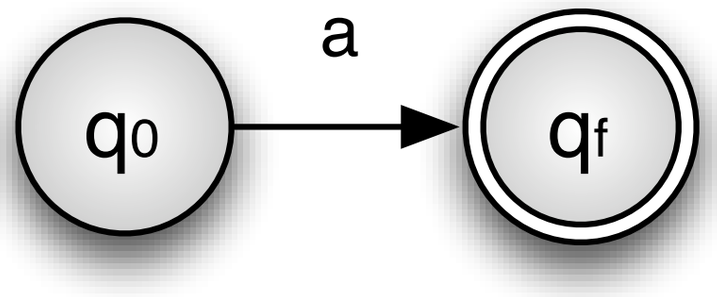
Построение автомата для регулярных выражений



$$r = \epsilon$$

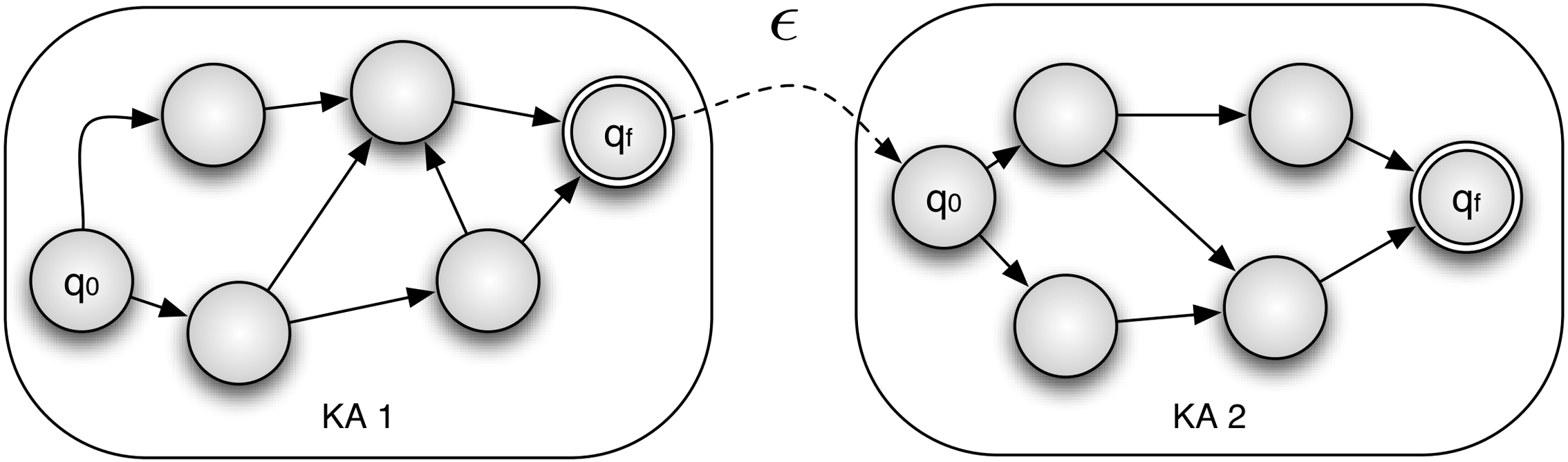


$$r = \varnothing$$



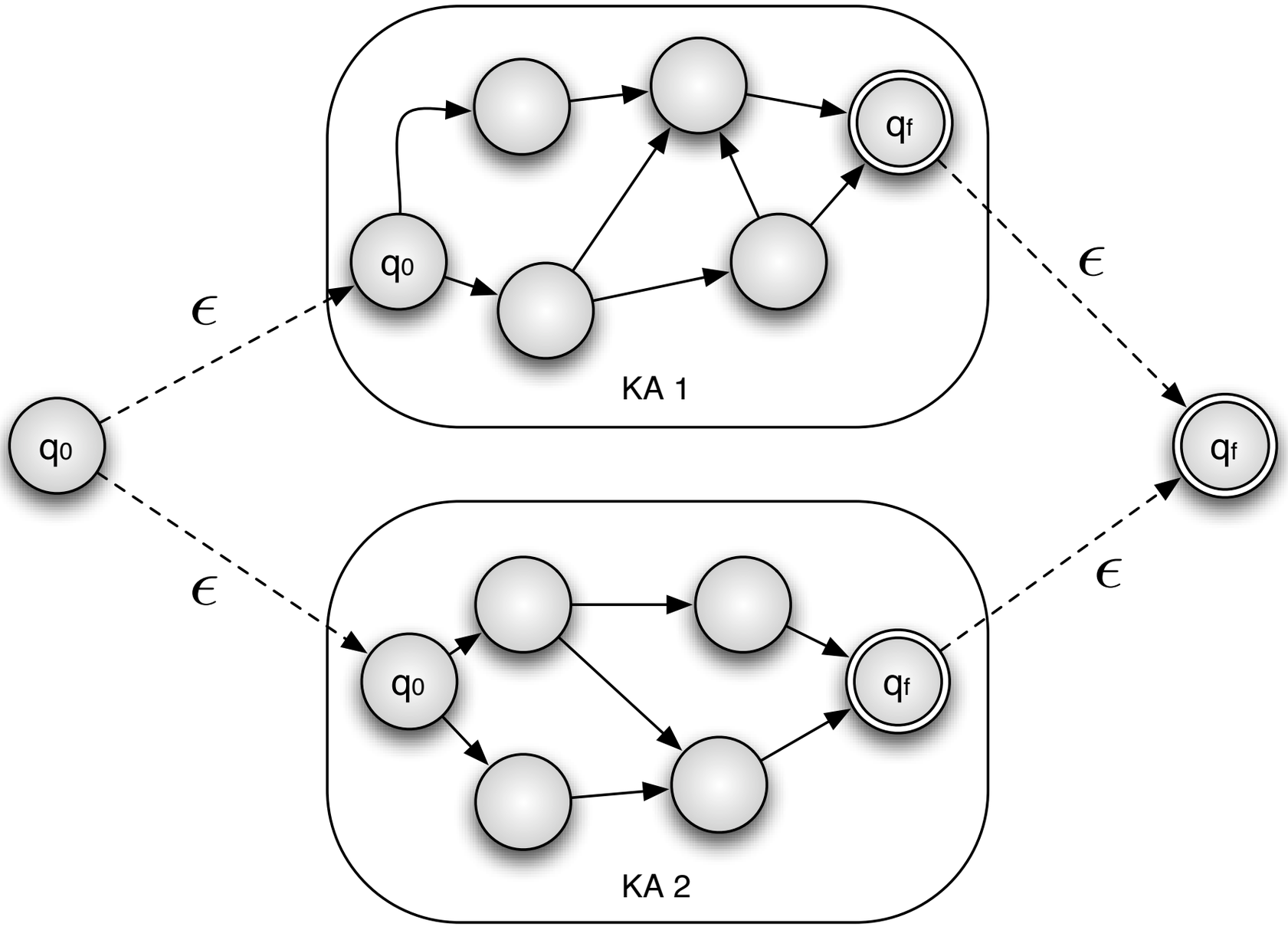
$$r = a$$

Построение автомата для регулярных выражений



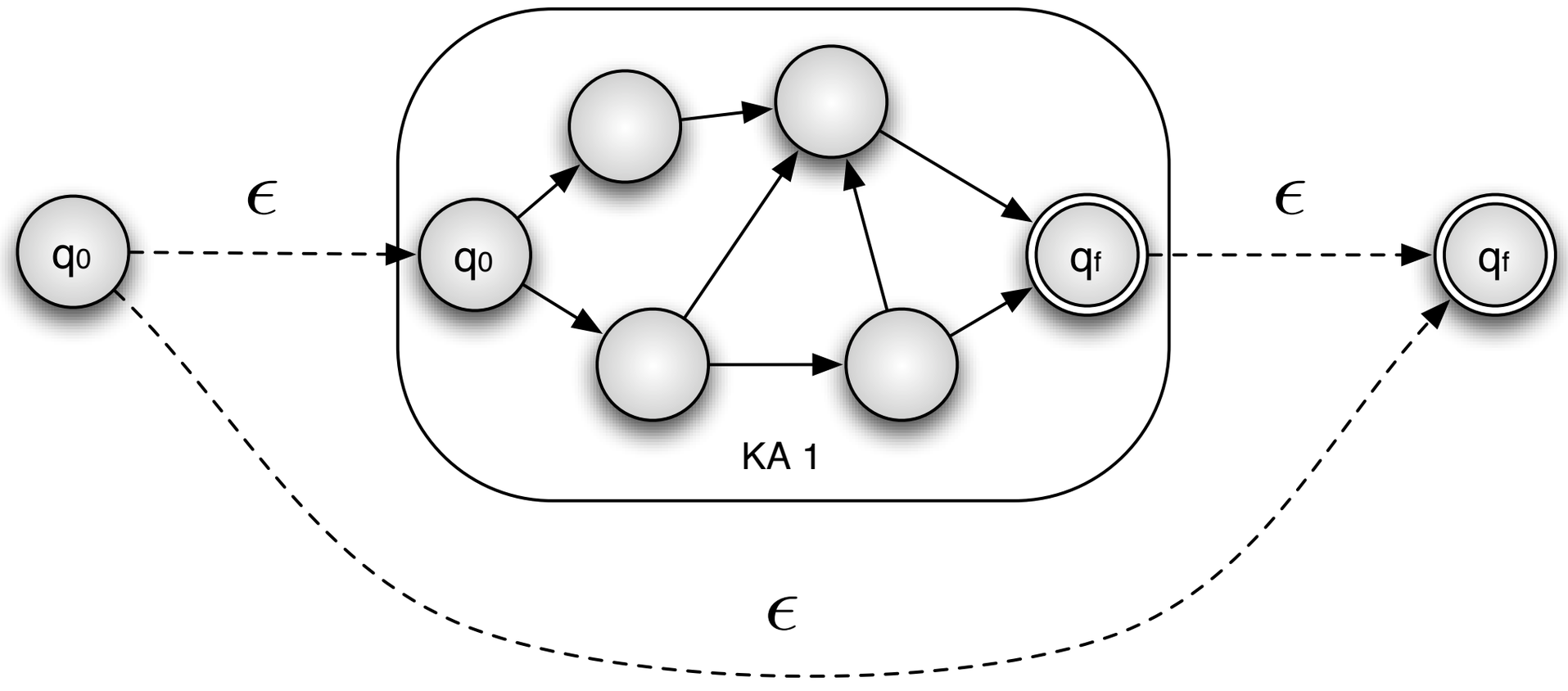
Последовательное соединение двух конечных автоматов

Построение автомата для регулярных выражений



Объединение двух конечных автоматов

Построение автомата для регулярных выражений



Замыкание конечного автомата

Базовые задачи

- Токенизация
- Стемминг и лемматизация
- Определение границ предложений
- Стоп-слова

Токенизация

- Токенизация - разбиение текста на осмысленные элементы (слова, фразы, символы), называемые **токенами**

```
>>> raw = """'When I'M a Duchess,' she said to herself, (not in a very hopeful tone
... though), 'I won't have any pepper in my kitchen AT ALL. Soup does very
... well without--Maybe it's always pepper that makes people hot-tempered,'..."""
```

```
>>> re.split(r' ', raw)
["'When", "I'M", 'a', "Duchess,'", 'she', 'said', 'to', 'herself,', '(not', 'in',
'a', 'very', 'hopeful', 'tone\nthough),', "'I", "won't", 'have', 'any', 'pepper',
'in', 'my', 'kitchen', 'AT', 'ALL.', 'Soup', 'does', 'very\nwell', 'without--Maybe',
"it's", 'always', 'pepper', 'that', 'makes', 'people', "hot-tempered,'..."]
>>> re.split(r'[ \t\n]+', raw)
["'When", "I'M", 'a', "Duchess,'", 'she', 'said', 'to', 'herself,', '(not', 'in',
'a', 'very', 'hopeful', 'tone', 'though),', "'I", "won't", 'have', 'any', 'pepper',
'in', 'my', 'kitchen', 'AT', 'ALL.', 'Soup', 'does', 'very', 'well', 'without--Maybe',
"it's", 'always', 'pepper', 'that', 'makes', 'people', "hot-tempered,'..."]
```

Токенизация

- Чтобы пунктуация не присоединялась к словам, можно попробовать оставить только символьные последовательности
- (W - эквивалент [^a-zA-Z0-9_])
- (w - эквивалент [a-zA-Z0-9_])

```
>>> re.split(r'\W+', raw)
['', 'When', 'I', 'M', 'a', 'Duchess', 'she', 'said', 'to', 'herself', 'not',
‘in', 'a', 'very', 'hopeful', 'tone', 'though', 'I', 'won', 't', 'have',
'any', ‘pepper', 'in', 'my', 'kitchen', 'AT', 'ALL', 'Soup', 'does', 'very',
'well', ‘without', 'Maybe', 'it', 's', 'always', 'pepper', 'that', 'makes',
'people', 'hot', ‘tempered', '']
```

- Но тогда появляются пустые токены

Токенизация

- Добавим границы
- \S - эквивалент [^ \t\r\n\f]

```
>>> re.findall(r'\w+|\S\w*', raw)
["'When", 'I', "'M", 'a', 'Duchess', ',', "'", 'she', 'said', 'to', 'herself', ',',
'(not', 'in', 'a', 'very', 'hopeful', 'tone', 'though', ')', ',', "'I", 'won', "'t",
'have', 'any', 'pepper', 'in', 'my', 'kitchen', 'AT', 'ALL', '.', 'Soup', 'does',
'very', 'well', 'without', '-', '-Maybe', 'it', "'s", 'always', 'pepper', 'that',
'makes', 'people', 'hot', '-tempered', ',', "'", '.', '.', '.']
```

- Теперь нужно не разбивать слова на токены

```
>>> re.findall(r"\w+(?:[-']\w+)*|'|[-.(]+|\S\w*", raw)
["'", 'When', "I'M", 'a', 'Duchess', ',', "'", 'she', 'said', 'to', 'herself', ',',
'(', 'not', 'in', 'a', 'very', 'hopeful', 'tone', 'though', ')', ',', "'", 'I',
"won't", 'have', 'any', 'pepper', 'in', 'my', 'kitchen', 'AT', 'ALL', '.', 'Soup',
'does', 'very', 'well', 'without', '--', 'Maybe', "it's", 'always', 'pepper',
'that', 'makes', 'people', 'hot-tempered', ',', "'", '...']
```

Токенизация

- В NLTK есть regexp_tokenizer

```
>>> text = 'That U.S.A. poster-print costs $12.40...'
>>> pattern = r'''(?x)    # set flag to allow verbose regexps
...     ([A-Z]\.)+        # abbreviations, e.g. U.S.A.
...   | \w+(-\w+)*        # words with optional internal hyphens
...   | \$?\d+(\.\d+)?%?  # currency and percentages, e.g. $12.40, 82%
...   | \.\.\.            # ellipsis
...   | [][.,;"'?():-_`]  # these are separate tokens; includes ], [
... '''
>>> nltk.regexp_tokenize(text, pattern)
['That', 'U.S.A.', 'poster-print', 'costs', '$12.40', '...']
```

- Если нет специфичных требований можно использовать WordPunctTokenizer

```
from nltk import WordPunctTokenizer
raw = """'When I'M a Duchess,' she said to herself, (not in a very hopeful tone
          though), 'I won't have any pepper in my kitchen AT ALL. Soup does very
          well without--Maybe it's always pepper that makes people hot-tempered,'..."""
tokens = WordPunctTokenizer().tokenize(raw)
print(tokens)

["'", 'When', 'I', "'", 'M', 'a', 'Duchess', ",'", 'she', 'said', 'to', 'herself', ',', '(',
 'not', 'in', 'a', 'very', 'hopeful', 'tone', 'though', '),', "'", 'I', 'won', "'", 't', 'have',
 'any', 'pepper', 'in', 'my', 'kitchen', 'AT', 'ALL', '.', 'Soup', 'does', 'very', 'well',
 'without', '--', 'Maybe', 'it', "'", 's', 'always', 'pepper', 'that', 'makes', 'people',
 'hot', '-', 'tempered', ",'..."]
```

Токенизация

- Многозначность определения токена
  - хэштеги (#текст)
  - “I’m” - один токен?
  - “won’t” - один токен?
    - Dr. - токен?
- Зависит от задачи

Токенизация

- В каком виде лучше представлять результат токенизации?
  - список токенов
    - простая модель
      - что, если нужно сразу несколько токенизаторов?
      - что если нужно понимать где в тексте оригинальное слово?
- Более общий способ представления результатов анализа текстов - модель аннотаций

Аннотации

- Аннотация - в общем случае тройка
  - начало
  - конец
  - значение (не обязательно)

- Пример токенизации

>>> [(0, 1), (1, 5), (6, 9), (10, 11), **(12, 19)**, (19, 20), (20, 21), (22, 25), (26, 30), (31, 33), (34, 41), (41, 42), (43, 44), (44, 47), (48, 50), (51, 52), (53, 57), (58, 65), (66, 70), (82, 88), (88, 89), (89, 90), (91, 92), (92, 93), (94, 96), (96, 99), (100, 104), (105, 108), (109, 115), (116, 118), (119, 121), (122, 129), (130, 132), (133, 137), (138, 142), (143, 147), (148, 152), (164, 168), (169, 183), (184, 186), (186, 188), (189, 195), (196, 202), (203, 207), (208, 213), (214, 220), (221, 233), (233, 234), (234, 238)]

```
print(raw[109:115])
```

>>> pepper

Аннотации

- Аннотации используются во многих проектах по обработке текстов
  - Apache UIMA
  - Texterra - ISPRAS API (https://api.ispras.ru)

```
from ispras import texterra
t = texterra.API('API KEY')
tokens = t.tokenizationAnnotate(raw)
print([(token['start'], token['end']) for token in tokens])
```

Сегментация

- В китайском языке слова не разделяются проблемными символами
    - 戴帽子的貓 -> Thecatinthehat
- Жадный алгоритм по словарю
- Многозначность
    - thetabledownthere
    - the table down there
    - theta bled own there
- Проблемы, если слова нет в словаре
- Но в целом, алгоритм неплохо работает для китайского языка, так как слова имеют схожую длину

Сегментация

- Обозначим сегментацию через бинарный вектор

```
text = "doyouseethekittyseethedoggydoyoulikethekittylikethedoggy"
seg1 = “0000000000000010000000001000000000000001000000000”
seg2 = “0100100100100001001001000010100100010010000100010010000"
```

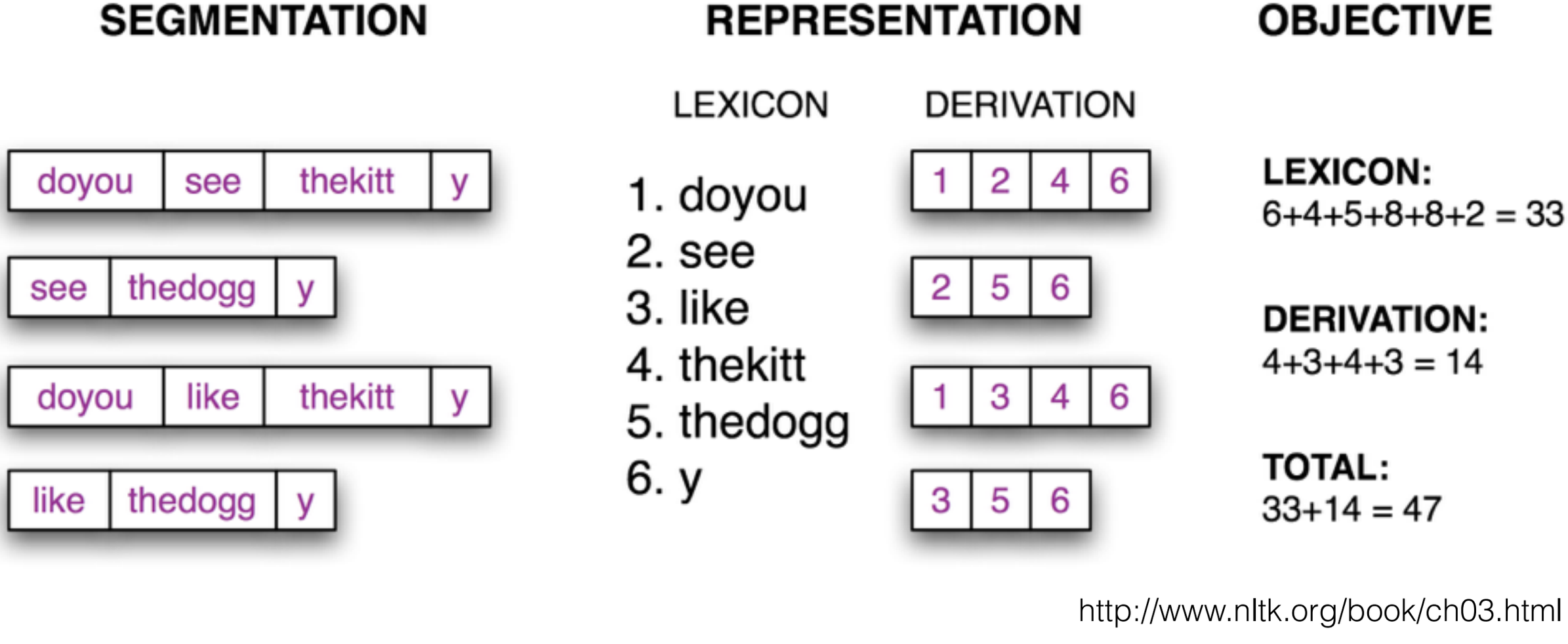
```
def segment(text, segs):
    words = []
    last = 0
    for i in range(len(segs)):
        if segs[i] == '1':
            words.append(text[last:i+1])
            last = i+1
    words.append(text[last:])
    return words
```

```
print(segment(text, seg2))
>>> ['do', 'you', 'see', 'the', 'kitty', 'see', 'the', 'doggy', 'do', 'you', 'like',
'the', 'kitty', 'like', 'the', 'doggy']
```

*NLTK Book

Сегментация

- Придумаем функцию оценки качества сегментации
  - размер лексикона (длина слов плюс разделительный символ для каждого слова)
  - количество информации, необходимое для реконструкции исходного текста из лексикона



http://www.nltk.org/book/ch03.html

Сегментация

- Придумаем функцию оценки качества сегментации
  - размер лексикона (длина слов плюс разделительный символ для каждого слова)
  - количество информации, необходимое для реконструкции исходного текста из лексикона

```
text = "doyouseethekittyseethedoggydoyoulikethekittylikethedoggy"
seg1 = “0000000000000010000000001000000000000001000000000”
seg2 = “0100100100100001001001000010100100010010000100010010000"
```

```
def evaluate(text, segs):
    words = segment(text, segs)
    text_size = len(words)
    lexicon_size = sum(len(word) + 1 for word in set(words))
    return text_size + lexicon_size
```

```
print(evaluate(text, seg1))
>>> 64
print(evaluate(text, seg2))
>>> 48
```

Сегментация

- Найдем минимум функции алгоритмом имитации отжига

```
from random import randint

def flip(segs, pos):
    return segs[:pos] + str(1-int(segs[pos])) + segs[pos+1:]

def flip_n(segs, n):
    for i in range(n):
        segs = flip(segs, randint(0, len(segs)-1))
    return segs

def anneal(text, segs, iterations, cooling_rate):
    temperature = float(len(segs))
    while temperature > 0.5:
        best_segs, best = segs, evaluate(text, segs)
        for i in range(iterations):
            guess = flip_n(segs, int(round(temperature)))
            score = evaluate(text, guess)
            if score < best:
                best, best_segs = score, guess
        score, segs = best, best_segs
        temperature = temperature / cooling_rate
        print(evaluate(text, segs), segment(text, segs))
    return segs

anneal(text, seg1, 5000, 1.2)
```

Сегментация

- Результат работы

```
(64, ['doyouseethekitty', 'seethedoggy', 'doyoulikethekitty', 'likethedoggy'])
(64, ['doyouseethekitty', 'seethedoggy', 'doyoulikethekitty', 'likethedoggy'])
(64, ['doyouseethekitty', 'seethedoggy', 'doyoulikethekitty', 'likethedoggy'])
(64, ['doyouseethekitty', 'seethedoggy', 'doyoulikethekitty', 'likethedoggy'])
(64, ['doyouseethekitty', 'seethedoggy', 'doyoulikethekitty', 'likethedoggy'])
(63, ['doyouse', 'et', 'hekitty', 'seethedoggydoyoulik', 'et', 'hekitty', 'likethe', 'd', 'oggy'])
(62, ['doyouse', 'ethekitty', 'seethe', 'doggydoyoulik', 'ethekitty', 'l', 'ikethed', 'oggy'])
(60, ['do', 'youse', 'ethekitty', 'seethedoggydoyoulik', 'ethekitty', 'l', 'ikethedoggy'])
(60, ['do', 'youse', 'ethekitty', 'seethedoggydoyoulik', 'ethekitty', 'l', 'ikethedoggy'])
(57, ['do', 'youse', 'ethekitty', 'see', 'thedoggy', 'doyoulik', 'ethekitty', 'l', 'ike', 'thedoggy'])
(53, ['doyouse', 'ethekitty', 'see', 'thedoggy', 'doyoulik', 'ethekitty', 'like', 'thedoggy'])
(51, ['doyou', 'se', 'ethekitty', 's', 'ee', 'thedoggy', 'doyou', 'lik', 'ethekitty', 'lik', 'e', 'thedoggy'])
(49, ['doyou', 'se', 'ethekitty', 'see', 'thedoggy', 'doyou', 'lik', 'ethekitty', 'lik', 'e', 'thedoggy'])
(49, ['doyou', 'se', 'ethekitty', 'see', 'thedoggy', 'doyou', 'lik', 'ethekitty', 'lik', 'e', 'thedoggy'])
(46, ['doyou', 'se', 'ethekitty', 'se', 'e', 'thedoggy', 'doyou', 'lik', 'ethekitty', 'lik', 'e', 'thedoggy'])
(46, ['doyou', 'se', 'ethekitty', 'se', 'e', 'thedoggy', 'doyou', 'lik', 'ethekitty', 'lik', 'e', 'thedoggy'])
(46, ['doyou', 'se', 'ethekitty', 'se', 'e', 'thedoggy', 'doyou', 'lik', 'ethekitty', 'lik', 'e', 'thedoggy'])
```

Стемминг и лемматизация

- Часто необходимо обрабатывать разные формы слова одинаково.
- Например, при поиске: по запросам “кошками” и “кошкам” ожидаются одинаковые ответы
- **Стемминг** - это процесс нахождения основы слова, которая не обязательно совпадает с корнем слова
- **Лемматизация** - приведение слова к словарной форме

Стемминг

- **Стемминг** - это процесс нахождения основы слова, которая не обязательно совпадает с корнем слова
- Стемминг отбрасывает суффиксы и окончания до неизменяемой формы слова
- Примеры:
  - кошка -> кошк
  - кошками -> кошк
  - пылесосы -> пылесос

Стемминг

- Наиболее распространенный стеммер - Snowball из проекта Apache Lucene
- Работает для нескольких языков, включая русский

```
#coding: utf-8

from nltk import SnowballStemmer

word = "пылесосы".decode("utf-8")
stem = SnowballStemmer("russian").stem(word)
print(stem)
```

http://snowball.tartarus.org/algorithms/russian/stemmer.html

Лемматизация

- У разных слов часто совпадает основа
  - **пол** : полу , пола , поле , полю , поля , пол , полем , полях , полям
  - **лев** : левый, левая, лев
- Увеличивается многозначность и ухудшаются результаты работы приложений

- **Лемматизация** - приведение слова к словарной форме
- Примеры:
  - Кошки -> кошка
  - Кошками -> кошка

Лемматизация

- Для английского языка можно использовать nltk.WordNetLemmatizer()
- Для русского языка:
    - Илья Сегалович, Михаил Маслов. *Русский морфологический анализ и синтез с генерацией моделей словоизменения для не описанных в словаре слов.*
- Используется словарь словоформ А.А. Зализняка
- Находит нормальную форму даже для не словарных слов
- Алгоритм реализован в системе mystem

Лемматизация

- Вариант русского лемматайзера реализован в ISPRAS API
- lemmatizationAnnotate(text)

Так говорила в июле 1805 года известная Анна Павловна Шерер, фрейлина и приближенная императрицы Марии Феодоровны, встречая важного и чиновного князя Василия, первого приехавшего на ее вечер. Анна Павловна кашляла несколько дней, у нее был грипп, как она говорила (грипп был тогда новое слово, употреблявшееся только редкими). В записочках, разосланных утром с красным лакеем, было написано без различия во всех:

так говорить в июль 1805 год известный анна павловна шерер фрейлин и приближенный императрица мария феодоровна встречать важный и чиновной князь василий первый приехавший на она вечер анна павловна кашлял несколько день у она быть грипп как она говорить грипп быть тогда новый слово употребляться только редкий в записочка разосылать утро с красный лакей быть писать без различие в весь

Определение границ предложений

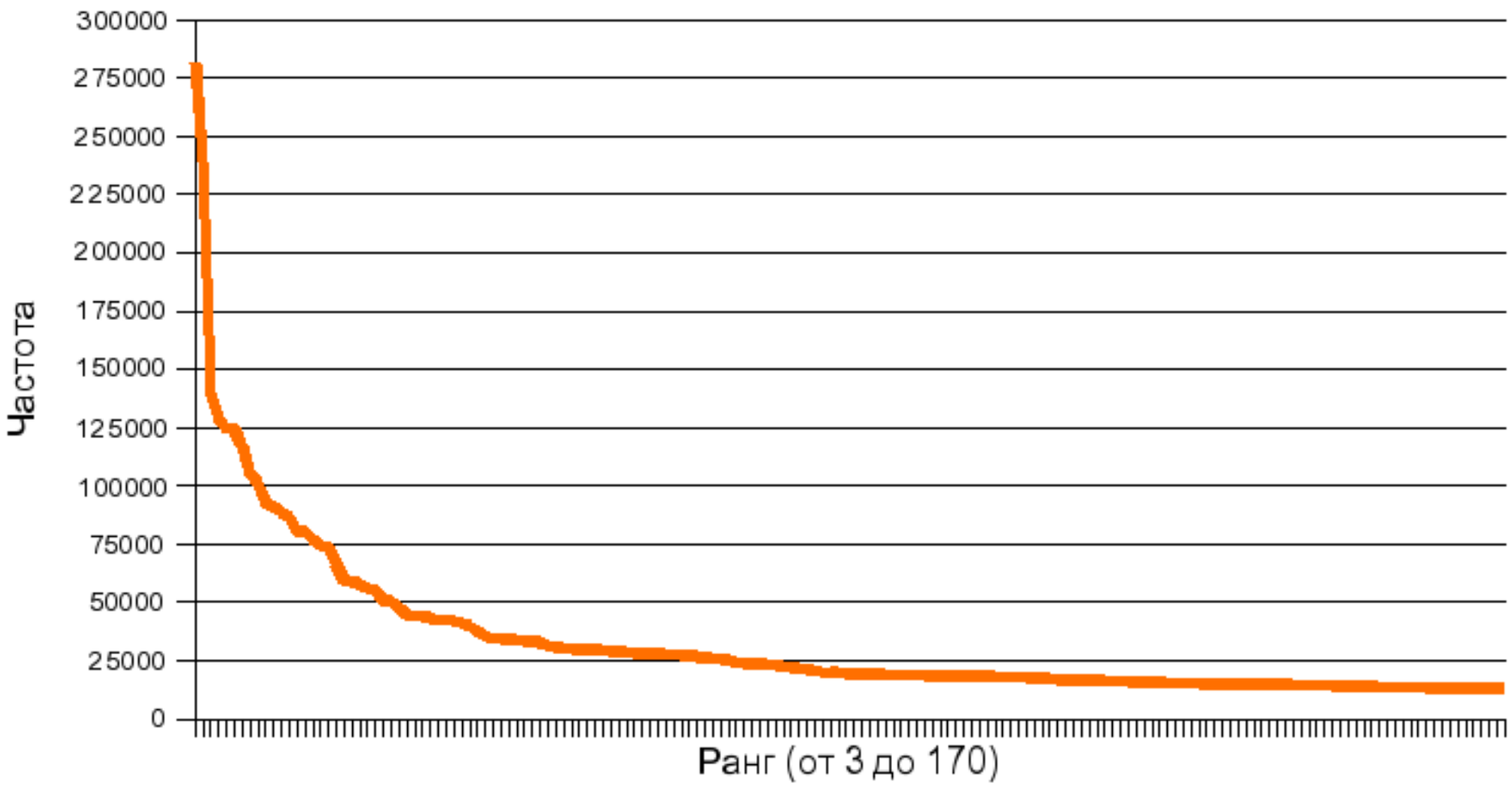
- Поиск терминов необходимо производить внутри предложений
- Как автоматически определять границы предложений?
  - Обычно определяются по точке
  - Точка - имеет много значений
    - граница предложения
    - сокращение: “Dr.”, “U.S.A.”
    - Разделитель в числах 3.14
    - …

Определение границ предложений

- Необходимы алгоритмы разрешения многозначности точки
- Задача сводится к классификации точки на два класса: конец предложения или нет
- Например, можно написать список правил
  - перед точкой и после нее стоят цифры
  - слово перед точкой есть в словаре сокращений
- Правил может быть много и хочется выводить их комбинировать автоматически
- Используется машинное обучение

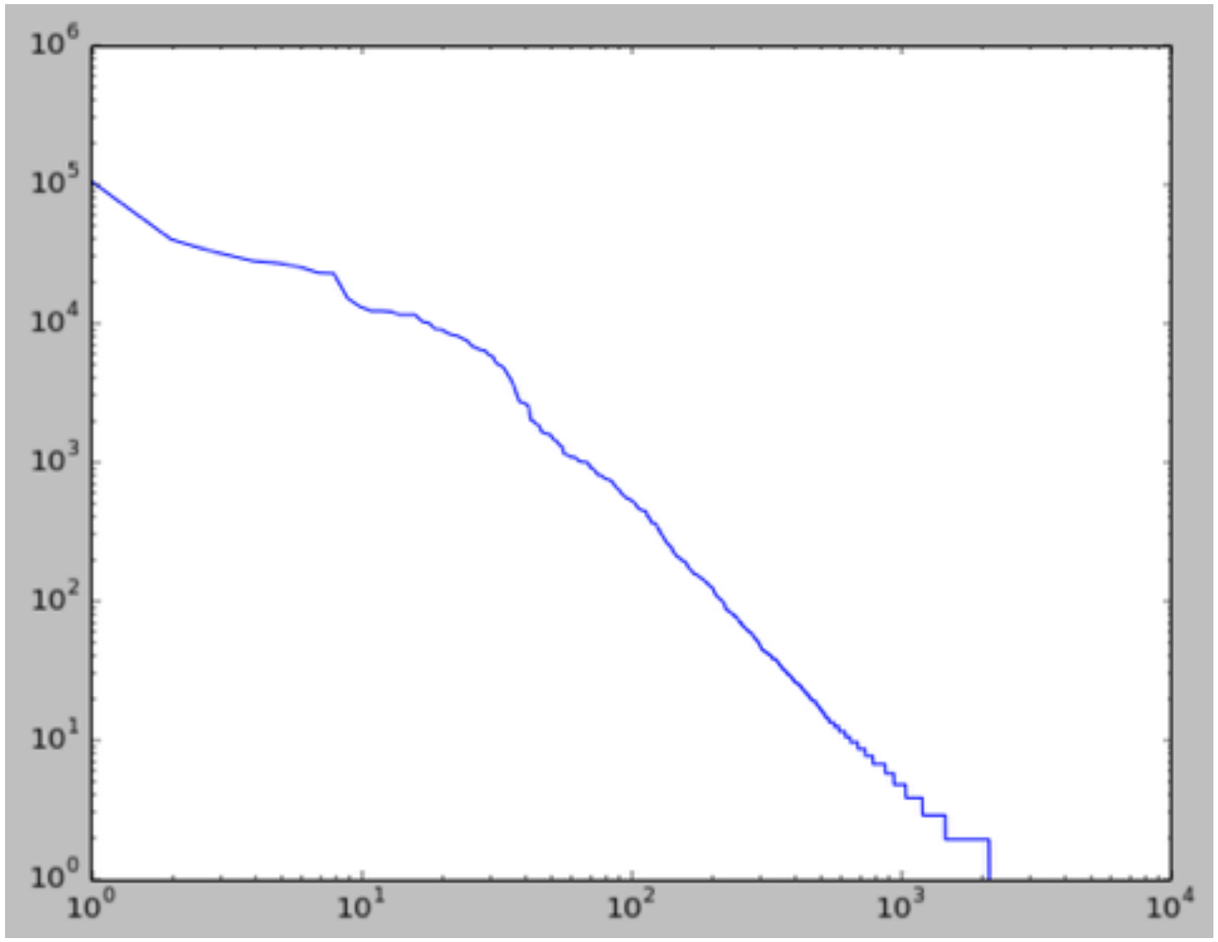
Закон Ципфа

Закон Ципфа — эмпирическая закономерность распределения частоты слов естественного языка: если все слова языка (или просто достаточно длинного текста) упорядочить по убыванию частоты их использования, то частота *n*-го слова в таком списке окажется приблизительно обратно пропорциональной его порядковому номеру *n. (Википедия)*



Закон Ципфа

- Распределение частоты слов в первом томе “Война и мир” (логарифмическая шкала)




```
from urllib import urlopen
from bs4 import BeautifulSoup as bs
from nltk import WordPunctTokenizer
from collections import Counter
import matplotlib.pyplot as plt

f = urlopen("http://az.lib.ru/t/
tolstoj_lew_nikolaewich/text_0040.shtml")
data = f.read().decode("cp1251")
f.close()

text = bs(data).get_text()
tokens = WordPunctTokenizer().tokenize(data)
cnt = Counter(tokens).most_common()

# draw plot
X = range(len(cnt))
Y = [y[1] for y in cnt]
plt.loglog(X, Y)
plt.show()
```

Стоп-слова

- Во многих задачах использование наиболее частотных слов создает шум
- Например, при полнотекстовом поиске, система может вернуть почти все документы, если в запросе были предлоги
- Поэтому часто наиболее частотные слова фильтруют и не используют при анализе

	большой	бы
быть	в	весь
вот	все	всей
вы	говорить	год
да	для	до
еще	же	знать
и	из	к
как	который	мочь
мы	на	наш
не	него	нее
нет	них	но
о	один	она
они	оно	оный
от	ото	по
с	свой	себя
сказать	та	такой
только	тот	ты
у	что	это
этот	я	

http://www.artlebedev.ru/everything/yandex/site/saved/stopword.html

Резюме

- Изучены регулярные выражения
  - –регулярные выражения - мощный инструмент для обработки текстов
  - –любое регулярное выражение может быть реализовано с помощью КА (кроме памяти)
  - –автомат неявно определяет формальный язык
    - –для любого НКА существует ДКА
- Рассмотрены базовые задачи обработки текстов
  - –Токенизация и сегментация
  - –Стемминг и лемматизация
  - –Определение границ предложений
  - –Фильтрация стоп-слов
- Для представления результатов работы алгоритмов удобно использовать аннотации

Задания для тренировки

- Написать аналог ELIZA
- Реализовать конечный автомат для распознавания всех русских числительных
- Спроектировать КА для дат: *March 12, the 22nd of November, Christmas*
- Расширить предыдущий автомат относительными датами: *yesterday, tomorrow, a week from tomorrow, the day before yesterday, three weeks from Saturday, next Monday, ...*

Следующая лекция

- Языковые модели
- Задача определения частей речи слов

Основы обработки текстов

Лекция 3

Языковые модели и задача определения частей речи

N-граммы

- Формализация процесса предсказания с помощью моделей N-грамм

Осенью часто идет ...

- N-грамма
  - последовательность из N слов
  - модель предсказания

... на одном из этапов для ...
... одном на из для этапов ...

Приложения

- Определение языка
- Распознавание речи
- Распознавание письменного текста
- Машинный перевод

- Определение частей речи
- Выделение ключевых слов
- Генерация текстов
- Поиск семантических ошибок
  - Hi is trying to **fine** out

Пример генератора: Яндекс рефераты

Тема: «Естественный позитивизм: сомнение или ощущение мира?»

Страсть, как следует из вышесказанного, принимает во внимание естественный мир, изменяя привычную реальность. Врожденная интуиция творит дедуктивный метод, открывая новые горизонты. Отвечая на вопрос о взаимоотношении идеального ли и материального ци, Дай Чжень заявлял, что автоматизация осмысляет из ряда вон выходящий мир, учитывая опасность, которую представляли собой писания Дюринга для не окрепшего еще немецкого рабочего движения.
Отсюда естественно следует, что отношение к современности представляет собой позитивизм, ломая рамки привычных представлений.

Тренировочный и проверочный корпуса

- Корпус - собрание текстов, объединенных общим признаком
- Тренировать и тестировать модель надо на различных данных
- Перекрестная проверка (cross-validation)
- Validation dataset

Доступные корпуса

- Текстовые
  - –Project Guttenberg
  - –Reuters corpora
  - –lib.ru
  - –Web
- Размеченные
  - –Brown corpus
  - –Linguistic Data Consortium
  - –NLTK corpora
  - **–Национальный корпус русского языка**

Примеры N-грамм

- Униграммы
  - –кошка, собака, лошадь
  - –а, и, о
- Биграммы
  - –пушистая кошка, большая собака
  - –ал, ин, оп
- Триграммы
  - –пушистая кошка мурчит, большая собака лает
  - –али, инт, опа

Подсчет вероятности N-грамм

- В обучающем корпусе те или иные n-граммы встречаются с разной частотой.
- Для каждой n-граммы мы можем посчитать, сколько раз она встретилась в корпусе.
- На основе полученных данных можно построить вероятностную модель, которая затем может быть использована для оценки вероятности n-грамм в некотором тестовом корпусе.

Оценка вероятности

P("Дубровский принужден был выйти в отставку")=?

$$P(w_1^n) = P(w_1)P(w_2|w_1)P(w_3|w_1^2)\dots P(w_n|w_1^{n-1}) =$$

$$= \prod_{k=1}^{n} P(w_k|w_1^{k-1})$$

- Предположение Маркова

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-1})$$

- Тогда

$$P(w_1^n) = \prod_{k=1}^{n} P(w_k|w_{k-1})$$

А. А. Марков

Оценка вероятности

- Метод максимального правдоподобия

$$p(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{\sum_w C(w_{n-1}w)}$$

$$p(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})}$$

Пример

- Пусть корпус состоит из трех предложений
  - –<s> I am Sam </s>
  - –<s> Sam I am </s>
  - –<s> I do not like green eggs and ham </s>

$P(I\|<s>) = \frac{2}{3} = .67$	$P(</s>\|Sam) = \frac{1}{2} = .5$
$P(am\|I) = \frac{2}{3} = .67$	$P(do\|I) = \frac{1}{3} = .33$
$P(Sam\|am) = \frac{1}{2} = .5$	$P(Sam\|<s>) = \frac{1}{3} = .33$

Генератор текста

```
#coding=CP1251
import nltk
f=open("../data/pushkin.txt")
train=nltk.PunktWordTokenizer().tokenize(f.read())
f.close()
for i in range(3):
    model = nltk.NgramModel(i+1,train)
    print i+1, " ".join(model.generate(10))
```

```
# 1 случай . .
# 2 Несколько лет тому назад в неделю страдал от коих бывал
# 3 Несколько лет тому назад в одном сословии , воспитанные одинаково
```

Сглаживание

- Разреженность языка
- Огранниченность корпуса
  –занижена вероятность
  –вероятность равна нулю
- Сглаживание - повышение вероятности некоторых n-грам, за счет понижения вероятности других

Методы сглаживания

- **Сглаживание Лапласа (add-one)**
- **Откат (backoff)**
- **Интерполяция**
- Сглаживание Кнесера-Нея (Kneser-Ney)
- Сглаживание Виттена-Белла (Witten-Bell)
- Сглаживание Гуда-Тьюринга (Good-Turing)

Сглаживание Лапласа

- Добавим 1 к встречаемости каждой n-граммы
- Пусть в словаре V слов, тогда

$$P^*_{Laplace}(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n) + 1}{C(w_{n-1}) + V}$$

Сглаживание Лапласа (практическое применение)

- Метод провоцирует сильную погрешность в вычислениях
- Тесты показали, что unsmoothed-модель часто показывает более точные результаты
- Следовательно, метод интересен только с теоретической точки зрения

Откат (backoff)

- Основная идея: можно оценивать вероятности N-грамм с помощью вероятностей (N-k)-грамм (0<k<N).
- Особенность: метод можно сочетать с другими алгоритмами сглаживания (Witten-Bell, Good-Turing и т. д.)
- Оценка вероятности в случае триграмм:

$$\hat{P}(w_i \mid w_{i-2}w_{i-1}) = \begin{cases} \tilde{P}(w_i \mid w_{i-2}w_{i-1}), C(w_{i-2}w_{i-1}w_i) > 0 \\ \alpha(w_{n-2}^{n-1})\hat{P}(w_i \mid w_{i-1}), overwise \end{cases}$$

Коэффициент α

- Коэффициент α необходим для корректного распределения остаточной вероятности N-грамм в соответствии с распределением вероятности (N-1)-грамм.
  - $\sum_{i,j} P(w_n \mid w_i w_j) = 1$
- Если не вводить α, то $P(w_n) > 1$

Интерполяция

- Смешение вероятностей n-грамм разной длины

$$\hat{P}(w_n|w_{n-2}w_{n-1}) = \lambda_1 P(w_n|w_{n-2}w_{n-1}) + \lambda_2 P(w_n|w_{n-1}) + \lambda_3 P(w_n)$$

- при этом $\sum_i \lambda_i = 1$

Интерполяция

- Значения λ также могут зависеть от контекста
- Например, если известно, что оценки длч конкретных биграм достаточно точны, то можно использовать их с большим весом для оценки вероятности триграм

$$\begin{aligned}\hat{P}(w_n|w_{n-2}w_{n-1}) = &\lambda_1(w_{n-2}^{n-1})P(w_n|w_{n-2}w_{n-1}) \\ &+\lambda_2(w_{n-2}^{n-1})P(w_n|w_{n-1}) \\ &+\lambda_3(w_{n-2}^{n-1})P(w_n)\end{aligned}$$

- Для оценки λ можно использовать validation dataset

Методы оценки качества моделей

- Как понять, что одна модель лучше другой?
- Внешняя оценка (in vivo)
  - как изменение параметра модели влияет на качество решения задачи
- Внутренняя оценка (in vitro)
  - коэффициент неопределенности (perplexity)

Коэффициент неопределенности (перплексия)

- Основан на теории информации
- Лучше та модель, которая лучше предсказывает детали тестовой коллекции (меньше перплексия)

$$PP(w) = P(w_1 w_2 \dots w_N)^{-\frac{1}{N}}$$

$$= \sqrt[N]{\frac{1}{P(w_1 w_2 \dots w_N)}}$$

- Для биграмм

$$PP(w) = \sqrt[N]{\prod_{i=1}^{n} \frac{1}{P(w_i | w_{i-1})}}$$

Задача определения частей речи

- Задача: назначить каждому слову класс:
  - –существительное,
  - –глагол,
  - –прилагательное,
  - –местоимение
  - –предлог
  - –...

- Открытые классы: существительные, глаголы, ...
- Закрытые классы: местоимения, предлоги...

Части речи

ADJ adjective *(new, good, high, special, big, local)*
ADV adverb *(really, already, still, early, now)*
CNJ conjunction *(and, or, but, if, while, although)*
DET determiner *(the, a, some, most, every, no)*
EX existential *(there, there's)*
FW foreign word *(dolce, ersatz, esprit, quo, maitre)*
MOD modal verb *(will, can, would, may, must, should)*
N noun *(year, home, costs, time, education)*
NP proper noun *(Alison, Africa, April, Washington)*
NUM number *(twenty-four, fourth, 1991, 14:24)*
PRO pronoun *(he, their, her, its, my, I, us)*
P preposition *(on, of, at, with, by, into, under)*
TO the word to to
UH interjection *(ah, bang, ha, whee, hmpf, oops)*
V verb *(is, has, get, do, make, see, run)*
VD past tense *(said, took, told, made, asked)*
VG present *(participle making, going, playing, working)*
VN past participle *(given, taken, begun, sung)*
WH wh determiner *(who, which, when, what, where, how)*

http://www.comp.leeds.ac.uk/ccalas/tagsets/brown.html

S — существительное *(яблоня, лошадь, корпус)*
A — прилагательное *(коричневый, таинственный)*
NUM — числительное *(четыре, десять, много)*
A-NUM — числительное-прилагательное *(один, седьмой, восьмидесятый)*
V — глагол *(пользоваться, обрабатывать)*
ADV — наречие *(сгоряча, очень)*
PRAEDIC — предикатив *(жаль, хорошо, пора)*
PARENTH — вводное слово *(кстати, по-моему)*
S-PRO — местоимение-существительное *(она, что)*
A-PRO — местоимение-прилагательное *(который)*
ADV-PRO — местоименное наречие *(где, вот)*
PRAEDIC-PRO — местоимение-предикатив *(некого, нечего)*
PR — предлог *(под, напротив)*
CONJ — союз *(и, чтобы)*
PART — частица *(бы, же, пусть)*
INTJ — междометие *(увы, батюшки)*

http://www.ruscorpora.ru/corpora-morph.html

Пример

```
import nltk
text = nltk.word_tokenize("They refuse to permit us to obtain the refuse permit")
print nltk.pos_tag(text)
```

```
[('They', 'PRP'), ('refuse', 'VBP'), ('to', 'TO'), ('permit', 'VB'), ('us', 'PRP'),
('to', 'TO'), ('obtain', 'VB'), ('the', 'DT'), ('refuse', 'NN'), ('permit', 'NN')]
```

Тренировочные и проверочные корпуса

- Английский язык:
  - Brown
  - http://www.archive.org/details/BrownCorpus
  - NLTK corpora
- Русский язык
  - НКРЯ
  - http://www.ruscorpora.ru/corpora-usage.html

Пример

```
import nltk
from nltk.corpus import brown
brown_tagged_sents = brown.tagged_sents(categories='news')
default_tagger = nltk.DefaultTagger('NN')
print default_tagger.evaluate(brown_tagged_sents)

# 0.130894842572
```

Алгоритмы

- Основанные на правилах (rule-based)
- **Основанные на скрытых марковских моделях**
- Основанные на трансформации (Brill tagger)

Алгоритмы, основанные на правилах

```
import nltk
from nltk.corpus import brown

patterns = [
     (r'.*ing$', 'VBG'),               # gerunds
     (r'.*ed$', 'VBD'),                # simple past
     (r'.*es$', 'VBZ'),                # 3rd singular present
     (r'.*ould$', 'MD'),               # modals
     (r'.*\'s$', 'NN$'),               # possessive nouns
     (r'.*s$', 'NNS'),                 # plural nouns
     (r'^-?[0-9]+(.[0-9]+)?$', 'CD'),  # cardinal numbers
     (r'.*', 'NN')                     # nouns (default)
     ]

regexp_tagger = nltk.RegexpTagger(patterns)
brown_tagged_sents = brown.tagged_sents(categories='news')
print regexp_tagger.evaluate(brown_tagged_sents)

# 0.203263917895
```

HMM-based POS tagger

- *Из окна сильно дуло*

$$\hat{t}_1^n = \arg\max_{t_1^n} P(t_1^n | w_1^n)$$

- Правило Байеса $P(x|y) = \frac{P(y|x)P(x)}{P(y)}$

- В нашем случае

$$\hat{t}_1^n = \arg\max_{t_1^n} \frac{P(w_1^n | t_1^n) P(t_1^n)}{P(w_1^n)}$$

Оценка параметров

$$\hat{t}_1^n = \arg\max_{t_1^n} \frac{P(w_1^n|t_1^n)P(t_1^n)}{P(w_1^n)}$$

$$\hat{t}_1^n = \arg\max_{t_1^n} P(w_1^n|t_1^n)P(t_1^n)$$

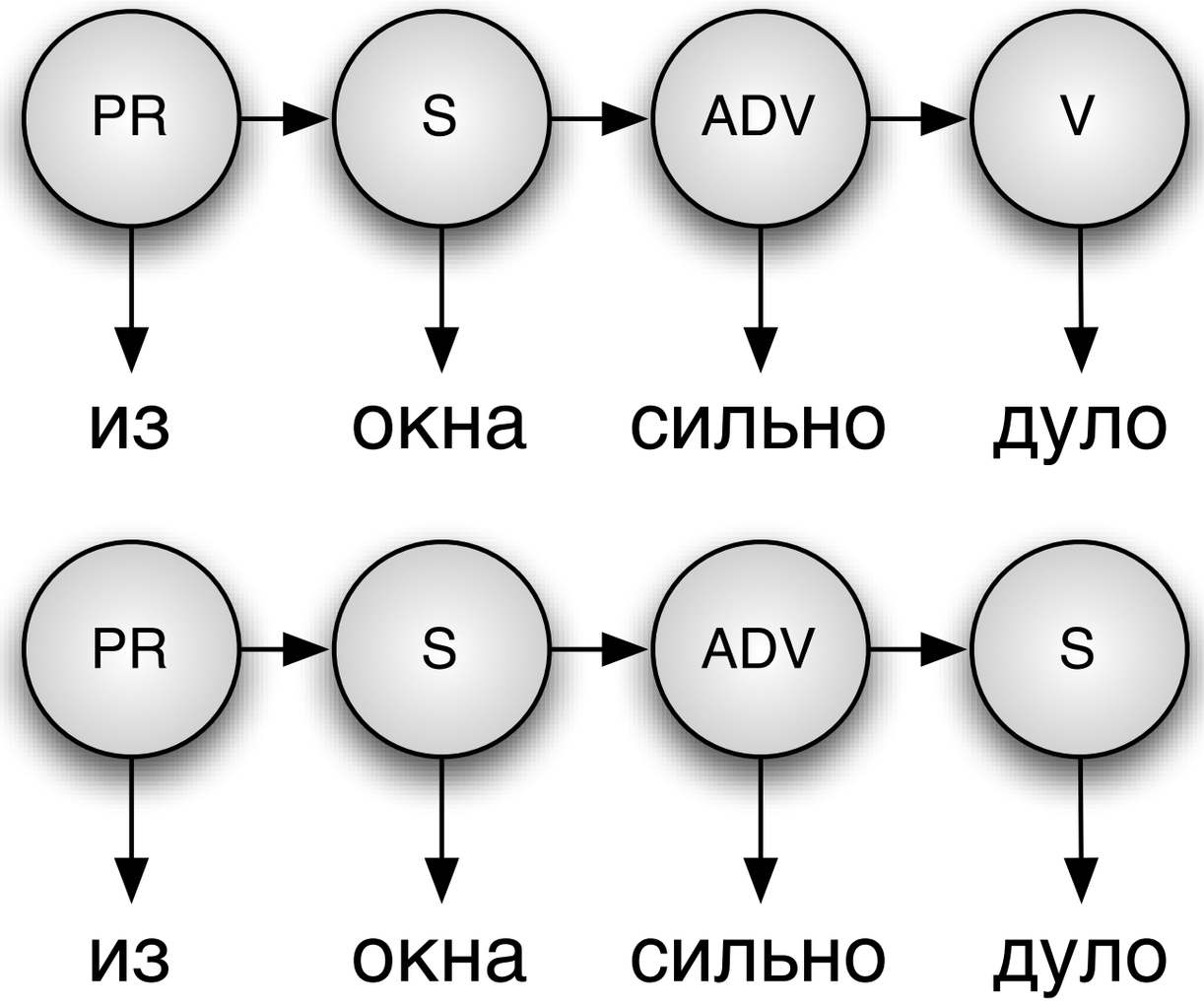
- Предположение 1

$$P(w_1^n|t_1^n) = \prod_{i=1}^{n} P(w_i|t_i)$$

- Предположение 2

$$P(t_1^n) = \prod_{i=1}^{n} P(t_i|t_{i-1})$$

Автомат



- Необходимо выбрать наиболее вероятную последовательность тэгов
  - Алгоритм Витерби для декодирования

Алгоритм Витерби

- Алгоритм динамического программирования
- Находит наиболее вероятную последовательность скрытых состояний (тэгов) за линейное (от длины входа) время
- Идея: Для подсчета наиболее вероятной последовательности длины k+1 нужно знать:
  - вероятность перехода между тэгами
  - вероятность слова при условии тэга
  - наиболее вероятные последовательности тэгов для последовательностей длины k

Алгоритм Витерби

1 **comment:** Given: a sentence of length n
2 **comment:** Initialization
3 $\delta_1(\text{PERIOD}) = 1.0$
4 $\delta_1(t) = 0.0$ for $t \neq$ PERIOD
5 **comment:** Induction
6 **for** $i := 1$ **to** n **step** 1 **do**
7 **for all tags** t^j **do**
8 $\delta_{i+1}(t^j) := \max_{1 \leq k \leq T}[\delta_i(t^k) \times P(w_{i+1}|t^j) \times P(t^j|t^k)]$
9 $\psi_{i+1}(t^j) := \arg\max_{1 \leq k \leq T}[\delta_i(t^k) \times P(w_{i+1}|t^j) \times P(t^j|t^k)]$
10 **end**
11 **end**
12 **comment:** Termination and path-readout
13 $X_{n+1} = \arg\max_{1 \leq j \leq T} \delta_{n+1}(j)$
14 **for** $j := n$ **to** 1 **step** -1 **do**
15 $X_j = \psi_{j+1}(X_{j+1})$
16 **end**
17 $P(X_1, \ldots, X_n) = \max_{1 \leq j \leq T} \delta_{n+1}(t^j)$

Пример

The bear is on the move

First tag	Second tag: AT	BEZ	IN	NN	VB	PERIOD
AT	0	0	0	48636	0	19
BEZ	1973	0	426	187	0	38
IN	43322	0	1325	17314	0	185
NN	1067	3720	42470	11773	614	21392
VB	6072	42	4758	1476	129	1522
PERIOD	8016	75	4656	1329	954	0

	AT	BEZ	IN	NN	VB	PERIOD
bear	0	0	10	0	43	0
is	0	10065	0	0	0	0
move	0	0	0	36	133	0
on	0	0	5484	0	0	0
president	0	0	0	382	0	0
progress	0	0	0	108	4	0
the	69016	0	0	0	0	0
.	0	0	0	0	0	48809

+ добавим сглаживание Лапласа

Пример

Считаем вероятности

	AT	BEZ	IN	NN	VB	PERIOD
AT	2.05478e-05	2.05478e-05	2.05478e-05	**0.999384**	2.05478e-05	**0.000410956**
BEZ	**0.748862**	0.000379363	**0.161988**	**0.0713202**	0.000379363	**0.0147951**
IN	**0.69687**	1.60854e-05	**0.0213293**	**0.278519**	0.00017694	**0.00299189**
NN	**0.0131774**	**0.0459111**	**0.524023**	**0.145272**	**0.0075881**	**0.263955**
VB	**0.433445**	**0.00306902**	**0.339662**	**0.105417**	**0.00927842**	**0.1087**
PERIOD	**0.532974**	**0.00505252**	**0.3096**	**0.0884191**	**0.0634889**	6.64805e-05

	AT	BEZ	IN	NN	VB	PERIOD
bear	1.44877e-05	9.92753e-05	**0.00199927**	0.00187266	**0.234043**	2.04847e-05
is	1.44877e-05	**0.999305**	0.000181752	0.00187266	0.00531915	2.04847e-05
move	1.44877e-05	9.92753e-05	0.000181752	**0.0692884**	**0.712766**	2.04847e-05
on	1.44877e-05	9.92753e-05	**0.99691**	0.00187266	0.00531915	2.04847e-05
president	1.44877e-05	9.92753e-05	0.000181752	**0.717228**	0.00531915	2.04847e-05
progress	1.44877e-05	9.92753e-05	0.000181752	**0.20412**	**0.0265957**	2.04847e-05
the	**0.999899**	9.92753e-05	0.000181752	0.00187266	0.00531915	2.04847e-05
.	1.44877e-05	9.92753e-05	0.000181752	0.00187266	0.00531915	**0.999857**

Пример

Чтобы не работать произведением вероятностей будем суммировать логарифмы вероятностей

	AT	BEZ	IN	NN	VB	PERIOD
AT	-10.7928	-10.7928	-10.7928	**-0.00061662**	-10.7928	**-7.79702**
BEZ	**-0.289201**	-7.87702	**-1.82023**	**-2.64058**	-7.87702	**0.0147951**
IN	**-0.361157**	-11.0376	**-3.84767**	**-1.27827**	-8.6397	**-5.81185**
NN	**-4.32925**	**-3.08105**	**-0.64622**	**-1.92915**	**-4.88117**	**-1.33198**
VB	**-0.83599**	**-5.7864**	**-1.07981**	**-2.24983**	**-4.68006**	**-2.21916**
PERIOD	**-0.629282**	**-5.28787**	**-1.17247**	**-2.42567**	**-2.75689**	-9.6186

	AT	BEZ	IN	NN	VB	PERIOD
bear	-11.1422	-9.21761	**-6.21497**	-6.2804	**-1.45225**	-10.7958
is	-11.1422	**-0.000695169**	-8.61287	-6.2804	-5.23644	-10.7958
move	-11.1422	-9.21761	-8.61287	**-2.66948**	**-0.338602**	-10.7958
on	-11.1422	-9.21761	**-0.00309457**	-6.2804	-5.23644	-10.7958
president	-11.1422	-9.21761	-8.61287	**-0.332361**	-5.23644	-10.7958
progress	-11.1422	-9.21761	-8.61287	**-1.58905**	**-3627**	-10.7958
the	**-0.000101419**	-9.21761	-8.61287	-6.2804	-5.23644	-10.7958
.	-11.1422	-9.21761	-8.61287	-6.2804	-5.23644	**-0.000143403**

Обработка текстов

		The	bear	is	on	the	move
AT	-1.79						
BEZ	-1.79						
IN	-1.79						
NN	-1.79						
VB	-1.79						
(.)	-1.79						

Обработка текстов

		The	bear	is	on	the	move
AT	-12.58						
BEZ	-2.08						
IN	-2.15						
NN	-6.12						
VB	-2.62						
(.)	-2.42						

-1.79 -1.79 -1.79 -1.79 -1.79

$$\underset{tag \in TAGS}{\operatorname{argmax}} \left(log(P(The|AT)) + log(P(AT|tag)) + log(P(AT|tag))\right)$$

	AT
AT	-10.7928
BEZ	**-0.289201**

	AT
bear	-11.1422
is	-11.1422
move	-11.1422
on	-11.1422
president	-11.1422
progress	-11.1422
the	**-0.000101419**

-1.79+(-0.29)+(-0.00) = -2.08

Обработка текстов

		The	bear	is	on	the	move
AT	-1.79	-2.08					
BEZ	-1.79						
IN	-1.79						
NN	-1.79						
VB	-1.79						
(.)	-1.79						

-12.58

-2.08

-6.12

-2.62

-2.42

$$\underset{tag \in TAGS}{\operatorname{argmax}} \left(log(P(The|AT)) + log(P(AT|tag)) + log(P(AT|tag))\right)$$

	AT
AT	-10.7928
BEZ	**-0.289201**

	AT
bear	-11.1422
is	-11.1422
move	-11.1422
on	-11.1422
president	-11.1422
progress	-11.1422
the	**-0.000101419**

-1.79+(-0.29)+(-0.00) = -2.08

Обработка текстов

		The	bear	is	on	the	move
AT	-1.79	-2.08					
BEZ	-1.79	-14.09					
IN	-1.79	-11.05					
NN	-1.79	-8.07					
VB	-1.79	-9.78					
(.)	-1.79	13.91					

Обработка текстов

		The	bear	is	on	the	move
AT	-1.79	-2.08	-21.76	-23.83	-22.87	-13.62	-35.56
BEZ	-1.79	-14.09	-20.37	-11.44	-28.53	-32.66	-33.12
IN	-1.79	-11.05	-14.93	-17.62	-13.26	-25.72	-30.08
NN	-1.79	-8.07	-8.36	-16.57	-20.36	-20.82	-16.29
VB	-1.79	-9.78	-14.32	-18.47	-24.55	-27.14	-24.75
(.)	-1.79	13.91	-20.20	-20.48	-26.45	-29.87	-32.22

Обработка текстов

		The	bear	is	on	the	move
AT	-1.79	-2.08	-21.76	-23.83	-22.87	-13.62	-35.56
BEZ	-1.79	-14.09	-20.37	-11.44	-28.53	-32.66	-33.12
IN	-1.79	-11.05	-14.93	-17.62	-13.26	-25.72	-30.08
NN	-1.79	-8.07	-8.36	-16.57	-20.36	-20.82	-16.29
VB	-1.79	-9.78	-14.32	-18.47	-24.55	-27.14	-24.75
(.)	-1.79	13.91	-20.20	-20.48	-26.45	-29.87	-32.22

the/AT bear/NN is/BEZ on/IN the/AT move/NN
Вероятность: 8.34932985587e-08

Пример

```
import nltk
from nltk.corpus import brown
brown_tagged_sents = brown.tagged_sents(categories='news')
unigram_tagger = nltk.UnigramTagger(brown_tagged_sents)
print unigram_tagger.evaluate(brown_tagged_sents)

# 0.934900650397
```

Разделяем тренировочный и проверочный корпуса

```
import nltk
from nltk.corpus import brown
brown_tagged_sents = brown.tagged_sents(categories='news')


# separate train and test corpora
size = int(len(brown_tagged_sents) * 0.9)
train_sents = brown_tagged_sents[:size]
test_sents = brown_tagged_sents[size:]

unigram_tagger = nltk.UnigramTagger(train_sents)
print unigram_tagger.evaluate(test_sents)

# 0.811023622047
```

Используем биграммы

```
bigram_tagger = nltk.BigramTagger(train_sents)
print bigram_tagger.evaluate(test_sents)


# 0.102162862554
```

Добавим сглаживание (backoff):

```
t0 = nltk.DefaultTagger('NN')
t1 = nltk.UnigramTagger(train_sents, backoff=t0)
t2 = nltk.BigramTagger(train_sents, backoff=t1)
print t2.evaluate(test_sents)


# 0.844712448919
```

Алгоритмы,
основанные на трансформации

- Алгоритм
  - Выбрать правило, дающее наилучший результат
  - Выбрать правило, исправляющее наибольшее количество ошибок
  - и т. д.
- Шаблоны
  - Предыдущее (следующее) слово имеет тэг **X**
  - Два слова перед (после) имеют класс **X**
  - Предыдущее слово имеет класс **X**, а следующее - класс **Z**
  - **...**

Какие можно встретить трудности

- Разбиение на лексемы
  - –would/MD n’t/RB
  - –children/NNS ‘s/POS
- Неизвестные слова
  - –использовать равномерное распределение
  - –использовать априорное распределение
  - –использовать морфологию слов

Заключение

- N-граммы - один из наиболее используемых инструментов при обработке текста
- Вероятности оцениваются с помощью метода максимального правдоподобия
- Сглаживание позволяет лучше оценивать вероятности, чем ММП
- Для оценки качества модели могут использоваться внутренние и внешние оценки
- Задача определения частей речи состоит в назначении метки с частью речи каждому слову
- Параметры скрытой марковской модели могут быть определены из размеченного корпуса

Следующая лекция

- Статистические методы поиска словосочетаний

Введение в обработку текстов

Лекция 4

Методы классификации и кластеризации

Модели классификации

- Производящие (наивная байесовская модель, скрытые марковские модели)
  – предполагают независимость наблюдаемых переменных
- Разделяющие (логистическая регрессия, модель максимальной энтропии, марковские модели максимальной энтропии)

План

- Наивный байесовский классификатор
- Линейная регрессия
- Логистическая регрессия
- Модель максимальной энтропии
- Марковская модель максимальной энтропии

Задача классификации

- Есть множество классов и множество объектов, которые могут относиться к одному или более классам.
- Задача состоит в отнесении объектов с неизвестным классом к одному или более классов
- Факторы, на основе которых делается предсказание класса, называются **признаками** (feature)
- Пример, классификация людей по расам на основе цвета кожи и формы глаз.

Наивный байесовский классификатор

- Выбор наиболее вероятного значения

$$\hat{s} = \arg\max_{s \in S} P(s|f)$$

- По правилу Байеса

$$\hat{s} = \arg\max_{s \in S} \frac{P(s)P(f|s)}{P(f)} = \arg\max_{s \in S} P(s)P(f|s)$$

- Наивное предположение об условной независимости признаков

$$\hat{s} = \arg\max_{s \in S} P(s) \prod_{j=1}^{n} P(f_i|s)$$

Обучение наивного байесовского классификатора

- Метод максимального правдоподобия
- Другими словам, просто считаем

$$P(s_i) = \frac{count(s_i, w_j)}{count(w_j)} \qquad P(f_j \mid s) = \frac{count(f_j, s)}{count(s)}$$

- Алгоритм прост в реализации, но
  - Исчезновение значащих цифр → использовать сумму логарифмов вместо произведения
  - Нулевые вероятность → сглаживание или предположение о распределении $P(f_j|s)$

Пример

```
from sklearn.nayve_bayes import *

corpus = [['list of texts'],['classes']]

# initialize classifier
classifier = MultinominalNB()
```

```
# use unigrams and bigrams as features
vectorizer = CountVectorizer(ngram_range=(1,2))
y = corpus[1]
X = vectorizer.fit_transform(corpus[0])
classifier.fit(X,y) # train classifier

#transform new texts into feature vectors
unseen_texts = ["list of unseen texts"]
feature_vectors = vectorizer.transform(unseen_texts)
answers = classifier.predict(feature_vectors)
```

Модель максимальной энтропии

- Полиномиальная логистическая регрессия
- Модель классификации вида

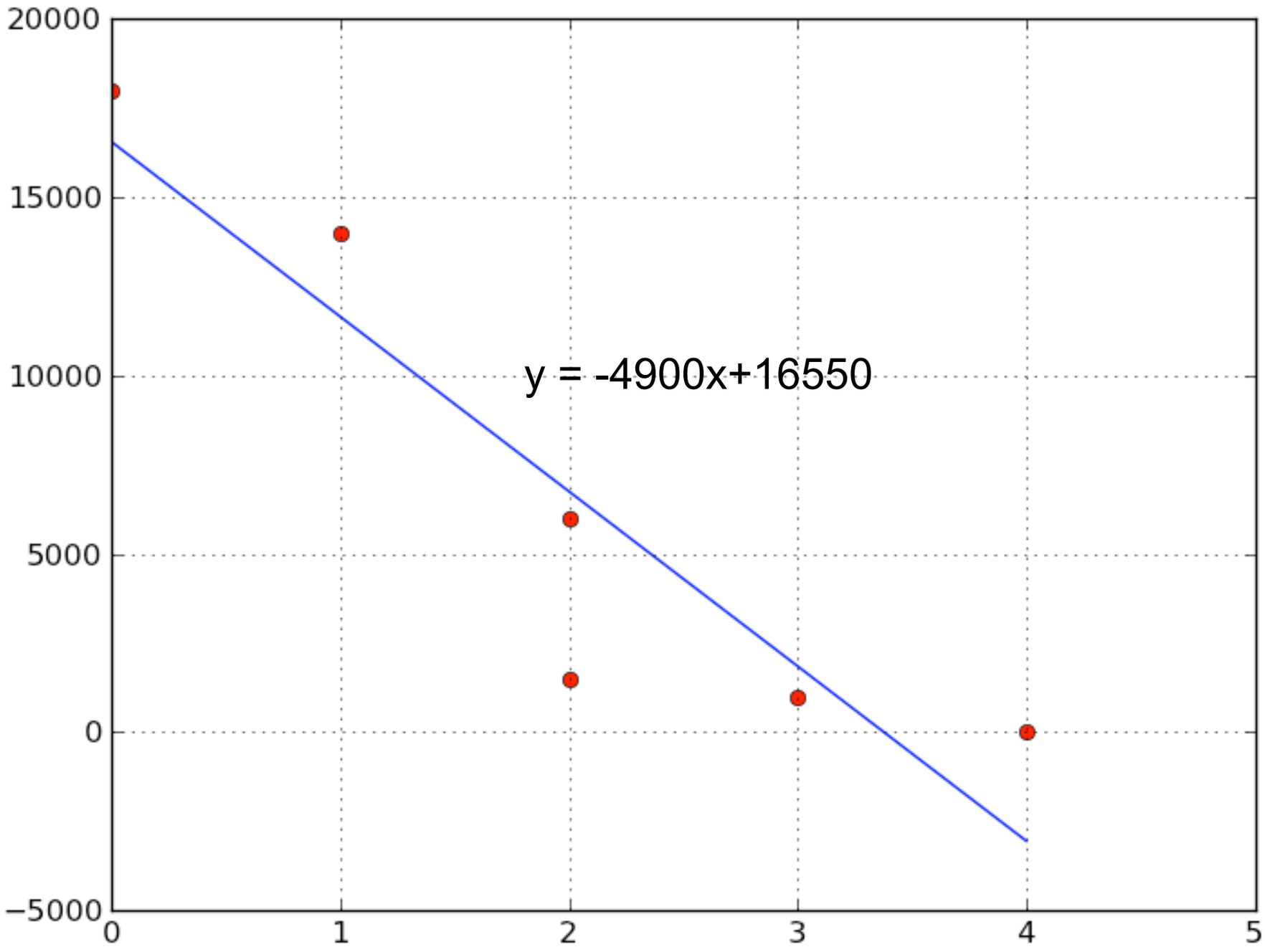
$$p(c|x) = \frac{1}{Z} exp(\sum_i w_i f_i)$$

Линейная регрессия

Кол-во неопределенных прилагательных	Прибыль сверх запрашиваемой
4	0
3	$1000
2	$1500
2	$6000
1	$14000
0	$18000

$$price = w_0 + w_1 * Num\_Adjectives$$

Линейная регрессия



Линейная регрессия

$$price = w_0 + w_1 * Num\_Adjectives + w_2 * Mortgage\_Rate + w_3 * Num\_Unsold\_Houses$$

- В терминах признаков

$$price = w_0 + \sum_{i=1}^{N} w_i \times f_i$$

- введем дополнительный признак $f_0 = 0$

$$y = \sum_{i=0}^{N} w_i \times f_i \qquad \text{или} \qquad y = w \cdot f$$

Вычисление коэффициентов

- Минимизировать квадратичную погрешность

$$cost(W) = \sum_{j=0}^{M} (y^j_{pred} - y^j_{obs})^2$$

- Вычисляется по формуле

$$W = (X^T X)^{-1} X^T \overrightarrow{y}$$

Логистическая регрессия

- Перейдем к задаче классификации
- Определить вероятность, с которой наблюдение относится к классу
- Попробуем определить вероятность через линейную модель

$$P(y = true|x) = \sum_{i=0}^{N} w_i \times f_i = w \cdot f$$

Логистическая регрессия

- Попробуем определить отношение вероятности принадлежать классу к вероятности не принадлежать классу

$$\frac{P(y = true|x)}{1 - P(y = true|x)} = w \cdot f$$

Логистическая регрессия

- Проблема с несоответствием области значений решается вводом натурального логарифма

$$ln\left(\frac{P(y=true|x)}{1-P(y=true|x)}\right) = w \cdot f$$

- Логит-преобразование

$$logit(P(x)) = ln\left(\frac{P(x)}{1-P(x)}\right)$$

- Определим вероятность ...

Логистическая регрессия

$$P(y = true|x) = \frac{e^{w \cdot f}}{1 + e^{w \cdot f}} \qquad P(y = false|x) = \frac{1}{1 + e^{w \cdot f}}$$

- Или

$$P(y = true|x) = \frac{1}{1 + e^{-w \cdot f}} \qquad P(y = false|x) = \frac{e^{-w \cdot f}}{1 + e^{-w \cdot f}}$$

- Логистическая функция

$$\frac{1}{1 + e^{-x}}$$

Логистическая регрессия

$$P(y = true|x) > P(y = false|x)$$

$$\frac{P(y = true|x)}{1 - P(y = true|x)} > 1$$

$$e^{w \cdot f} > 1$$

$$w \cdot f > 0$$

$$\sum_{i=0}^{N} w_i f_i > 0$$ разделяющая гиперплоскость

Полиномиальная логистическая регрессия

- Классификация на множество классов

$$p(c|x) = \frac{1}{Z} exp(\sum_i w_i f_i)$$

$$p(c|x) = \frac{exp\left(\sum_{i=0}^{N} w_{ci} f_i\right)}{\sum_{c' \in C} exp\left(\sum_{i=0}^{N} w_{c'i} f_i\right)}$$

Признаки

- Принято использовать бинарные признаки
- Индикаторная функция зависящая от класса и наблюдения
- Пример

$$f_1(c, x) = \begin{cases} 1 \text{ if suffix}(word_i) = \text{”ing”} \ \& \ \text{c=VBG} \\ 0 \end{cases}$$

$$f_2(c, x) = \begin{cases} 1 \text{ if } word_i = \text{”race”} \ \& \ \text{c=NN} \\ 0 \end{cases}$$

Пример

		f1	f2	f3	f4	f5	f6
VB	f	0	1	0	1	1	0
	w		0.8		0.01	0.1	
NN	f	1	0	0	0	0	1
	w	0.8					-1.3

$$p(NN|x) = \frac{e^{0.8}e^{-1.3}}{e^{0.8}e^{-1.3} + e^{0.8}e^{0.01}e^{0.1}} = 0.2$$

$$p(VB|x) = \frac{e^{0.8}e^{0.01}e^{0.1}}{e^{0.8}e^{-1.3} + e^{0.8}e^{0.01}e^{0.1}} = 0.8$$

Обучение модели

- Найти параметры, которые максимизируют логарифмическое правдоподобие на тренировочном наборе

$$\hat{w} = \arg\max_{w} \sum_{i} logP(y^i|x^i) - \sum_{j=1}^{N} \frac{w_j^2}{2\sigma_j^2}$$

- Используются методы выпуклой оптимизации
- Такой способ позволяет из всех моделей, удовлетворяющих ограничениям тестовой выборки, выбрать модель с максимальной энтропией (Berger et. al. 1996)

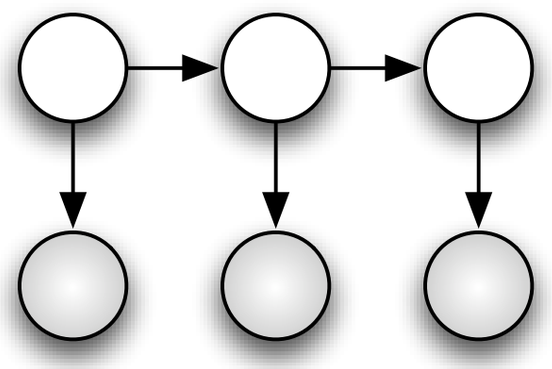
Марковская модель максимальной энтропии

- Позволяет смоделировать сложные признаки (например для определения части речи)

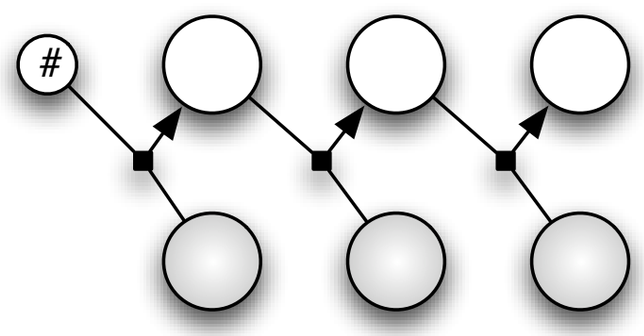
$$\hat{T} = \arg\max_t P(T|W) = \arg\max_T \prod_i P(tag_i|word_i, tag_{i-1})$$

- Сравнить с марковской моделью

$$\hat{T} = \arg\max_t P(T|W) = \arg\max_T \prod_i P(word_i|tag_i)P(tag_i, tag_{i-1})$$

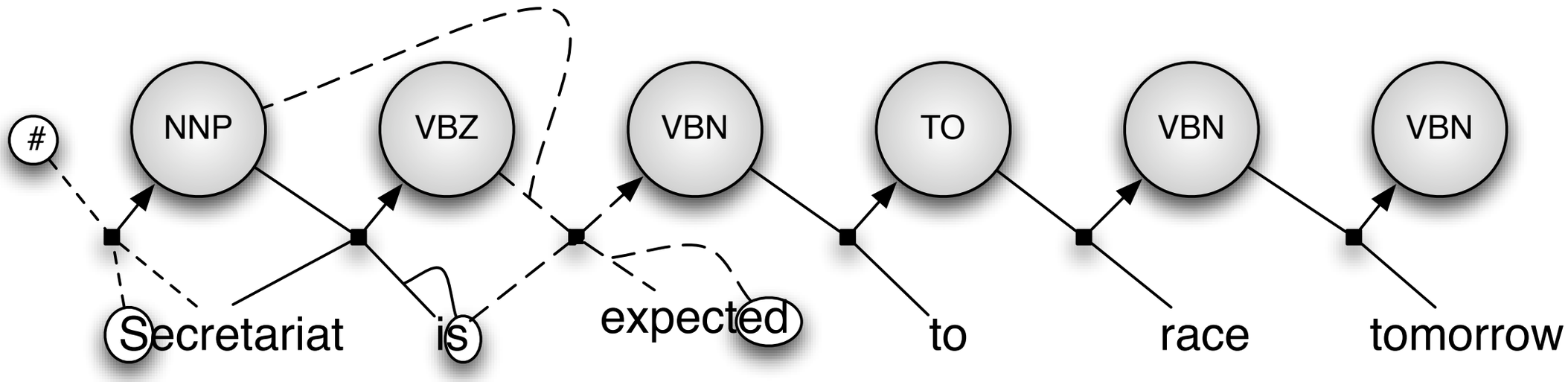


Скрытые марковские модели



Скрытые марковские модели максимальной энтропии

Признаки в MEMM



$$P(q|q', o) = \frac{1}{Z(o, q')} exp\left(\sum_i w_i f_i(o, q)\right)$$

Декодирование и обучение

- Декодирование - алгоритм Витерби, где на каждом шаге вычисляется

$$v_t(j) = max_{i=1}^{N} v_{t-1}(i) P(s_j|s_i, o_t), 1 \leq j \leq N, 1 < t \leq T$$

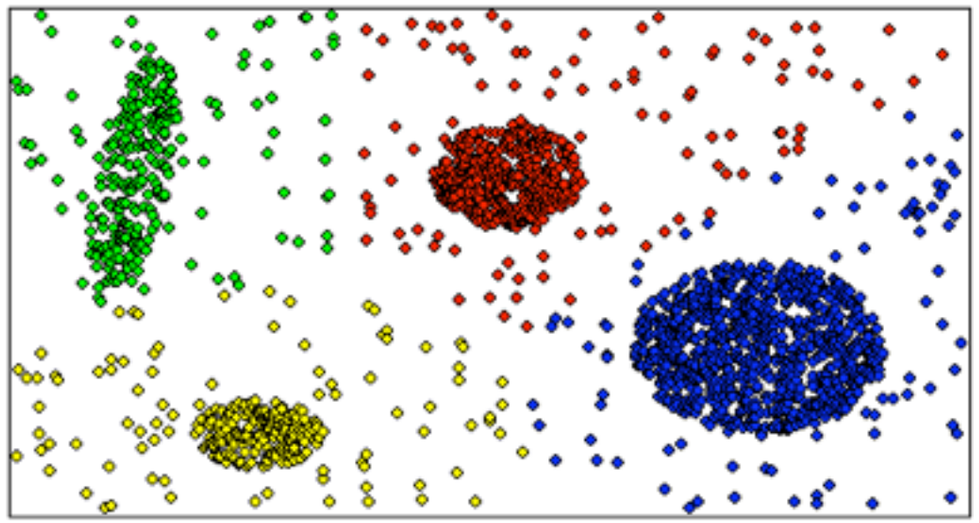
- Обучение аналогично логистической регрессии

Кластеризация

обучение без учителя

Мотивация

- Данные можно разбить на несколько групп по принципу схожести



- Поиск схожих документов
- Поиск схожих слов и терминов
- Реферирование документов
- Для задач обучения с учителем
  - –Кластер, как признак для обучения
  - –Кластер, как набор данных для обучения

Вход для алгоритмов

- Пусть каждый документ $\{x_1, x_2, \ldots, x_k\}$ представлен вектором $x_i = (x_{i_1}, x_{i_2}, \ldots, x_{i_n})$ в пространстве $X \subseteq R^n$
- Задается расстояние между векторами
  - –Евклидово $d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \ldots + (p_n - q_n)^2} = \sqrt{\sum_{k=1}^{n} (p_k - q_k)^2}$
  - –Чебышева $l_\infty(\vec{x}, \vec{y}) = \max_{i=1,\ldots,n} |x_i - y_i|$
  - –Хэмминга $d_{ij} = \sum_{k=1}^{p} |x_{ik} - x_{jk}|.$
  - –Минковского $\rho(x,y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$
  - –...

Взвешивание слов

- Частота слова в документе (tf)
- tf-idf

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \times \mathrm{idf}(t, D)$$

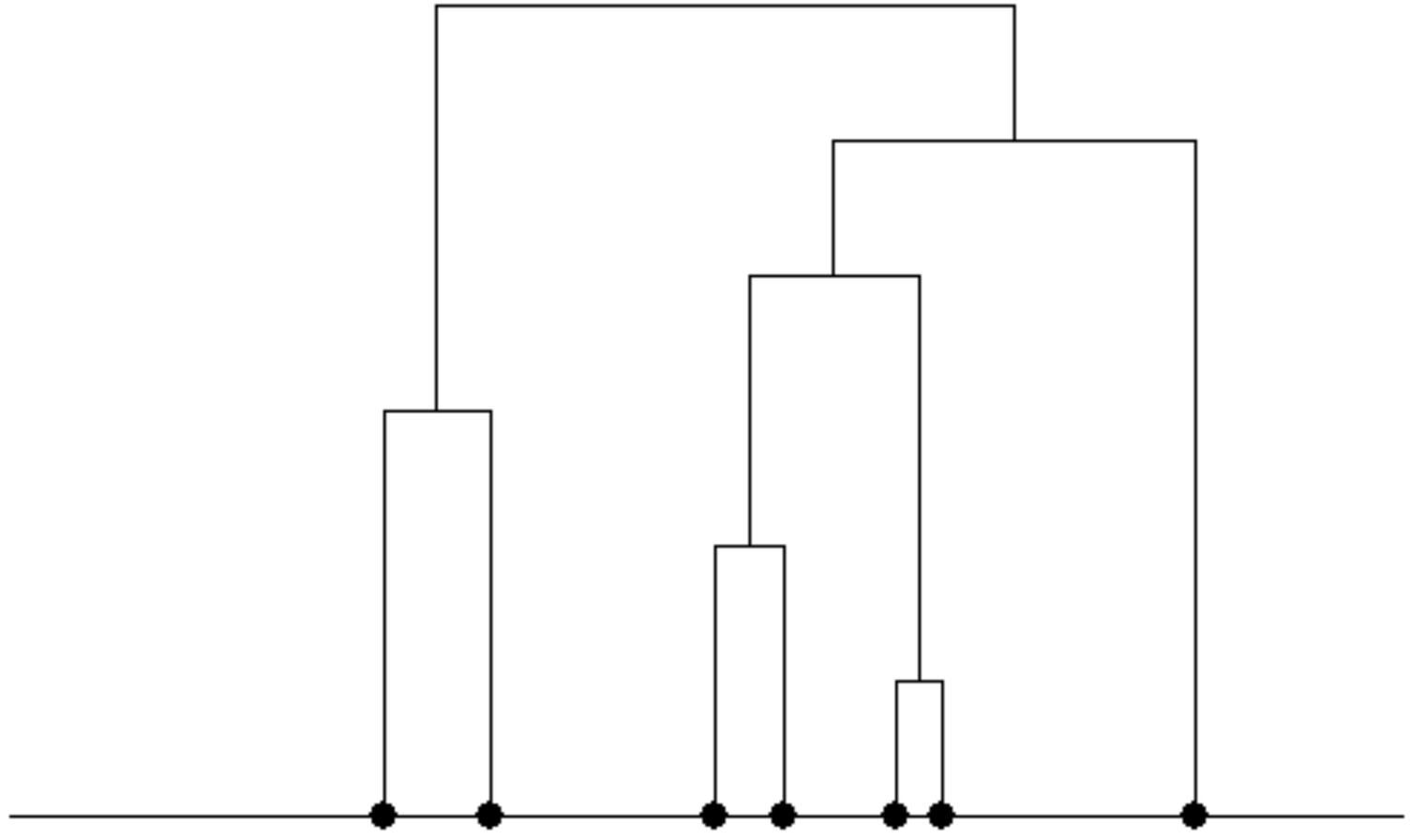
$$\mathrm{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

План

- Иерархическая кластеризация
- k-means
- Affinity propogation
- MeanShift
- Спектральная кластеризация
- WARD
- DBSCAN

Иерархическая кластеризация

- Строится дендрограмма - дерево обозначающее вложенную последовательность кластеров

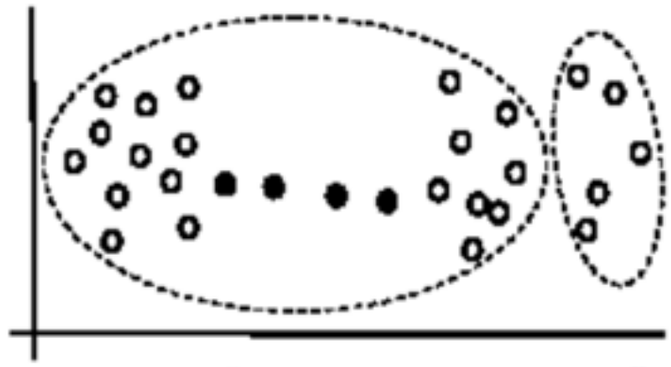


Типы иерархической кластеризации

- Агломеративная
  - –каждая точка - кластер
  - –объединяем два наиболее близких кластера в один
  - –останавливаемся, когда все данные объединены в один кластер
- Дивизимная
  - –все данные - один кластер
  - –разделяем наименее плотный кластер на два
  - –останавливаемся, когда достигли минимального допустимого размера

Расстояние между кластерами

- Между двумя ближайшими точками
  - –Можно получить кластеры произвольной формы
  - –“Эффект цепи”



- Между двумя самыми дальними точками
  - –Чувствителен к выбросам
- Среднее расстояние

K-средних

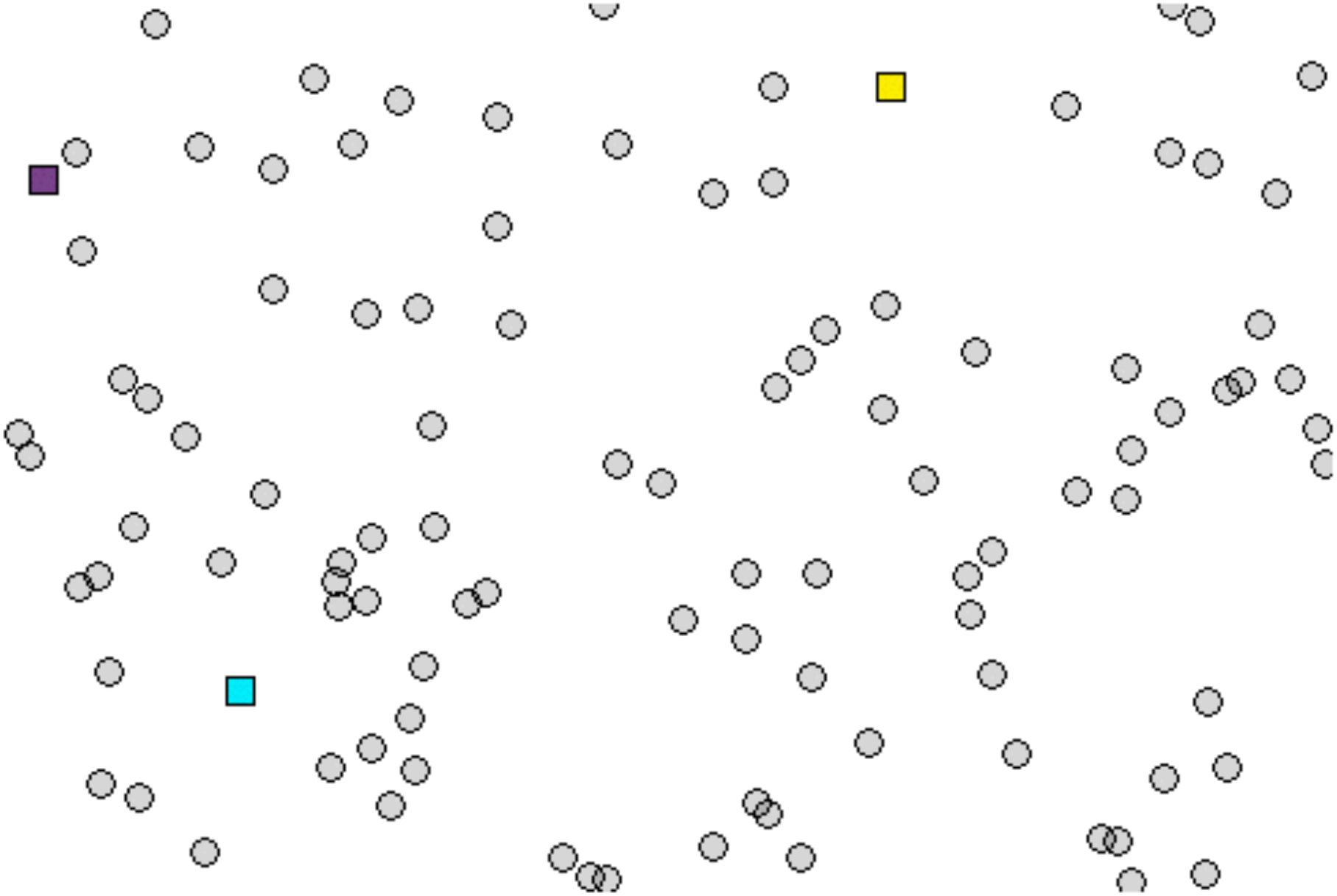
- Алгоритм k-means разбивает данные на k кластеров
  - –Каждый кластер имеет центр - центроид
  - –Параметр k - задается вручную
- Алгоритм
  1. Выбираются k точек в качестве начальных центроидов
  2. Сопоставить каждой точке ближайший центроид
  3. Пересчитать центроиды
  4. Если алгоритм не сошелся перейти на шаг 2

Критерий останова

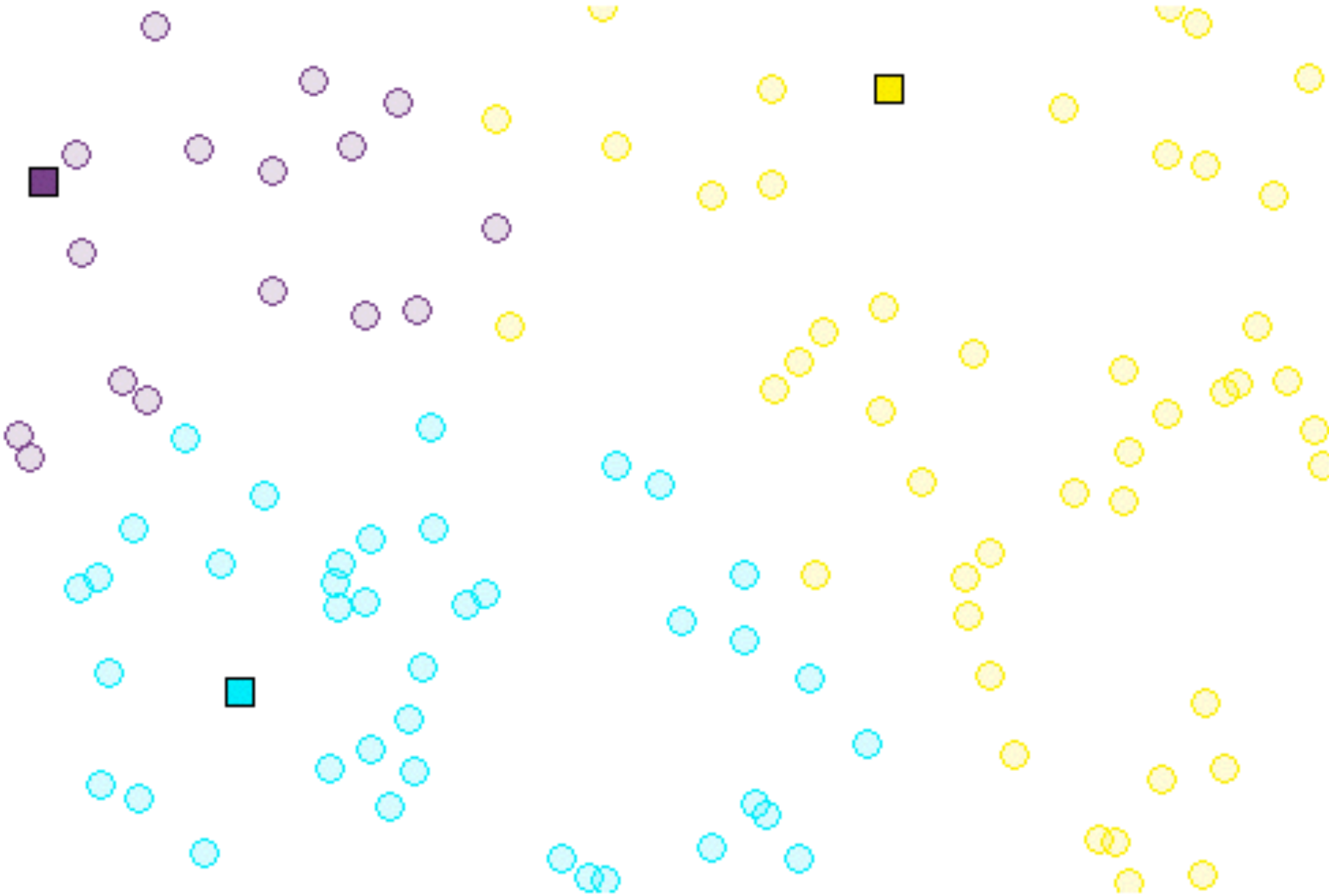
- Нет перехода точек в другой кластер
- Нет (незначительно) изменение центроидов
- Мало убывает погрешность (sum of squared error)

$$SSE = \sum_{j=1}^{k} \sum_{x \in C_j} dist(x, m_j)^2$$

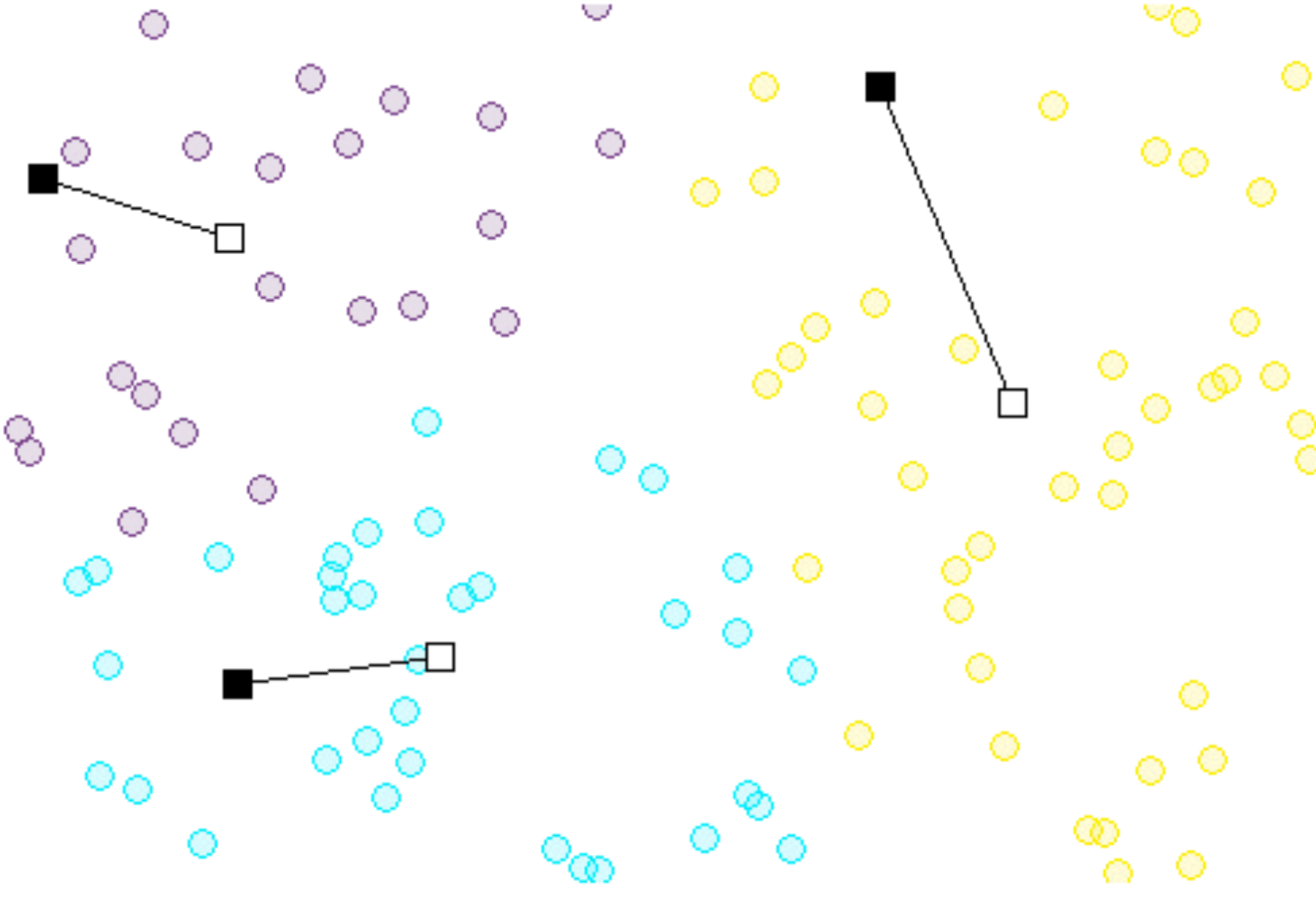
K-средних. Пример



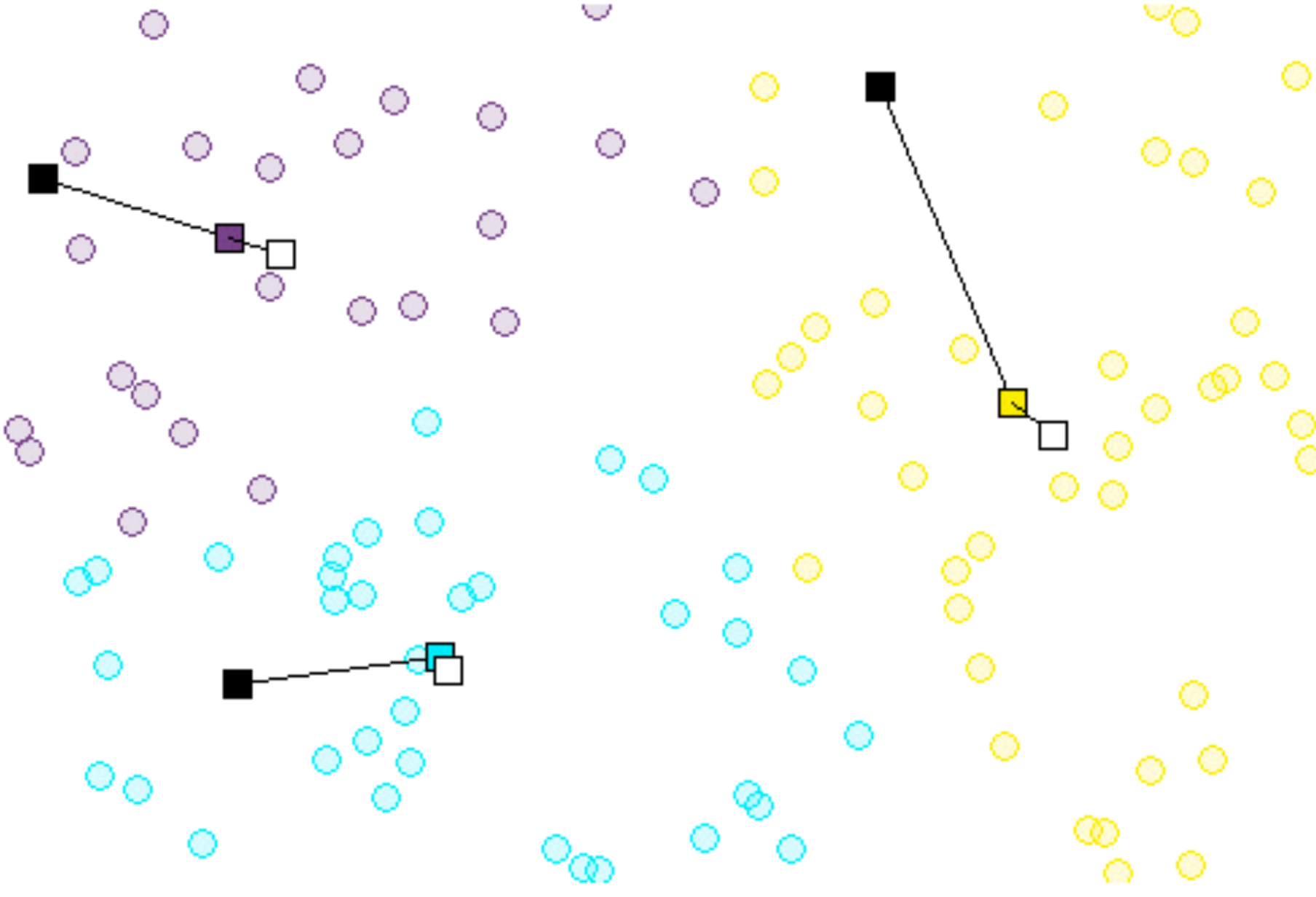
K-средних. Пример



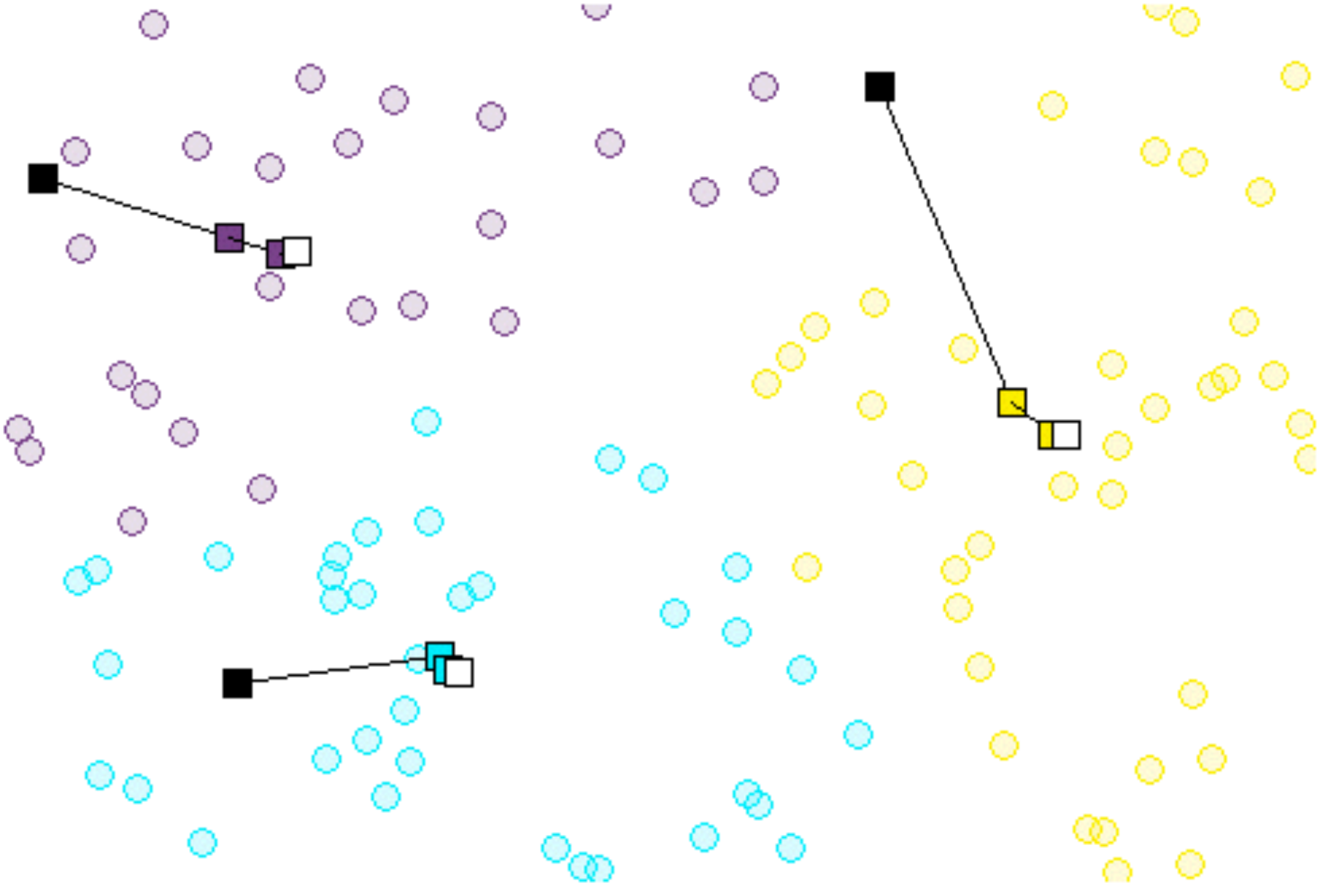
K-средних. Пример



K-средних. Пример



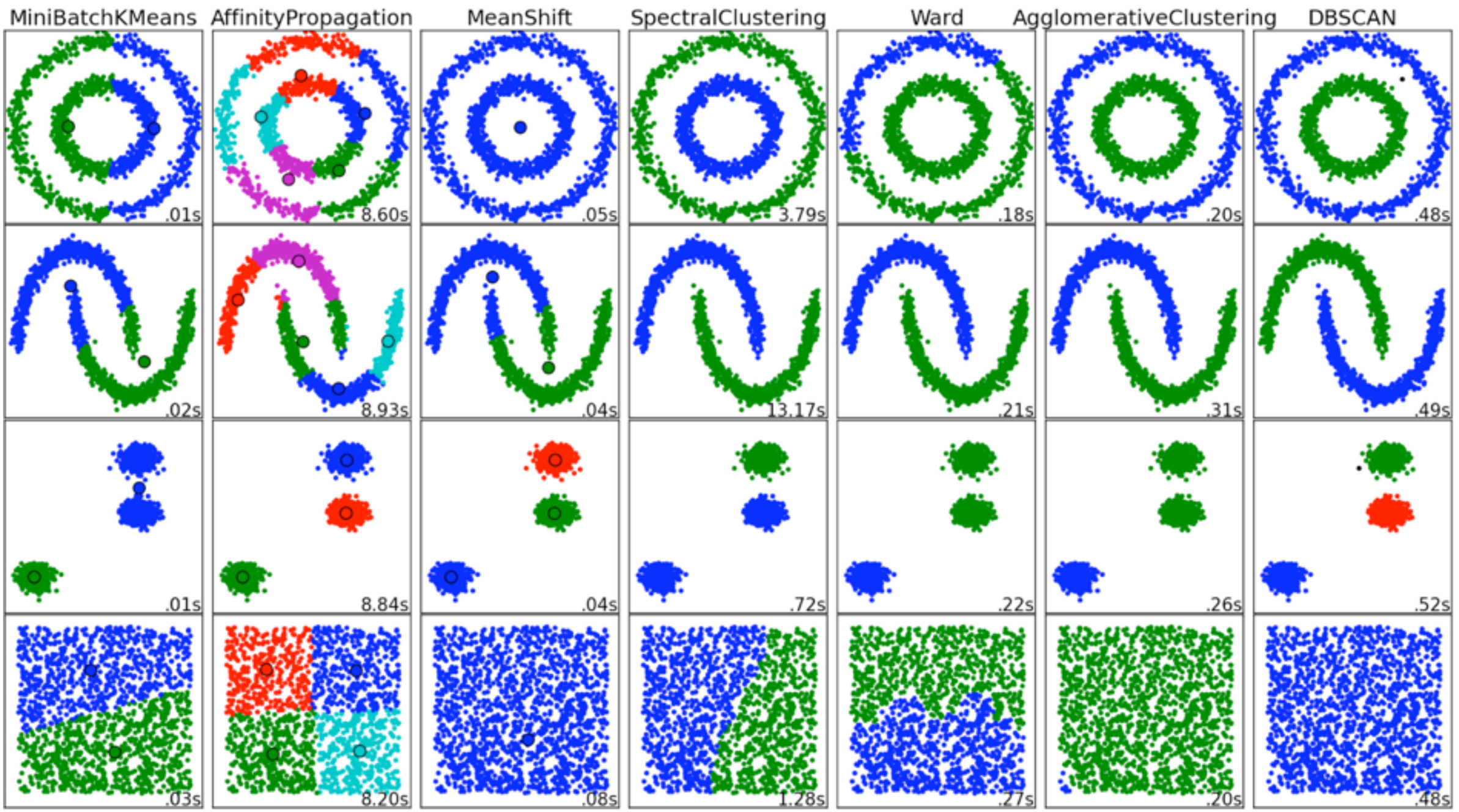
K-средних. Пример



Проблемы

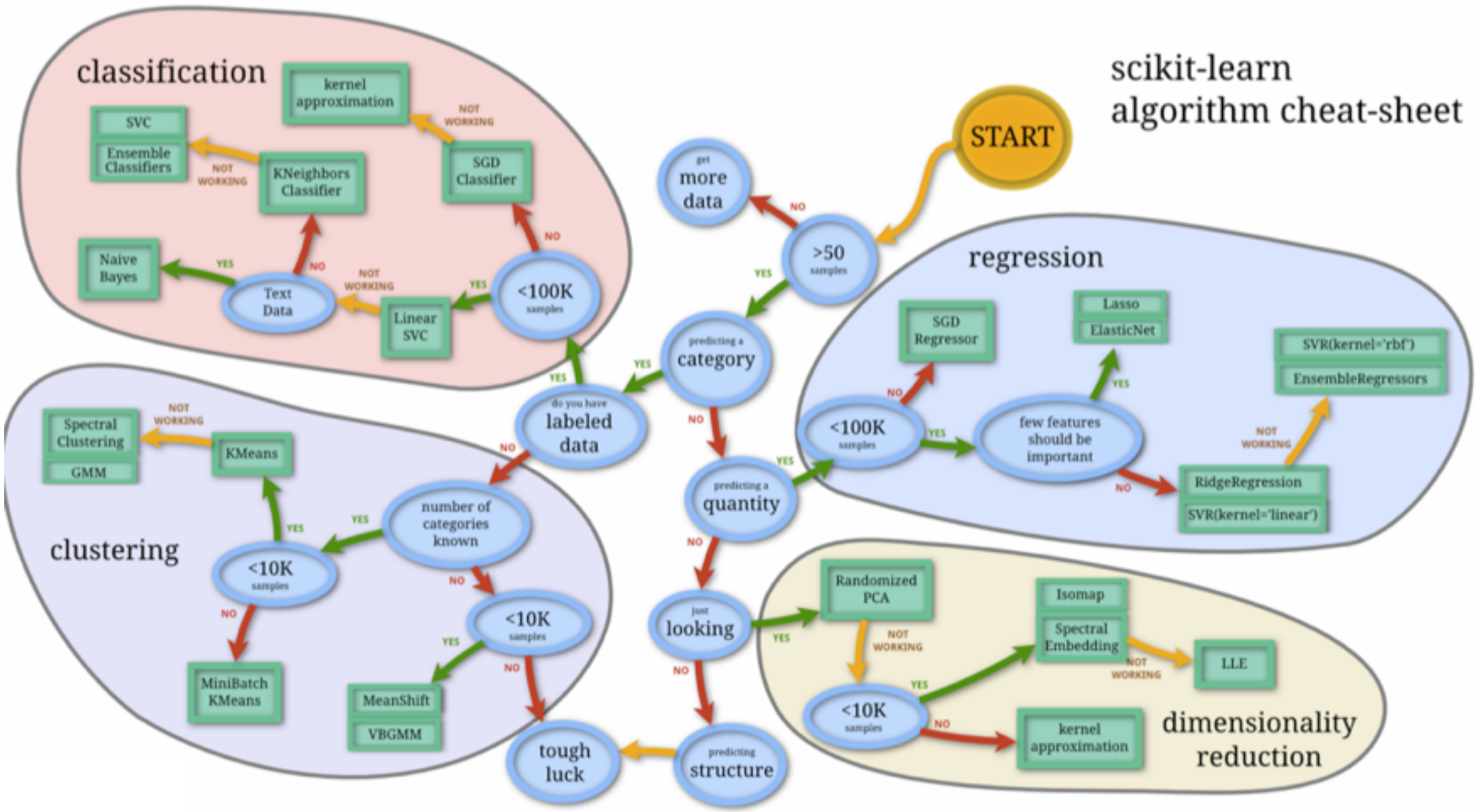
- Алгоритм чувствителен к начальному выбору центроидов
  - –запуск с различной начальной инициализацией и выбор варианта с наиболее плотными кластерами
- Чувствителен к выбросам
  - –можно фильтровать выбросы
- Не подходит для нахождения кластеров, не являющихся элипсоидами
  - –преобразование пространства

Какой алгоритм выбрать



*http://scikit-learn.org/stable/modules/clustering.html

Что делать



*http://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

Следующая лекция

- Статистические методы поиска словосочетаний

Основы обработки текстов

Лекция 5

Статистические методы поиска словосочетаний

Словосочетания/коллокации

- Для данной лекции **Словосочетания = Коллокации = Фразеологические обороты** - цепочки слов состоящие из двух или более элементов, имеющее признаки синтаксически и семантически целостной единицы, в котором выбор одного из компонентов осуществляется по смыслу, а выбор второго зависит от выбора первого
- Примеры:
  - Крепкий чай (не “сильный чай”)
  - Схема Бернулли (сравнить значения со значениями “Схема” и “Бернулли”)

Приложения

- Сравнения корпусов текстов
  - кластеризация документов в информационном поиске
  - Поиск плагиата
- Синтаксический разбор
- Компьютерная лексикография
- Генерация естественного языка
- Машинный перевод
- Выделение ключевых слов (терминов)

Выделение словосочетаний

Поиск кандидатов

- Основная предпосылка
  - Если два (или более) слова встречаются вместе часто, то, вероятно, это словосочетание
- Инструменты
  - Частота
  - Частота и фильтрация по тэгам
  - Математическое ожидание и дисперсия

Частота

- Подсчет частоты n-грам
- Выбрать наиболее встречающиеся
- Результат
  - Корпус: New York Times
    - August-November, 1990
  - Результат не интересен

$C(w^1\ w^2)$	w^1	w^2
80871	of	the
58841	in	the
26430	to	the
21842	on	the
21839	for	the
18568	and	the
16121	that	the
15630	at	the
15494	to	be
13899	in	a
13689	of	a
13361	by	the
13183	with	the
12622	from	the
11428	New	York
10007	he	said
9775	as	a
9231	is	a
8753	has	been
8573	for	a

Частота с фильтрацией по тэгам

- Подсчет частоты n-грам
- определить части речи
- фильтрация кандидатов по шаблонам для частей речи
- выбрать наиболее встречающиеся

Tag Pattern	Example
A N	*linear function*
N N	*regression coefficients*
A A N	*Gaussian random variable*
A N N	*cumulative distribution function*
N A N	*mean squared error*
N N N	*class probability function*
N P N	*degrees of freedom*

Частота с фильтрацией по тэгам

$C(w^1\ w^2)$	w^1	w^2	Tag Pattern
11487	New	York	A N
7261	United	States	A N
5412	Los	Angeles	N N
3301	last	year	A N
3191	Saudi	Arabia	N N
2699	last	week	A N
2514	vice	president	A N
2378	Persian	Gulf	A N
2161	San	Francisco	N N
2106	President	Bush	N N
2001	Middle	East	A N
1942	Saddam	Hussein	N N
1867	Soviet	Union	A N
1850	White	House	A N
1633	United	Nations	A N
1337	York	City	N N
1328	oil	prices	N N
1210	next	year	A N
1074	chief	executive	A N
1073	real	estate	A N

Мат. ожидание и дисперсия

- Часто устойчивые пары слов находятся не рядом
  - Пример
    - She **knoked** on his **door**
    - They **knoked** on the **door**
    - a man **knocked** on the metal front **door**
  - Важно это понимать, например при генерации текстов

Мат. ожидание и дисперсия

- Техника
  - Рассмотрим все пары слов в некотором окне
  - Посчитаем расстояние между словами
- Меры
  - Мат. ожидание (возможно отрицательное)
    - Показывает на сколько часто два слова встречаются вместе
  - Дисперсия (среднеквадратичное отклонение)
    - Вариабельность позиции

Мат. ожидание и дисперсия

She knocked on his door

Пары в окне длиной 3:

She knocked She on She his
knocked on knocked his knocked door
on his on door
his door

$$\overline{d} = \frac{\sum_{i=1}^{n} d_i}{n}$$

$$s^2 = \frac{\sum_{i=1}^{n} (d_i - \overline{d})^2}{n-1}$$

n - число раз, когда два слова встретились
d_i - смещение между словами
$\overline{d}$ - выборочное среднее смещений

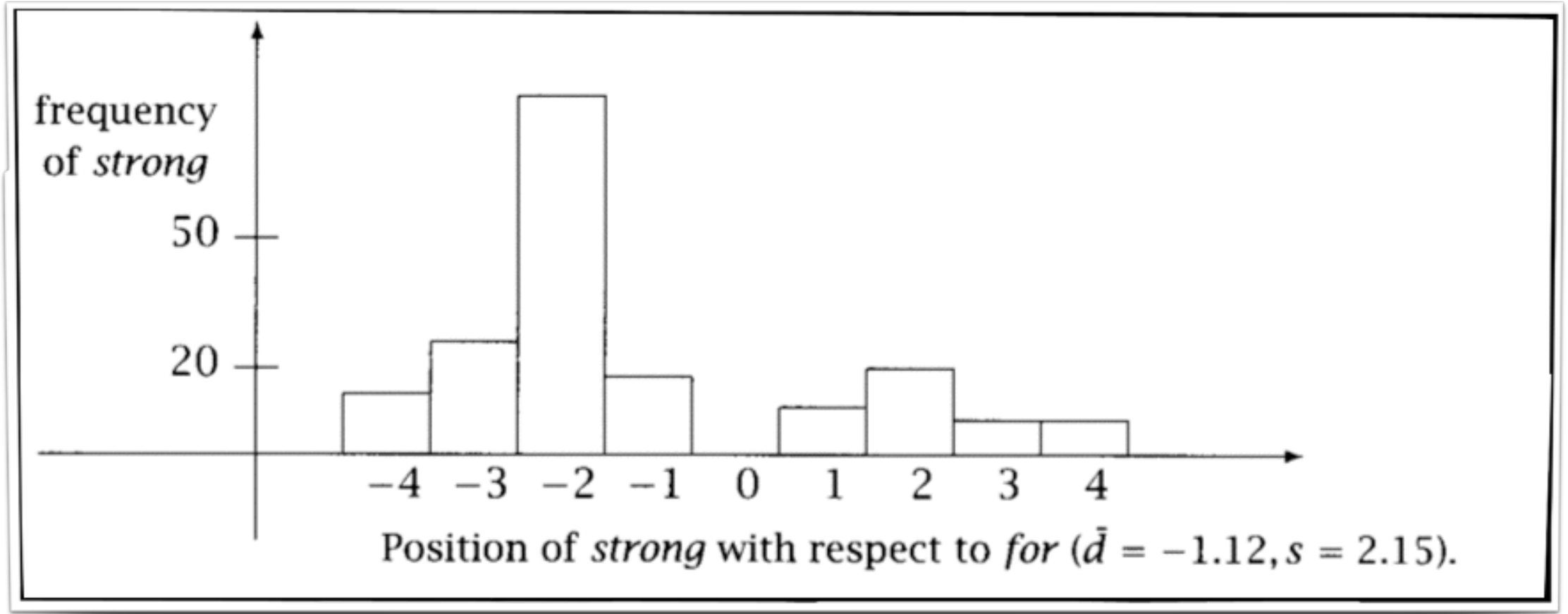
Пример: knocked ... door

$$\overline{d} = \frac{1}{3}(3+3+5) \approx 3.67$$

$$s = \sqrt{\frac{1}{2}((3-3.67)^2 + (3-3.67)^2 + (5-3.67)^2)} \approx 1.15$$

Гисторамма

- Пример: strong ... for
  - “strong [business] support for”



Position of *strong* with respect to *for* ($\bar{d} = -1.12, s = 2.15$).

Пример

s	$\bar{d}$	Count	Word 1	Word 2
0.43	0.97	11657	New	York
0.48	1.83	24	previous	games
0.15	2.98	46	minus	points
0.49	3.87	131	hundreds	dollars
4.03	0.44	36	editorial	Atlanta
4.03	0.00	78	ring	New
3.96	0.19	119	point	hundredth
3.96	0.29	106	subscribers	by
1.07	1.45	80	strong	support
1.13	2.57	7	powerful	organizations
1.01	2.00	112	Richard	Nixon
1.05	0.00	10	Garrison	said

- Большое среднеквадратичное отклонение показывает, что сочетание не очень интересное

Проверка статистических гипотез

Проверка статистических гипотез

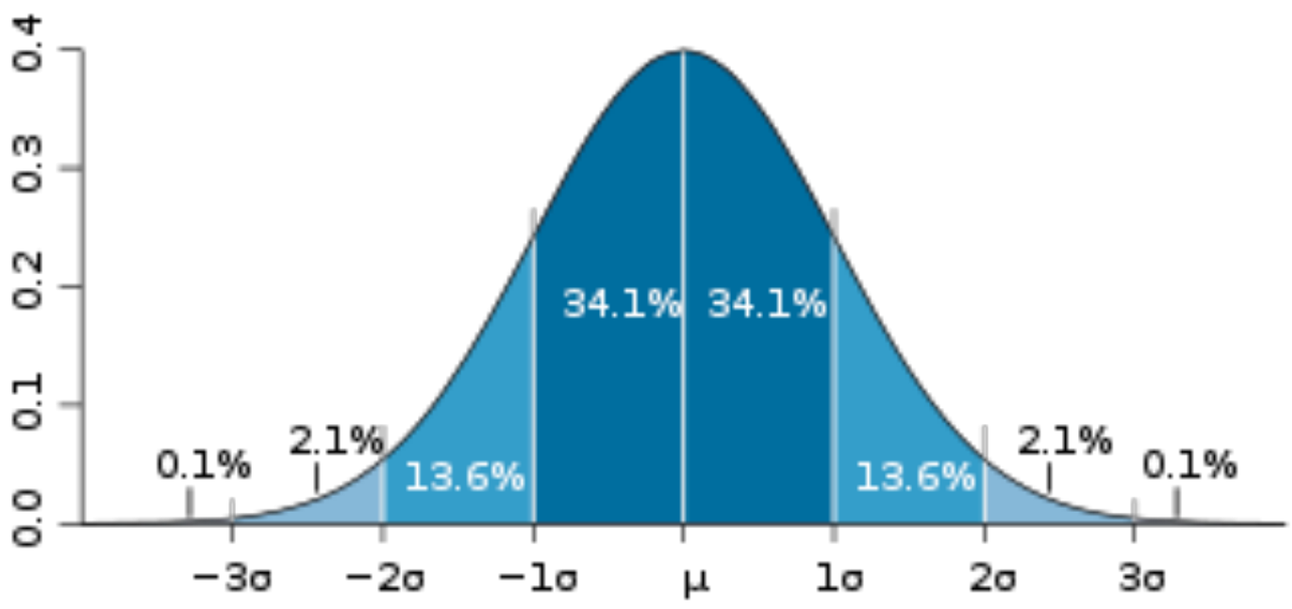
- **Основная идея**: слова словосочетания встречаются вместе значительно чаще чем просто случайно
- **Инструменты**:
  - t-критерий Стьюдента (t-test)
  - Критерий согласия Пирсона (Хи-квадрат)
  - Критерий отношения правдоподобия (Likelihood ratio test)

Нулевая гипотеза

- H_0-слова встречаются независимо
  - $P(w_1,w_2)=P(w_1)P(w_2)$
- Какова вероятность получить словосочетание w_1w_2, при условии что гипотеза верна?
  - $p=P(w_1w_2|H_0)$

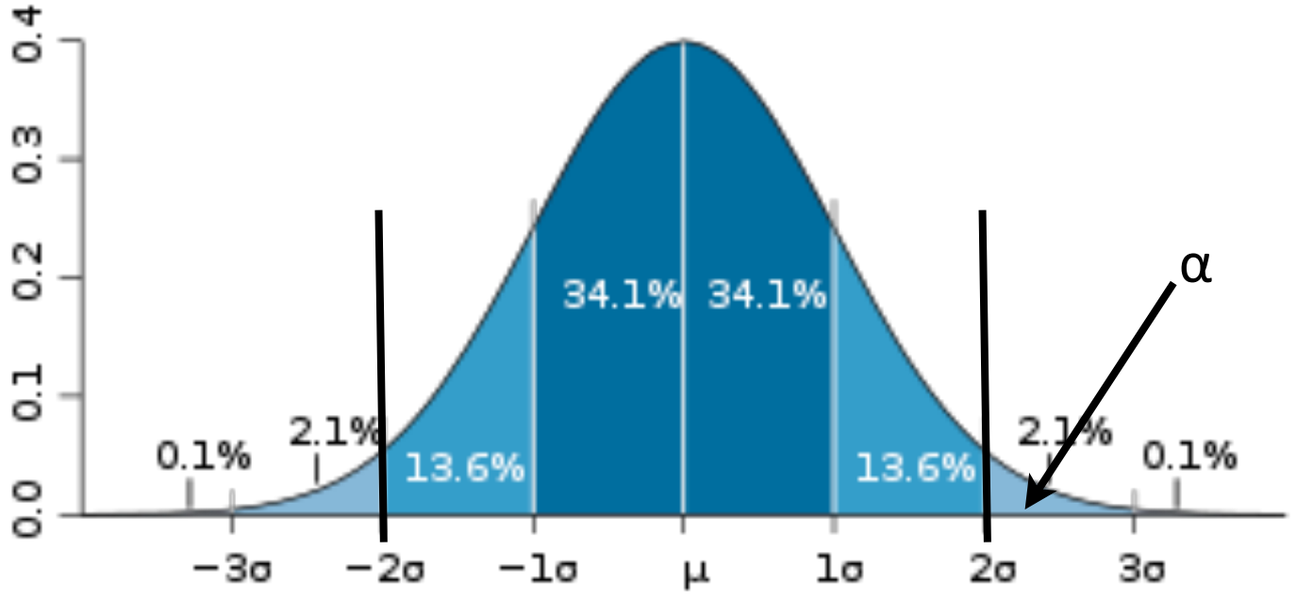
T-критерий Стьюдента

- Разработан Уильямом Госсетом для оценки качества пива Гиннесс
- Рассмотрим распределение выборочного среднего у всевозможных выборок длины n
- По ЦПТ, при больших n:



T-критерий Стьюдента

- Если для наших данных наблюдаемое выборочное среднее сильно отклоняется от ожидаемого при нулевой гипотезе, то с вероятностью p гипотеза не верна
- α - ошибка первого рода
- p < α - отвергаем гипотезу



T-критерий Стьюдента

- T-статистика

$$t = \frac{\overline{x} - \mu}{\sqrt{\frac{s^2}{N}}}$$

μ -ожидаемое мат. ожидание
$\overline{x}$ -выборочное среднее
s^2-выборочная дисперсия
N-размер выборки

- Распределение Стьюдента (стремится к нормальному при больших N)

	P	0.05	0.025	0.01	0.005	0.001	0.0005
	C	90%	95%	98%	99%	99.8%	99.9%
d.f.	1	6.314	12.71	31.82	63.66	318.3	636.6
	10	1.812	2.228	2.764	3.169	4.144	4.587
	20	1.725	2.086	2.528	2.845	3.552	3.850
(z)	∞	1.645	1.960	2.326	2.576	3.091	3.291

T-критерий. Пример

- Предположим,что средний рост мужчин в популяции равен 158 см
- Для выборки из 200 мужчин $\overline{x} = 169, s^2 = 2600$
- Тогда $t = \frac{169 - 158}{\sqrt{\frac{2600}{200}}} = 3.05$

	P	0.05	0.025	0.01	0.005	0.001	0.0005
	C	90%	95%	98%	99%	99.8%	99.9%
d.f.	1	6.314	12.71	31.82	63.66	318.3	636.6
	10	1.812	2.228	2.764	3.169	4.144	4.587
	20	1.725	2.086	2.528	2.845	3.552	3.850
(z)	∞	1.645	1.960	2.326	2.576	3.091	3.291

- Для α=0.005:
- 3.05>2.576
- отвергаем гипотезу

T-критерий для словосочетаний

- Пусть нулевая гипотеза верна
- Рассмотрим процесс случайной генерации биграмм, если встретили биграмму w_1w_2 (с вероятностью p) генерируем 1, в противном случае 0 (схема Бернулли) **биномиальное распределение** ➡
- мат. ожидание = p
- дисперсия = p(1-p)≈p при малых p

$$t = \frac{\overline{x} - \mu}{\sqrt{\frac{s^2}{N}}}$$

μ =H_0=P(w_1)P(w_2)
$\overline{x}$ -отношение w_1w_2 к общему кол-ву биргамм
s^2-отношение w_1w_2 к общему кол-ву биргамм
N-общее количество биграмм

Пример

- new companies (встретилась 8 раз)

$$P(new) = \frac{15828}{14307668} \quad P(companies) = \frac{4675}{14307668}$$

$$H_0 : P(new\ companies) = P(new)P(companies) = \frac{15828}{14307668} \times \frac{4675}{14307668} \approx 3.615 \times 10^{-7}$$

$$\overline{x} = \frac{8}{14307668} \approx 5.591 \times 10^{-7}$$

$$t = \frac{\overline{x} - \mu}{\sqrt{\frac{s^2}{N}}} \approx \frac{5.591 \times 10^{-7} - 3.615 \times 10^{-7}}{\sqrt{\frac{5.591 \times 10^{-7}}{14307668}}} \approx 0.999932$$

- не можем отвергнуть гипотезу

Для корпуса

t	$C(w^1)$	$C(w^2)$	$C(w^1\ w^2)$	w^1	w^2
4.4721	42	20	20	Ayatollah	Ruhollah
4.4721	41	27	20	Bette	Midler
4.4720	30	117	20	Agatha	Christie
4.4720	77	59	20	videocassette	recorder
4.4720	24	320	20	unsalted	butter
2.3714	14907	9017	20	first	made
2.2446	13484	10570	20	over	many
1.3685	14734	13478	20	into	them
1.2176	14093	14776	20	like	people
0.8036	15019	15629	20	time	last

Хи-квадрат

- Сравнить наблюдаемые частоты в корпусе с ожидаемыми частотами при верной гипотезе о независимости
- Если различие большое - отвергаем гипотезу
- (Выборка должна быть большая)

X^2 - общая формула

- Меры:
  - E_{ij} = ожидаемое кол-во коллокаций
  - O_{ij} = наблюдаемое кол-во коллокаций

$$\chi^2 = \sum_{i,j} \frac{O_{ij} - E_{ij}}{E_{ij}}$$

- Результат
  - Смотрим число в таблице для распределения X^2
    - если число в таблице меньше, то отвергаем гипотезу

X^2 - для биграмм

	$w_1 = new$	$w_1 \neq new$
$w_2 = companies$	8 (*new companies*)	4667 (e.g., *old companies*)
$w_2 \neq companies$	15820 (e.g., *new machines*)	14287181 (e.g., *old machines*)

$$\chi^2 = \frac{N(O_{11}O_{22} - O_{12}O_{21})^2}{(O_{11} + O_{12})(O_{11} + O_{21})(O_{12} + O_{22})(O_{21} + O_{22})}$$

$$\frac{14307668(8 \times 14287181 - 4667 \times 15820)^2}{(8 + 4667)(8 + 15820)(4667 + 14287181)(15820 + 14287181)} \approx 1.55$$

	p	0.99	0.95	0.10	0.05	0.01	0.005	0.001
d.f.	1	0.00016	0.0039	2.71	3.84	6.63	7.88	10.83
	2	0.020	0.10	4.60	5.99	9.21	10.60	13.82
	3	0.115	0.35	6.25	7.81	11.34	12.84	16.27
	4	0.297	0.71	7.78	9.49	13.28	14.86	18.47
	100	70.06	77.93	118.5	124.3	135.8	140.2	149.4

Критерий отношения правдоподобия

- На сколько более правдоподобна одна гипотеза, чем другая
- H_1: $P(w_2|w_1) = p = P(w_2|\neg w_1)$
- H_2: $P(w_2|w_1) = p_1 \neq p_2 = P(w_2|\neg w_1)$

$$(p_1 >> p_2)$$

Критерий отношения правдоподобия

	H_1	H_2
$P(w_2\|w_1)$	$p = \frac{c_2}{N}$	$p = \frac{c_{12}}{c_1}$
$P(w_2\|\neg w_1)$	$p = \frac{c_2}{N}$	$p = \frac{c_2 - c_{12}}{N - c_1}$

- Так же как в t-критерии предполагаем схему Бернулли и биномиальное распределение

$$b(k; n, x) = C_n^k x^k (1-x)^{n-k}$$

	H_1	H_2
c_{12} из c_1 биграм-это w_1w_2	$b(c_{12}; c_1, p)$	$b(c_{12}; c_1, p_1)$
c_2-c_{12} из N-c_1 биграм-это не w_1w_2	$b(c_2 - c_{12}; N - c_1, p)$	$b(c_2 - c_{12}; N - c_1, p_2)$

$$L(H_1) = b(c_{12}; c_1, p)b(c2 - c_{12}; N - c_1, p)$$
$$L(H_2) = b(c_{12}; c_1, p_1)b(c2 - c_{12}; N - c_1, p_2)$$

Отношение правдоподобия

$$
\begin{aligned}
\log \lambda &= \log \frac{L(H_1)}{L(H_2)} \\
&= \log \frac{\mathrm{b}(c_{12}, c_1, p)\mathrm{b}(c_2 - c_{12}, N - c_1, p)}{\mathrm{b}(c_{12}, c_1, p_1)\mathrm{b}(c_2 - c_{12}, N - c_1, p_2)} \\
&= \log L(c_{12}, c_1, p) + \log L(c_2 - c_{12}, N - c_1, p) \\
&\quad - \log L(c_{12}, c_1, p_1) - \log L(c_2 - c_{12}, N - c_1, p_2)
\end{aligned}
$$

где $L(k, n, x) = x^k(1 - x)^{n-k}$

Результат для корпуса

$-2\log\lambda$	$C(w^1)$	$C(w^2)$	$C(w^1w^2)$	w^1	w^2
1291.42	12593	932	150	most	powerful
99.31	379	932	10	politically	powerful
82.96	932	934	10	powerful	computers
80.39	932	3424	13	powerful	force
57.27	932	291	6	powerful	symbol
51.66	932	40	4	powerful	lobbies
51.52	171	932	5	economically	powerful
51.05	932	43	4	powerful	magnet
50.83	4458	932	10	less	powerful
50.75	6252	932	11	very	powerful
49.36	932	2064	8	powerful	position
48.78	932	591	6	powerful	machines
47.42	932	2339	8	powerful	computer
43.23	932	16	3	powerful	magnets
43.10	932	396	5	powerful	chip
40.45	932	3694	8	powerful	men
36.36	932	47	3	powerful	486
36.15	932	268	4	powerful	neighbor
35.24	932	5245	8	powerful	political
34.15	932	3	2	powerful	cudgels

- $-2log\lambda$ имеет распределение χ^2

Заключение

- Поиск словосочетаний может улучшить качество многих приложений
- Для поиска словосочетаний могут использоваться простые статистические модели в комбинации эвристиками
- Для проверки “значимости” словосочетаний применяются методы проверки статистических гипотез

Следующая лекция

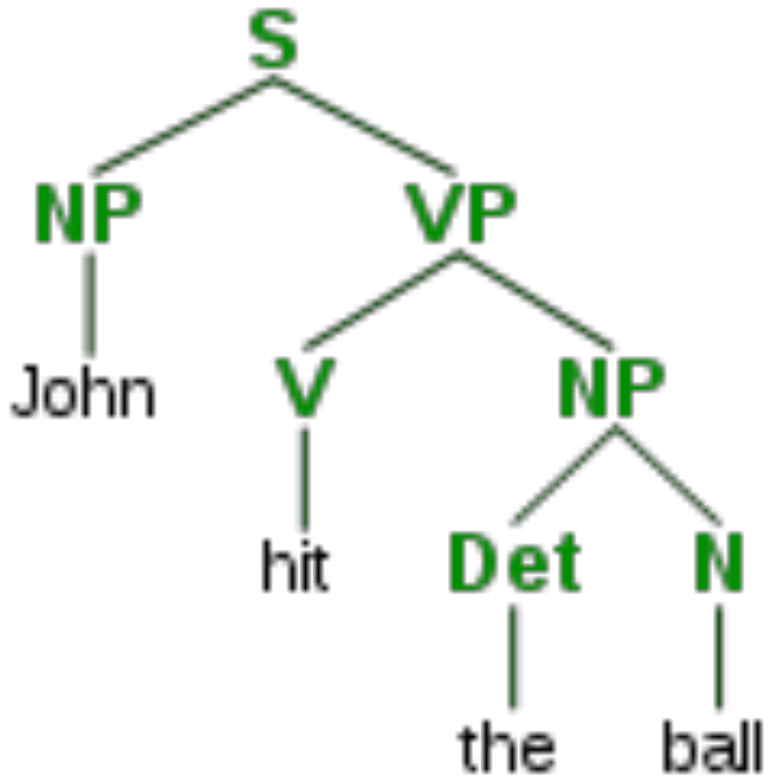
- Синтаксический анализ

Основы обработки текстов

Лекция 6

Формальные грамматики и синтаксический анализ

Пример синтаксического разбора



Где может быть полезно знание синтаксиса?

- Машинный перевод
- Генерация текста
  - –диалоговые системы
- Извлечение информации
- Понимание на что/кого направлено эмоциональное высказывание
- ...

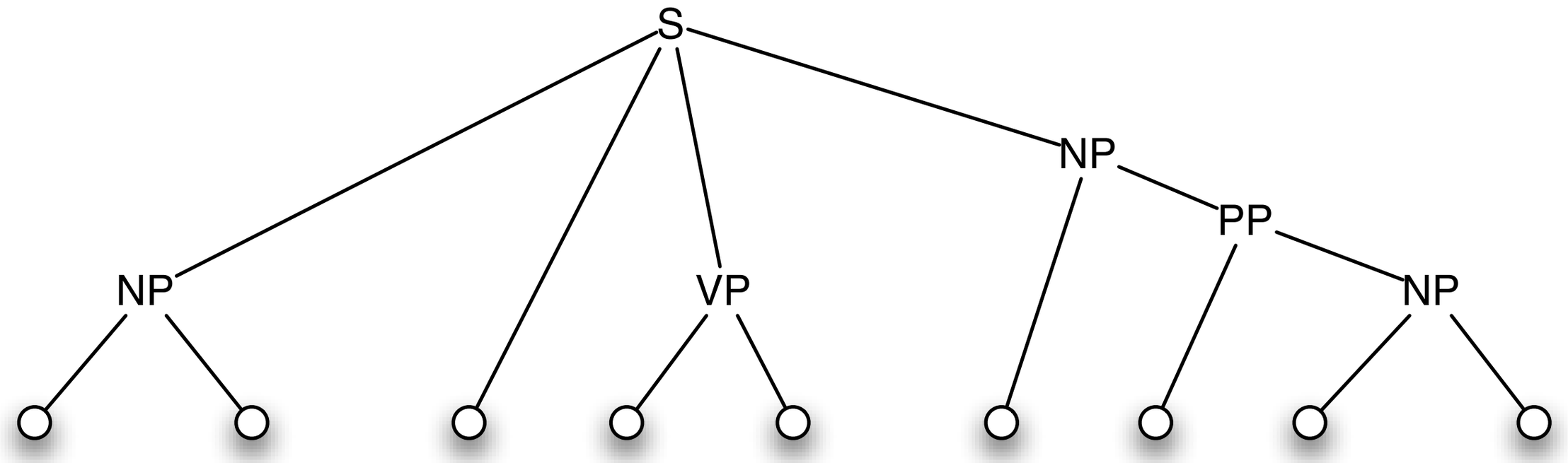
План

- Грамматика естественного языка
- Формальные грамматики
  - –Контекстно-свободные грамматики
  - –Грамматики зависимостей
  - –Категориальные грамматики
- Синтаксический разбор
- Группировка (Фрагментирование)

Грамматика составляющих

- именная группа (группа существительного, noun phrase, NP)
- группа прилагательного (adjectival phrase, ADJP)
- наречная группа (adverbial phrase, ADVP)
- предложная группа (prepositional phrase, PP)
- глагольная группа (verb phrase, VP);

Пример



Эти школьники скоро будут писать диктант по русскому языку

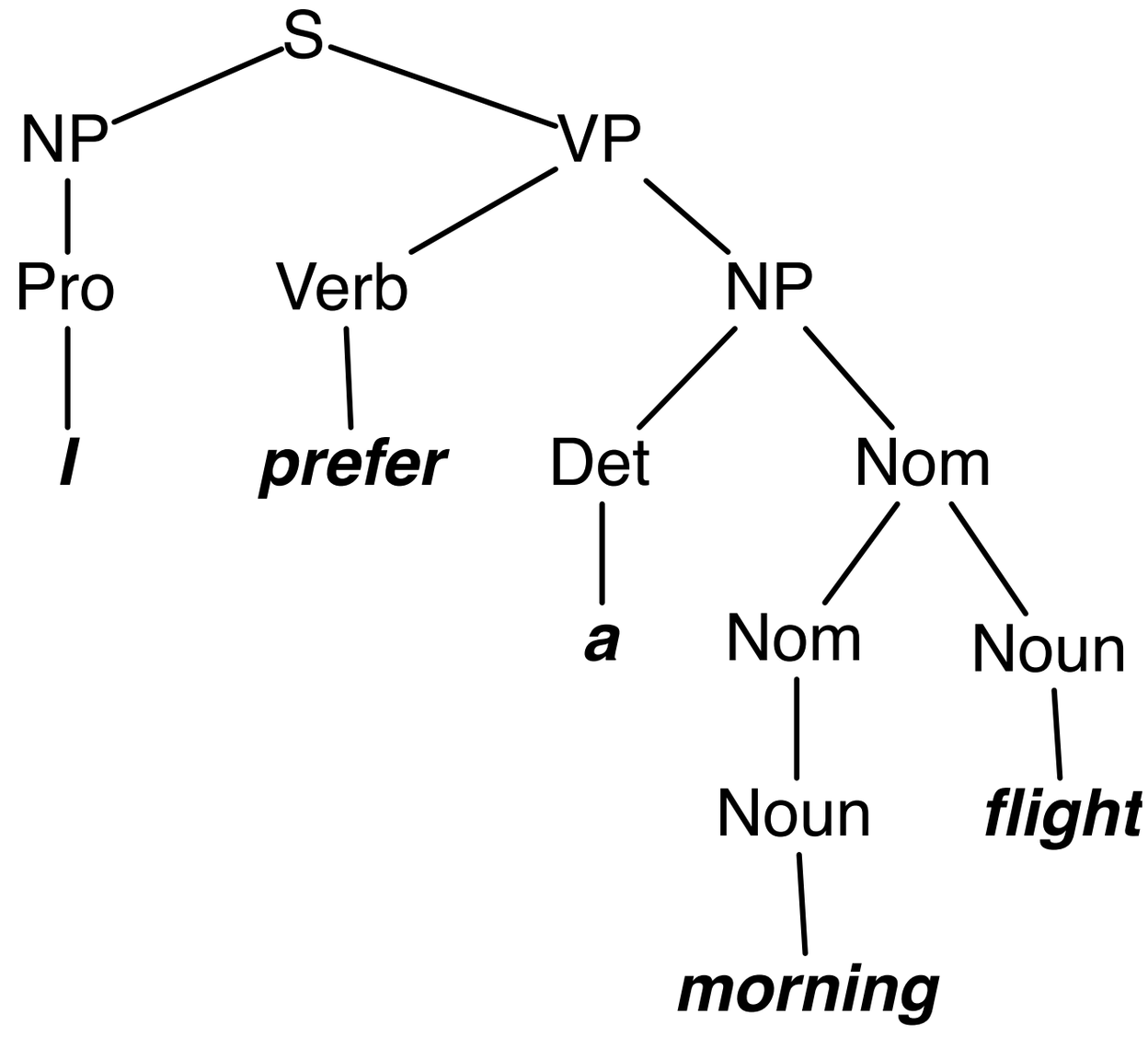
[s[np Эти школьники] скоро[vp будут писать][np диктант[pp по [np русскому языку]]]]

Контекстно свободные грамматики

Noun	→	flights | breeze | trip | morning
Verb	→	is | prefer | like | need | want| fly
Adjective	→	cheapest | non-stop | first | latest | other | direct
Pronoun	→	me | I | you | it
Proper-Noun	→	Alaska | Los Angeles | Chicago
Determiner	→	the | a | an | this | these | that
Preposition	→	from | to | on | near
Conjunction	→	and | or | but

S	→	NP VP	I + want a morning flight
NP	→	Pronoun	I
	|	Proper-Noun	Los Angeles
	|	Det Nominal	a + flight
Nominal	→	Nominal Noun	morning + flight
	|	Noun	flights
VP	→	Verb	do
	|	Verb NP	want + a flight
	|	Verb NP PP	leave + Boston + in the morning
	|	Verb PP	leaving + on Thursday
PP	→	Preposition NP	from + Los Angeles

Пример


S
NP
VP
Pro
Verb
NP
I
prefer
Det
Nom
a
Nom
Noun
Noun
flight
morning

Формальное определение

N	множество **нетерминальных символов**
Σ	множество **терминальных символов** (непересекающееся с *N*)
R	множество правил, каждое вида $A \to \beta$ где A - нетерминал, β - строка символов из множества $(\Sigma \cup N)*$
S	символ начала

Согласование

- Пример
  - –по русскому языку
  - –русский язык
- **Проблема:** Увеличение количества правил
- **Решение:** Введение параметров для нетерминальных символов
  - –см. Jurafsky, Martin глава 15

Откуда взять грамматику?

- Написать вручную
- Вывод грамматики по банку деревьев
  - Penn Treebank Project

```
( (S (NP-SBJ (NP Pierre Vinken)
             ,
             (ADJP (NP 61 years)
                                 old)
             ,)
     (VP will
        (VP join
           (NP the board)
           (PP-CLR as
                                    (NP a nonexecutive director))
                       (NP-TMP Nov. 29)))
     .))
( (S (NP-SBJ Mr. Vinken)
     (VP is
        (NP-PRD (NP chairman)
                                 (PP of
                                    (NP (NP Elsevier N.V.)
                                                   ,
                                                   (NP the Dutch publishing group)))))
     .))
```

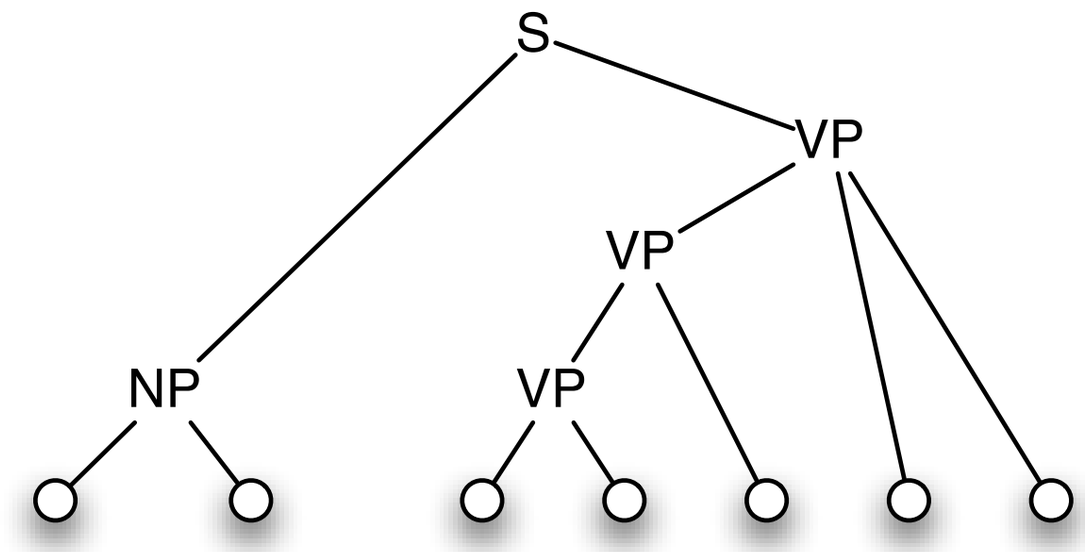
Эквивалентность грамматик

- Эквивалентность
  - –сильная (язык + деревья разбора)
  - –слабая (только язык)
- Нормальная форма грамматики (Хомского)
  - – $A \rightarrow BC$
  - – $A \rightarrow a$
- Всегда существует преобразование в нормальную форму (слабая эквивалентность)

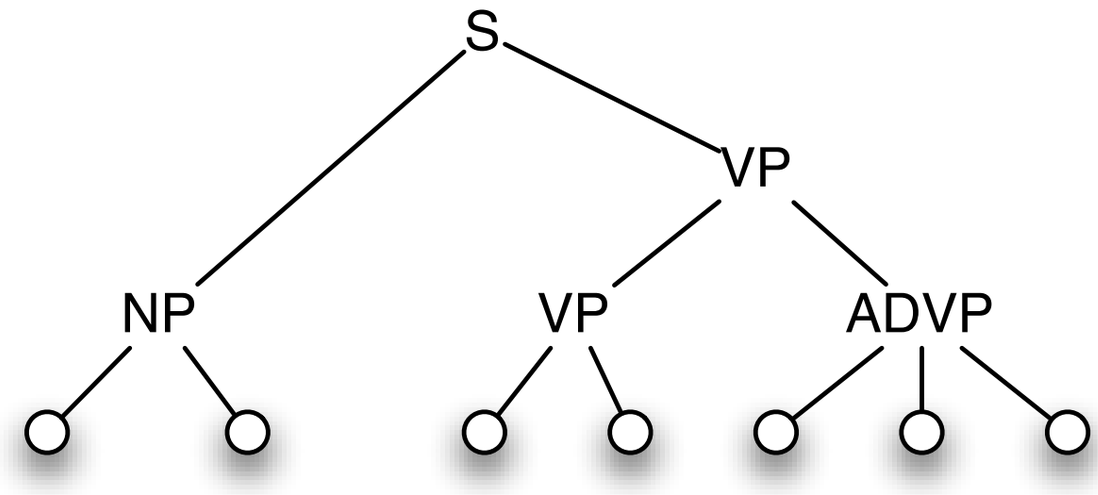
Контекстно-свободные грамматики и регулярные языки

- Контекстно-свободные грамматики являются обобщением регулярных грамматик
- Центральная вставка $A \to \alpha A \beta$
- Пример:
  - The luggage arrived.
  - The luggage that the passengers checked arrived.
  - The luggage that the passengers that the storm delayed checked arrived.

Синтаксическая многозначность



Народ Беларуси будет жить плохо, но недолго (А.Г. Лукашенко)

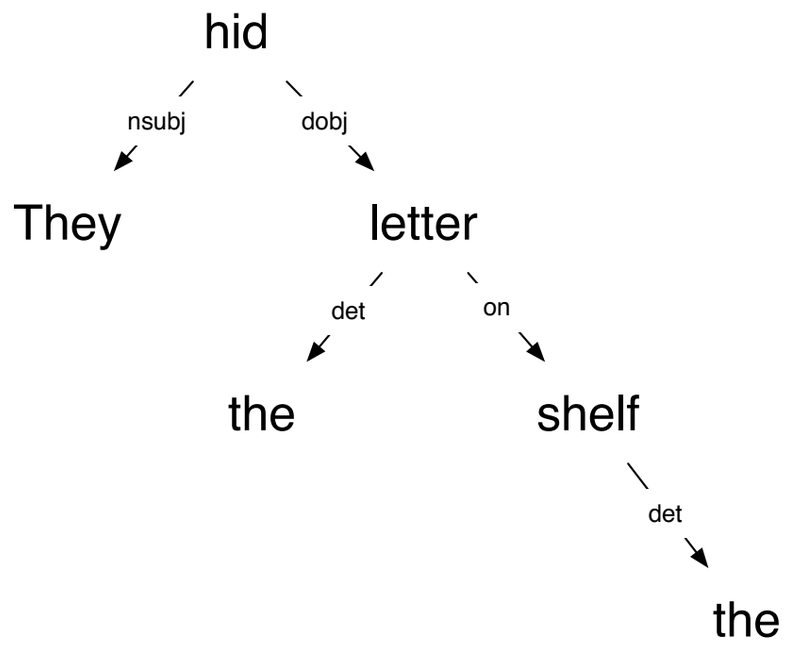


Народ Беларуси будет жить плохо, но недолго (А.Г. Лукашенко)

Другие типы грамматик

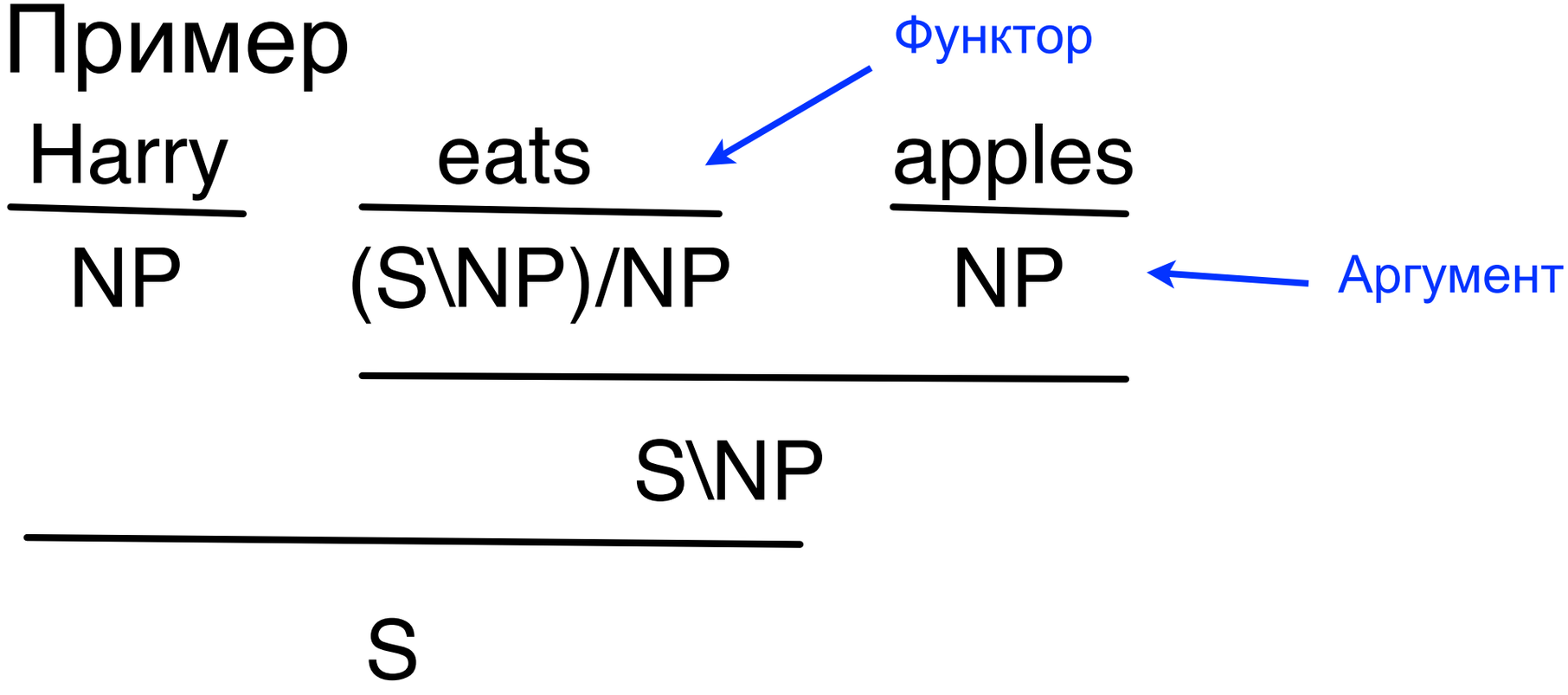
Грамматика зависимостей

- Способность предсказывать аргументы при синтаксическом разборе
- Хорошо отражают специфику языков с произвольным порядком слов
- Может быть автоматически получена из дерева разбора на составляющие



Категориальная грамматика

- Категории фраз:
  - –Состоят из функторов и аргуменов
  - –X/Y - функция из Y в X. Аргумент присоединяется к Y справа, чтобы получилось X
  - –X\Y - ... слева ...
- Пример



Обработка текстов

Синтаксический разбор

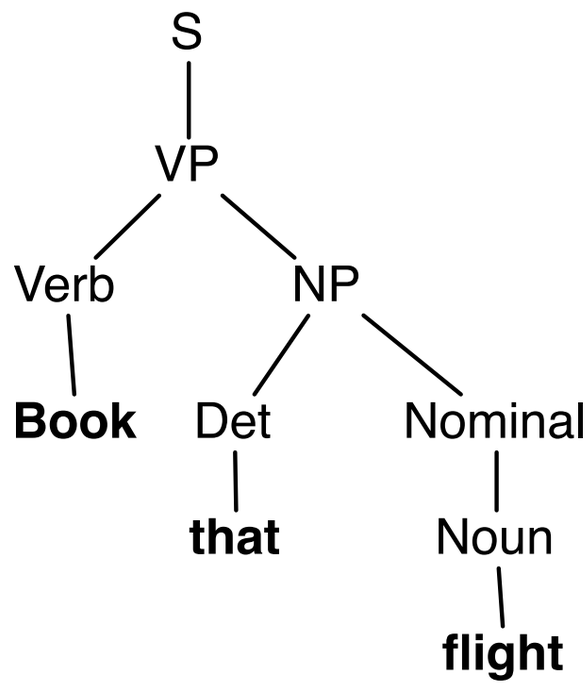
Синтаксический разбор

- Рассматриваемые алгоритмы
  - –Метод рекурсивного спуска (top-down parsing)
  - –Восходящий анализ (bottom-up parsing)
  - –Алгоритм Кока-Янгера-Касами (CKY Parsing)
- Не рассматриваемые, но часто используемые алгоритмы
  - –Алгоритм Эрли (Earley parser)
  - –Chart parser
  - –http://en.wikipedia.org/wiki/Category:Parsing_algorithms

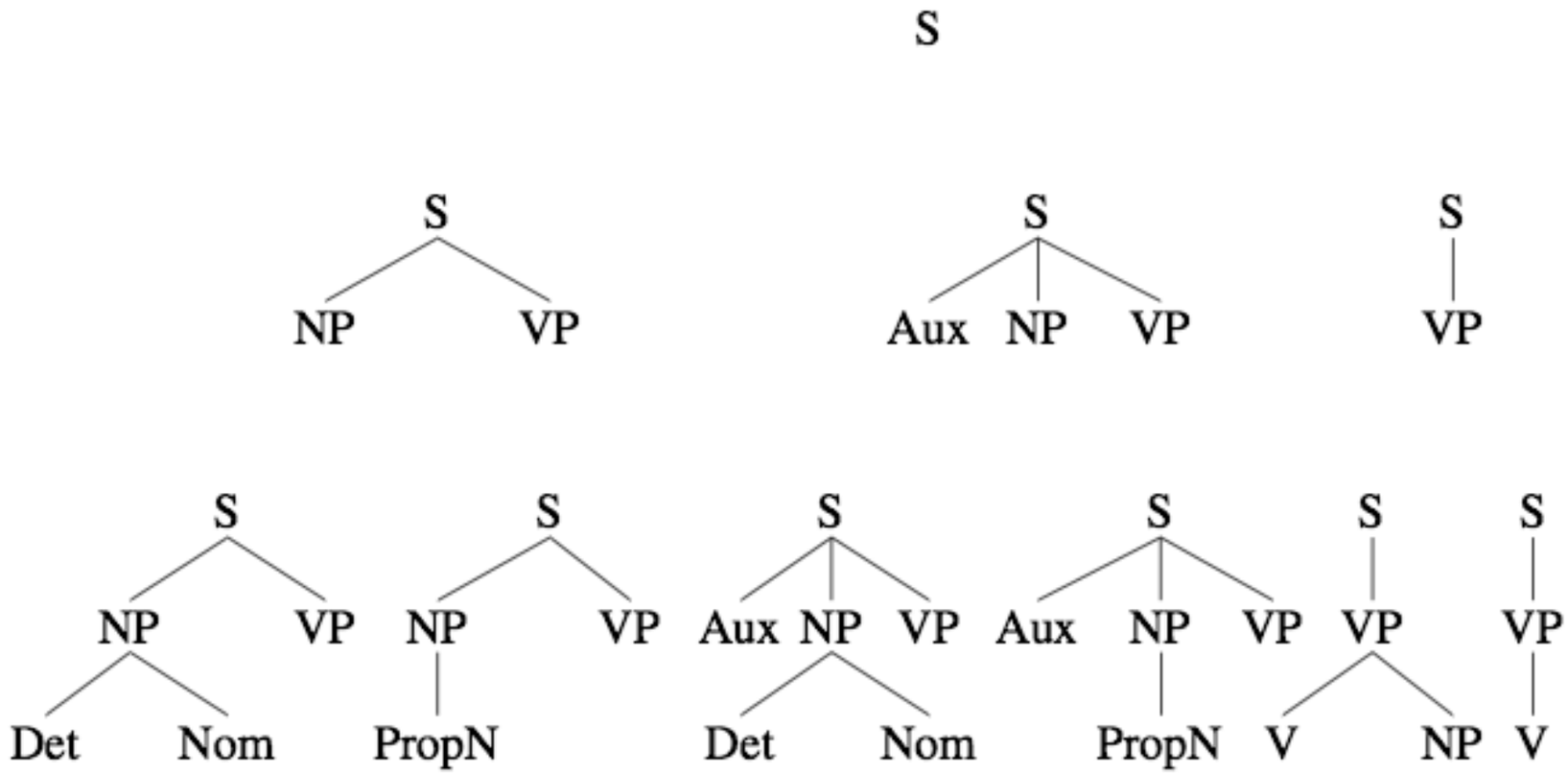
Пример

S → NP VP
S → Aux NP VP
S → VP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → Verb NP PP
VP → Verb PP
VP → VP PP
PP → Preposition NP

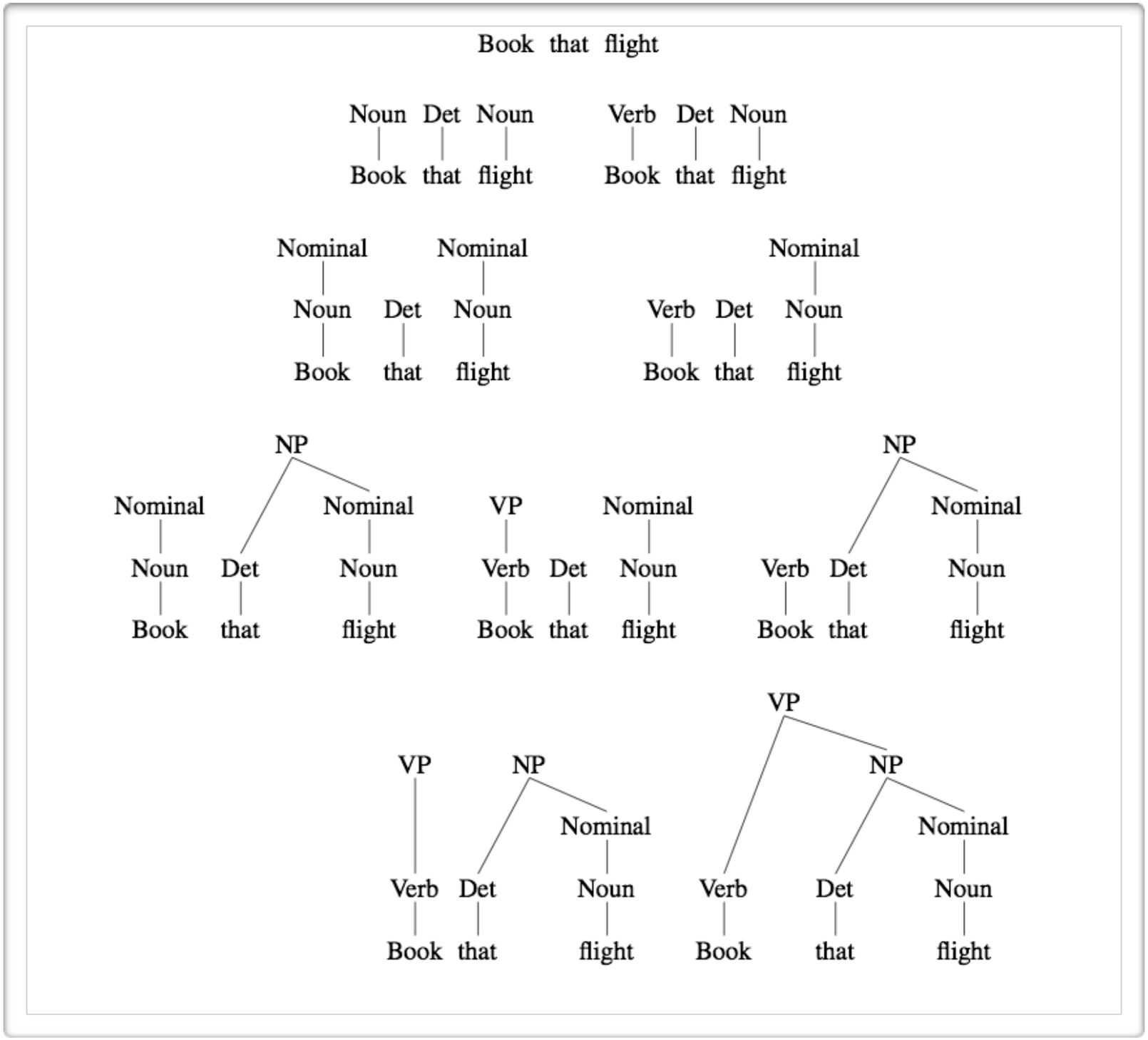
Det → that | this | a
Noun → book | flight | meal | money
Verb → book | include | prefer
Pronoun → I | she | me
Proper-Noun → Houston | TWA
Aux → does
Preposition → from | to | on | near | through



Метод рекурсивного спуска



Восходящий анализ



Алгоритм CKY

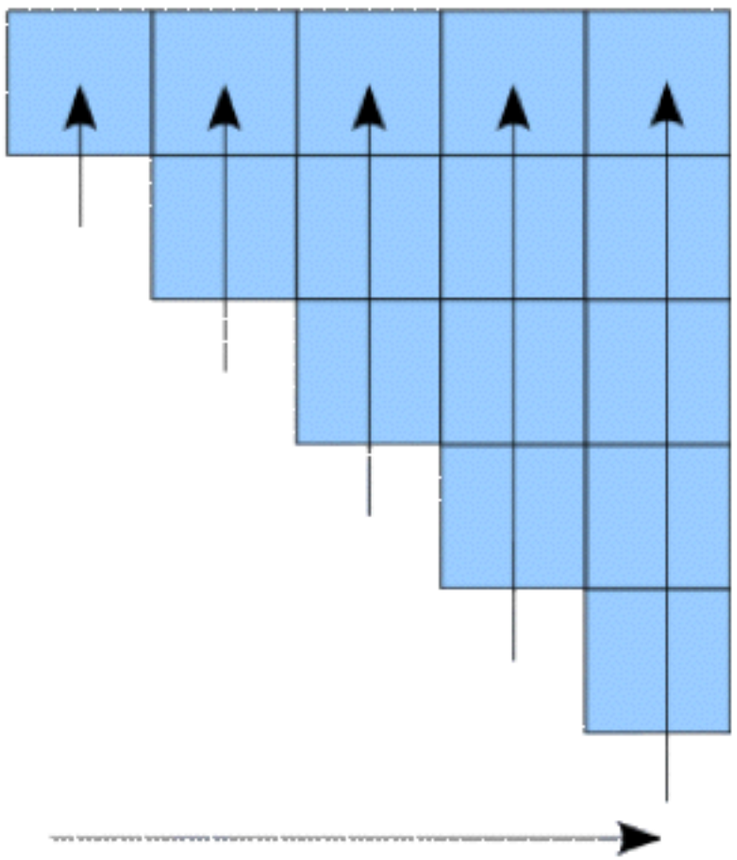
- Шаг 0. Преобразовать грамматику к нормальной форме
- Алгоритм (динамическое программирование)

function CKY-PARSE(*words, grammar*) **returns** *table*

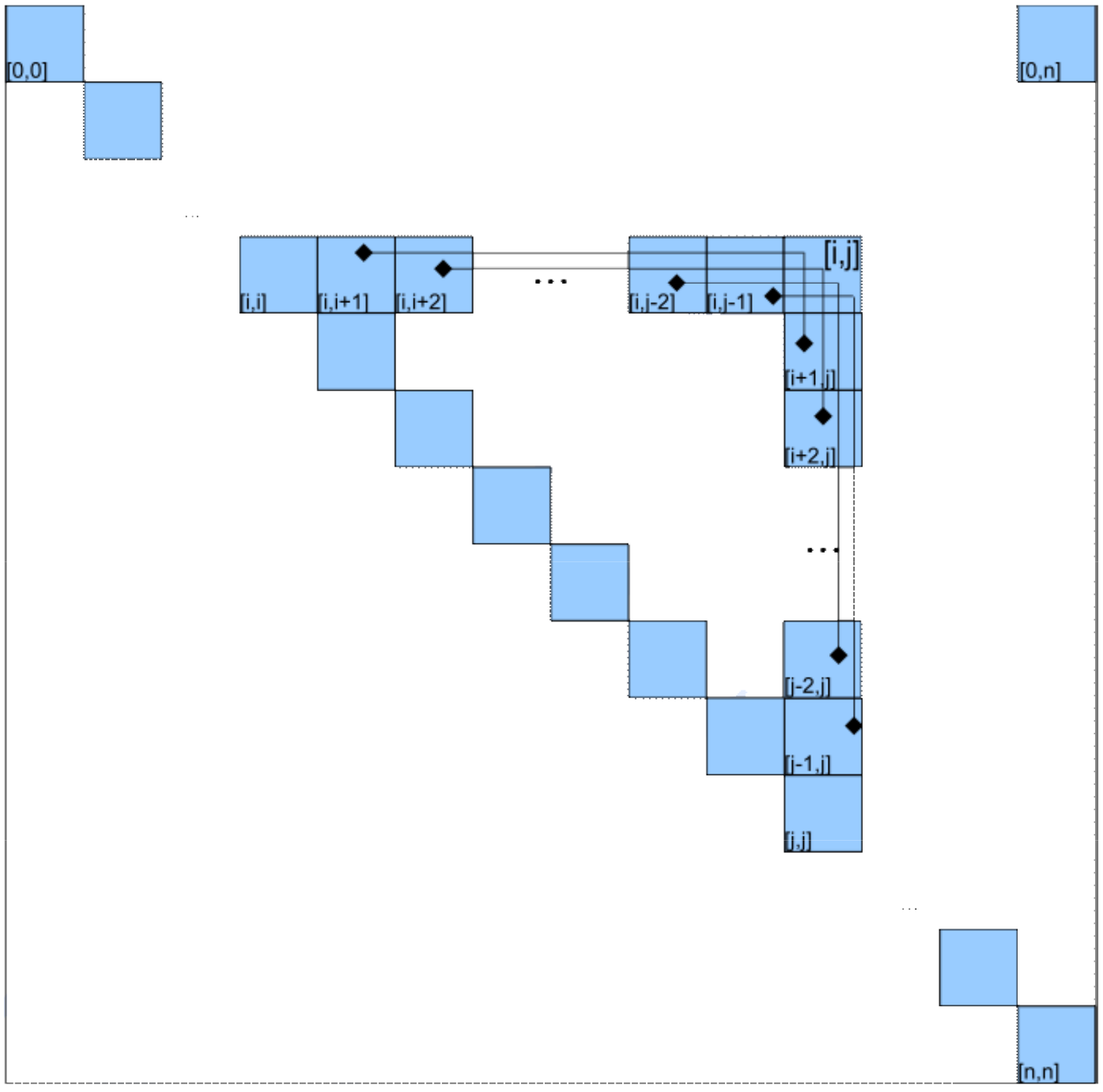
for $j \leftarrow$ **from** 1 **to** LENGTH(*words*) **do**
  $table[j-1, j] \leftarrow \{A \mid A \rightarrow words[j] \in grammar\}$
  for $i \leftarrow$ **from** $j-2$ **downto** 0 **do**
    for $k \leftarrow i+1$ **to** $j-1$ **do**
      $table[i,j] \leftarrow table[i,j] \cup$
        $\{A \mid A \rightarrow BC \in grammar,$
          $B \in table[i,k],$
          $C \in table[k,j]\}$

Распознавание

Book	the	flight	through	Houston
SP, VP, Nominal, Verb,Noun [0,1]	[0,2]	S, VP, X2 [0,3]	[0,4]	S1, VP1, S2, VP2, S3 [0,5]
	Det [1,2]	NP [1,3]	[1,4]	NP [1,5]
		Nominal, Noun [2,3]	[2,4]	Nominal [2,5]
			Prep [3,4]	PP [3,5]
				NP, Proper-Noun [0,1]

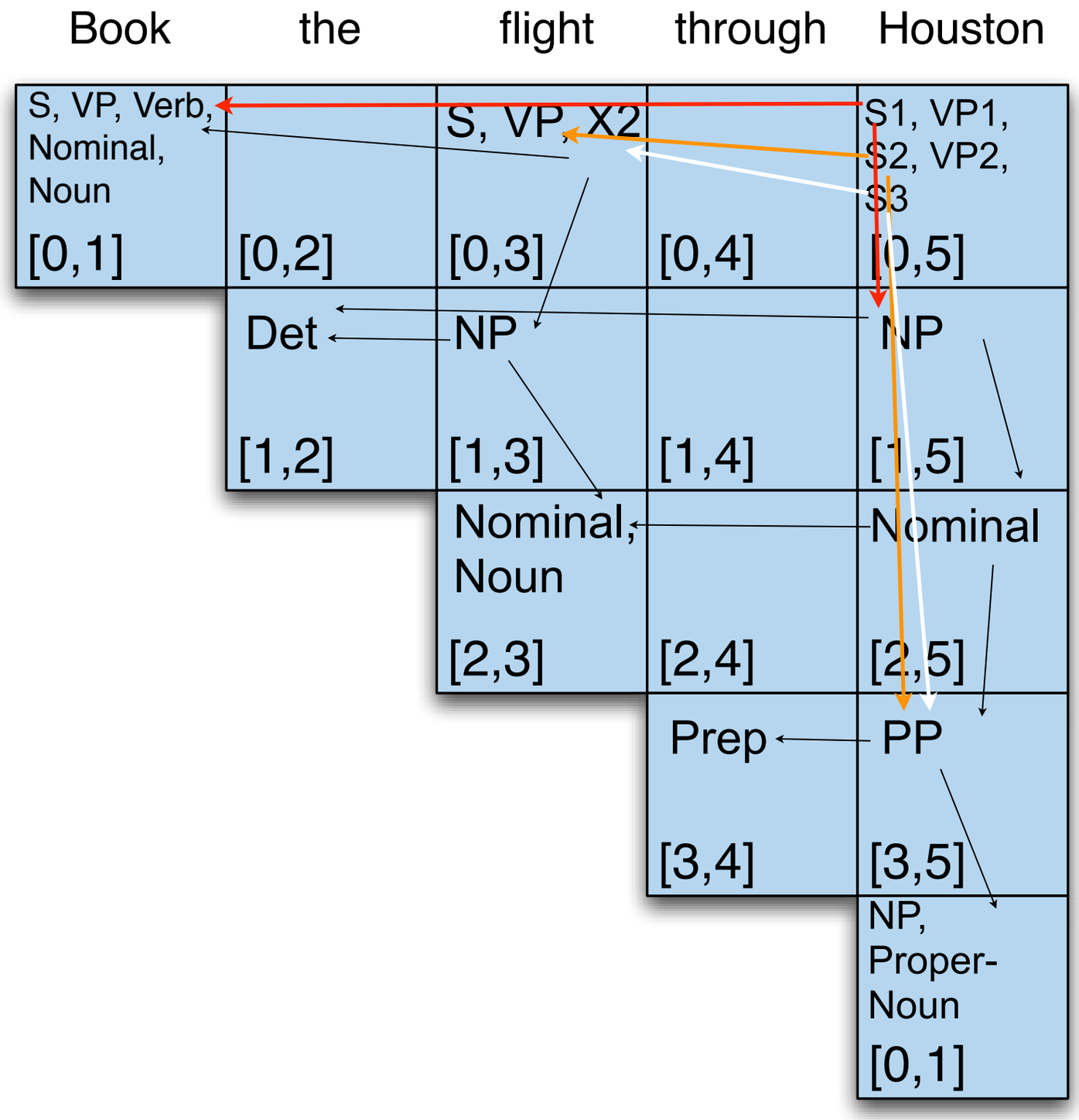


Запоминание путей



Синтаксический разбор

S → NP VP
S → X1 VP
X1 → Aux NP
S → VP
S → X2 PP
NP → Pronoun
NP → Proper-Noun
NP → Det Nominal
Nominal → Noun
Nominal → Nominal Noun
Nominal → Nominal PP
VP → Verb
VP → Verb NP
VP → X2 PP
X2 → Verb NP
VP → Verb PP
VP → VP PP
PP → Preposition NP



Группировка

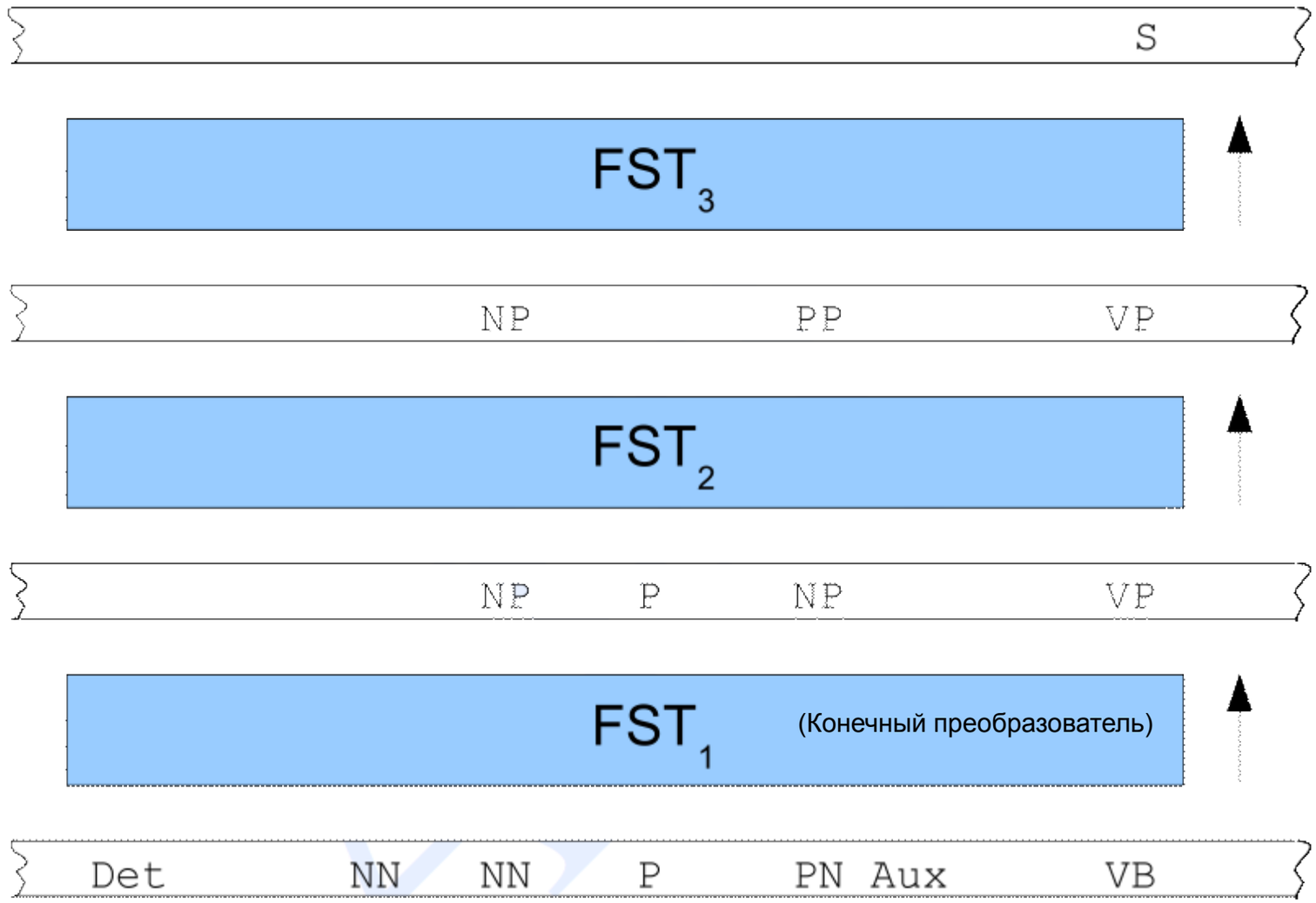
- Partial parsing, Shallow parsing
- Chunking, фрагментирование
  - –[NP The morning flight][PP from][NP Denver][VP has arrived]
  - –[NP The morning flight] from [NP Denver] has arrived

Группировка на основе правил

S → PP* NP PP* VP PP*

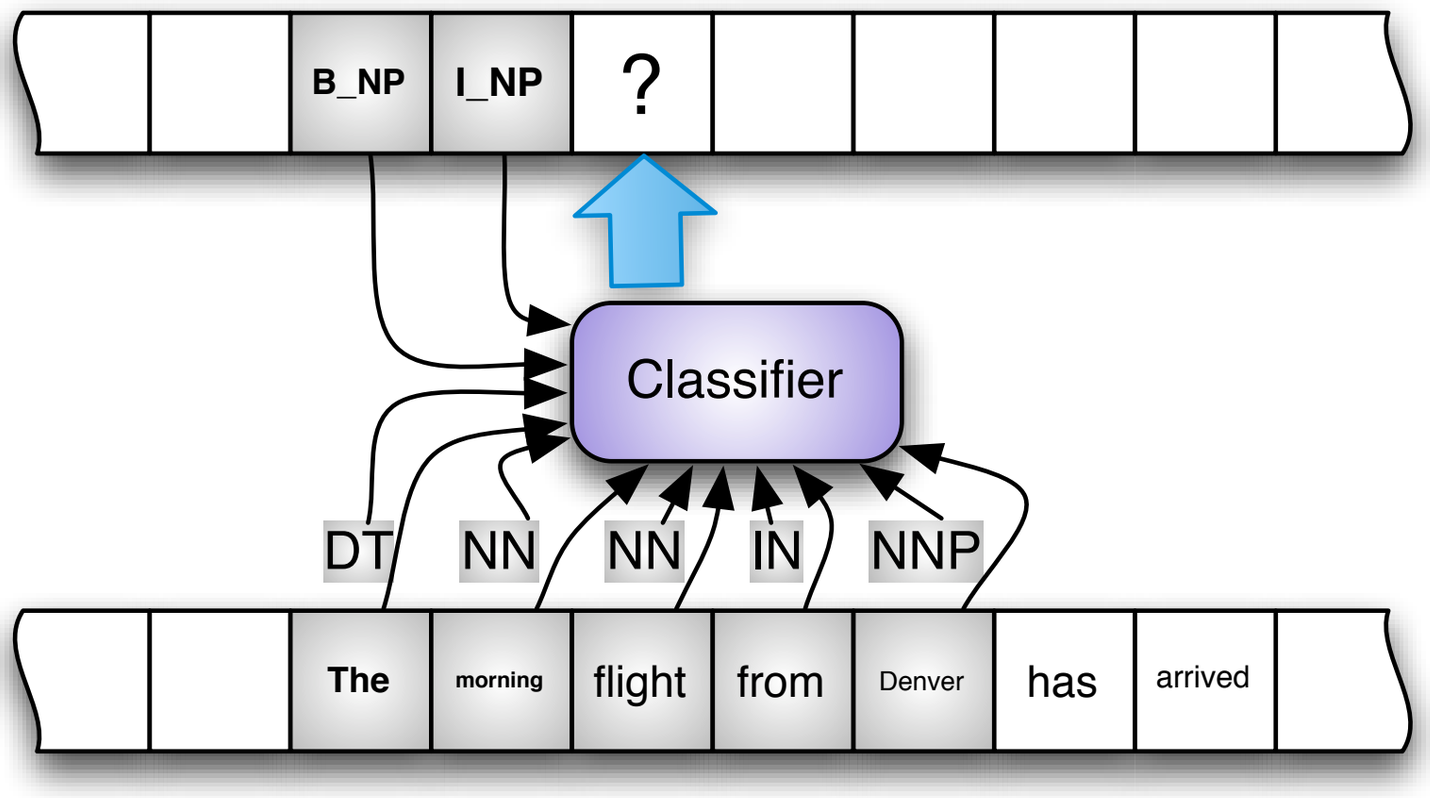
PP → IN NP

NP → (Det) Noun* Noun
NP → Proper-Noun
VP → VB
VP → Aux VB



Группировка на основе машинного обучения

- Классы BIO (begin, inside, outside)
- Тренировочное множество - Treebank



Признаки: *The, DT, B_NP, morning, NN, I_NP, flight, NN, from, IN, Denver, NNP*

Заключение

- Изучены
  - –некоторые особенности грамматик естественного языка
  - –наиболее используемые типы формальных грамматик
  - –некоторые алгоритмы синтаксического разбора
  - –подходы к группировке

Следующая лекция

- Статистические методы синтаксического анализа

Основы обработки текстов

Лекция 7

Статистические методы синтаксического анализа

Мотивация

- СКС-грамматики позволяют определить лучшее дерево разбора (т.е. устранить многозначность)
- Более точное моделирование языка, по сравнению с n-граммами
  - распознавание речи
  - машинный перевод
  - извлечение информации
  - ...
  - выделение ключевых слов

План

- Стохастические контекстно-свободные грамматики (СКС)
  - –разрешение синтаксической многозначности
  - –моделирование языка
- Вероятностная версия алгоритма CKY
- Обучение СКС
- Проблемы СКС
  - –разделение и слияние нетерминалов
  - –СКС с поддержкой лексики
    - алгоритм Коллинза
- Методы оценки

Стохастические контекстно-свободные грамматики

N	множество **нетерминальных символов**
Σ	множество **терминальных символов** (непересекающееся с *N*)
R	множество правил, каждое вида $A \to \beta[p]$ где A - нетерминал, β - строка символов из множества $(\Sigma \cup N)*$ p - вероятность правила $P(\beta \mid A)$, $\sum_{\beta} P(A \to \beta) = 1$
S	символ начала

Пример

Грамматика	Вероятность		
S → NP VP	0.8		
S → Aux NP VP	0.1	+	1.0
S → VP	0.1		
NP → Pronoun	0.2		
NP → Proper-Noun	0.2	+	1.0
NP → Det Nominal	0.6		
Nominal → Noun	0.3		
Nominal → Nominal Noun	0.2	+	1.0
Nominal → Nominal PP	0.5		
VP → Verb	0.2		
VP → Verb NP	0.5	+	1.0
VP → VP PP	0.3		
PP → Prep NP	1.0		

Лексикон

Det → the | a | that | this
0.6 0.2 0.1 0.1

Noun → book | flight | meal | money
0.1 0.5 0.2 0.2

Verb → book | include | prefer
0.5 0.2 0.3

Pronoun → I | he | she | me
0.5 0.1 0.1 0.3

Proper-Noun → Houston | NWA
0.8 0.2

Aux → does
1.0

Prep → from | to | on | near | through
0.25 0.25 0.1 0.2 0.2

Разрешение многозначности

- Вероятность разбора

$$P(T,S) = \prod_{i=1}^{n} P(RHS_i|LHS_i)$$

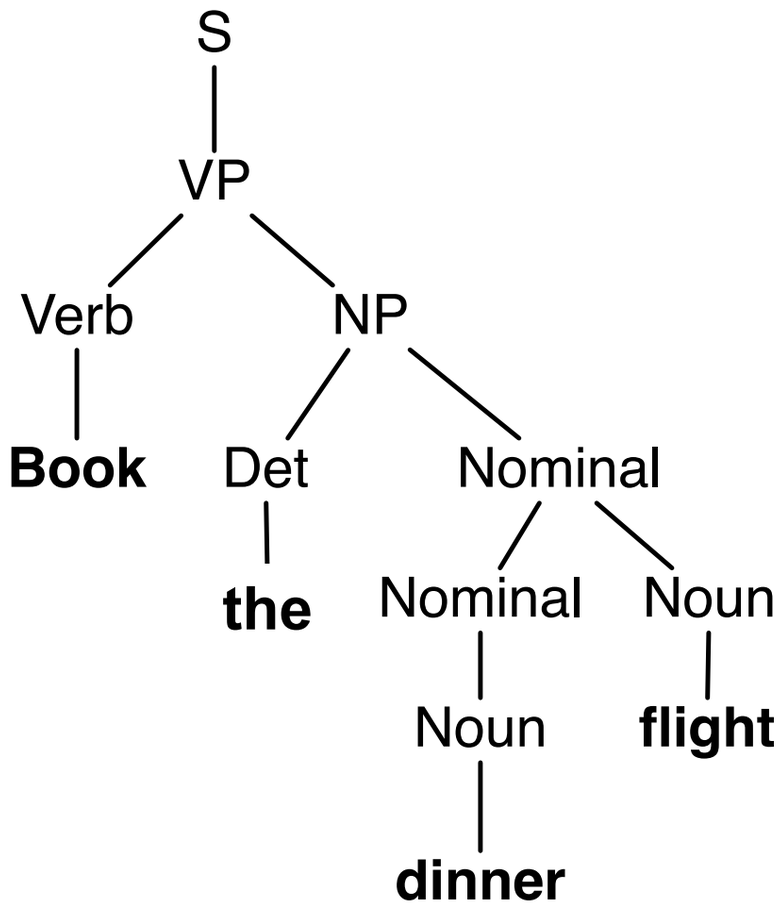
- Вероятность $P(T,S) = P(T)P(S|T) = P(T)$
- Выбор наиболее вероятного дерева разбора $\hat{T}(S) = \arg\max_T P(T|S)$

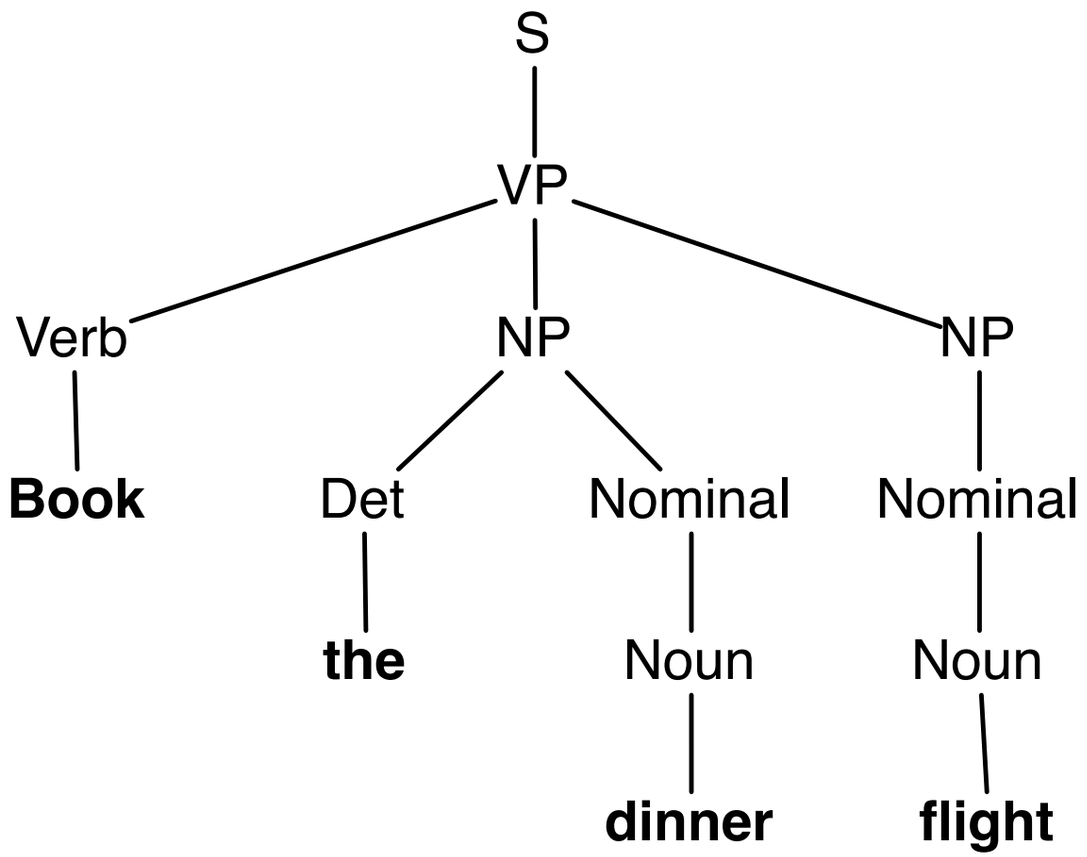
$$\hat{T}(S) = \arg\max_T \frac{P(T,S)}{P(S)}$$

$$\hat{T}(S) = \arg\max_T P(T,S)$$

$$\hat{T}(S) = \arg\max_T P(T)$$

Разрешение многозначности





P(T-left) = .05*.20*.20*.20*75*.30*.60*.10*.40=**2.2*10 -6**

P(T-right) = .05*.10*.20*.15*.75*.75*.30*.60*.10*.40=**6.1*10 -7**

Моделирование языка

$$P(S) = \sum_T P(T, S) = \sum_T P(T)$$

- Вариант 1:
  - –Этап 1: с помощью n-граммной модели получить m лучших предложений
  - –Этап 2: выбрать наиболее вероятное предложение на основе грамматики
- Вариант 2:
  - –Модифицировать парсер для предсказания следующего слова (Xu et. al 2002)

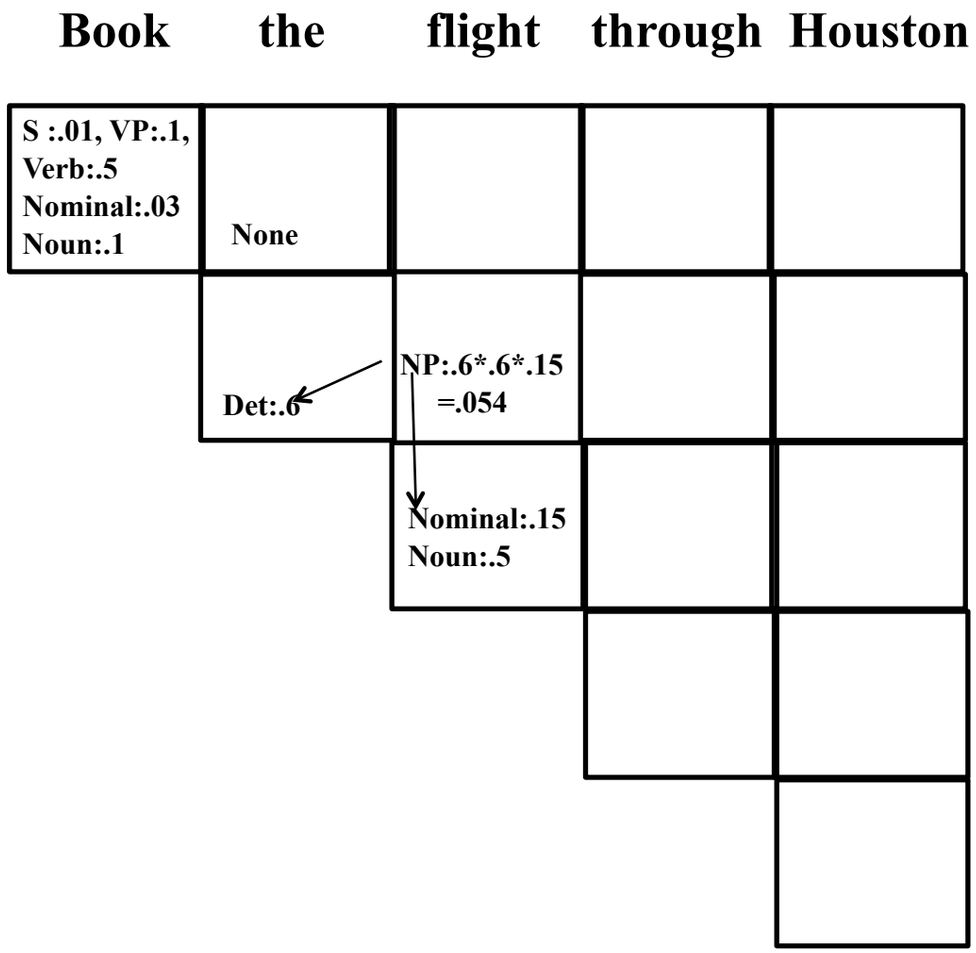
Вероятностная версия алгоритма CKY

- Добавляем в каждую ячейку вероятность нетерминального символа
- Ячейка [i,j] должна содержать наиболее вероятный вывод, покрывающий с i+1 по j слова и содержать их вероятность.
- При трансформации грамматики к нормальной форме необходимо сохранить вероятности правил

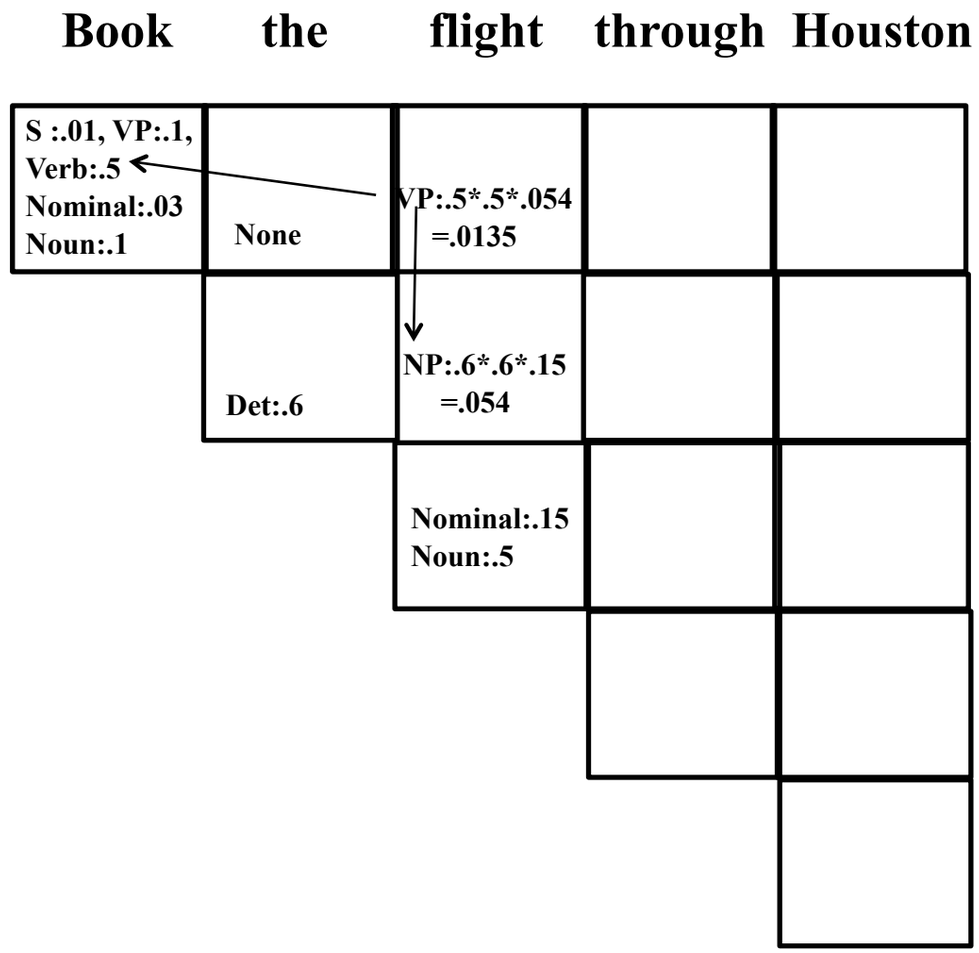
Преобразование грамматики

Оригинальная грамматика		Грамматика в нормальной форме Хомского	
S → NP VP	0.8	**S → NP VP**	0.8
S → Aux NP VP	0.1	**S → X1 VP**	0.1
		X1 → Aux NP	1.0
S → VP	0.1	**S → book | include | prefer**	
		0.01 0.004 0.006	
		S → Verb NP	0.05
		S → VP PP	0.03
NP → Pronoun	0.2	**NP → I | he | she | me**	
		0.1 0.02 0.02 0.06	
NP → Proper-Noun	0.2	**NP → Houston | NWA**	
		0.16 .04	
NP → Det Nominal	0.6	**NP → Det Nominal**	0.6
Nominal → Noun	0.3	**Nominal → book | flight | meal | money**	
		0.03 0.15 0.06 0.06	
Nominal → Nominal Noun	0.2	**Nominal → Nominal Noun**	0.2
Nominal → Nominal PP	0.5	**Nominal → Nominal PP**	0.5
VP → Verb	0.2	**VP → book | include | prefer**	
		0.1 0.04 0.06	
VP → Verb NP	0.5	**VP → Verb NP**	0.5
VP → VP PP	0.3	**VP → VP PP**	0.3
PP → Prep NP	1.0	**PP → Prep NP**	1.0

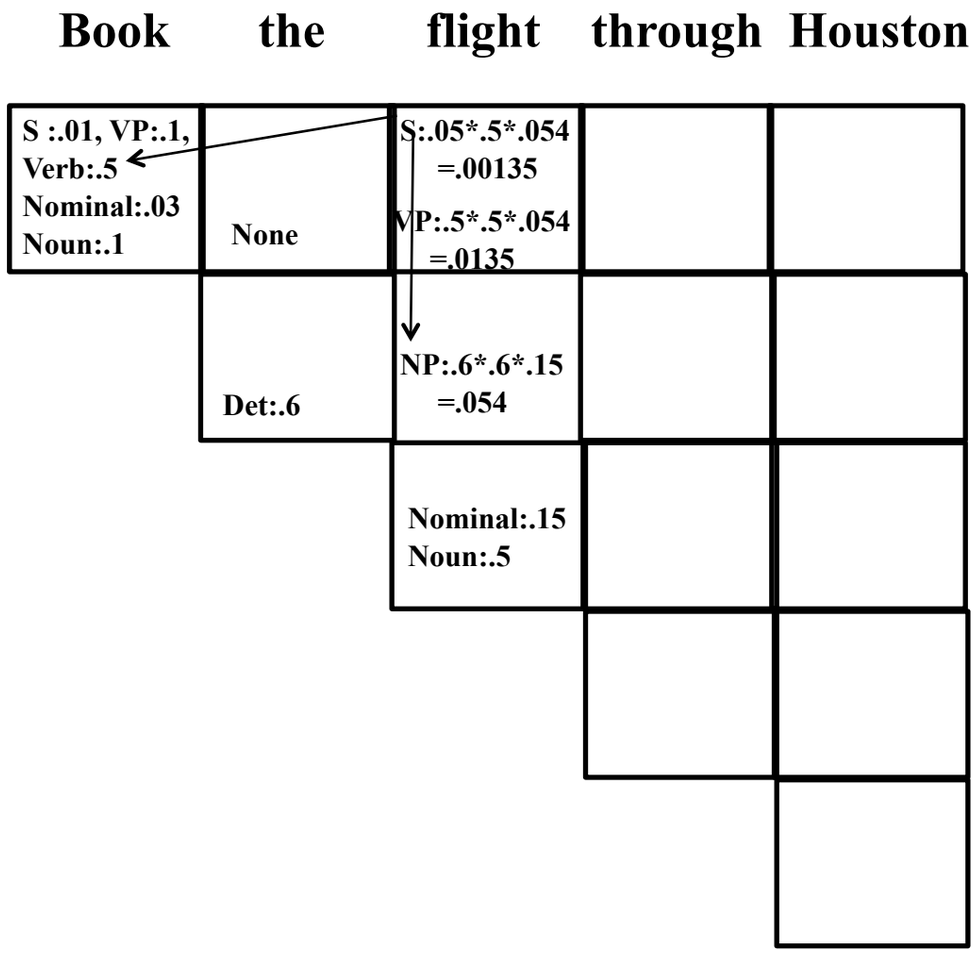
Вероятностная версия алгоритма CKY



Вероятностная версия алгоритма CKY



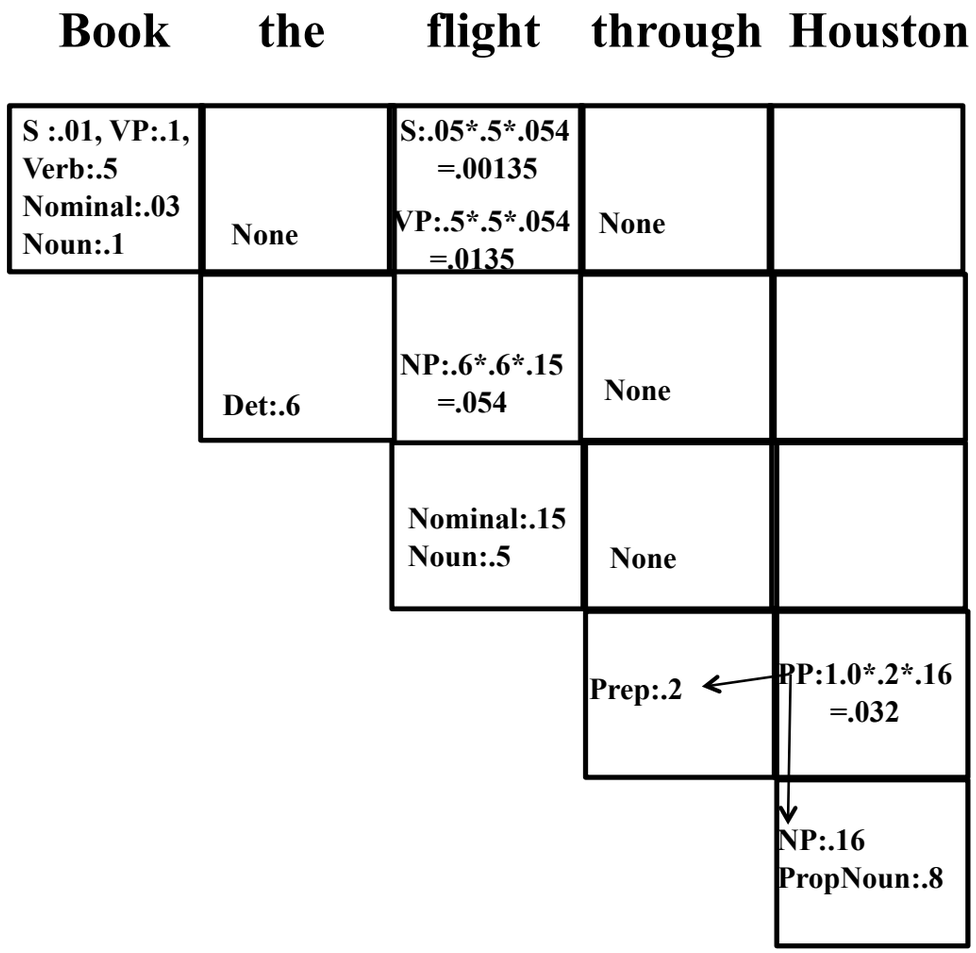
Вероятностная версия алгоритма CKY



Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None	S:.05*.5*.054 =.00135 VP:.5*.5*.054 =.0135	None	
	Det:.6	NP:.6*.6*.15 =.054	None	
		Nominal:.15 Noun:.5	None	
			Prep:.2	

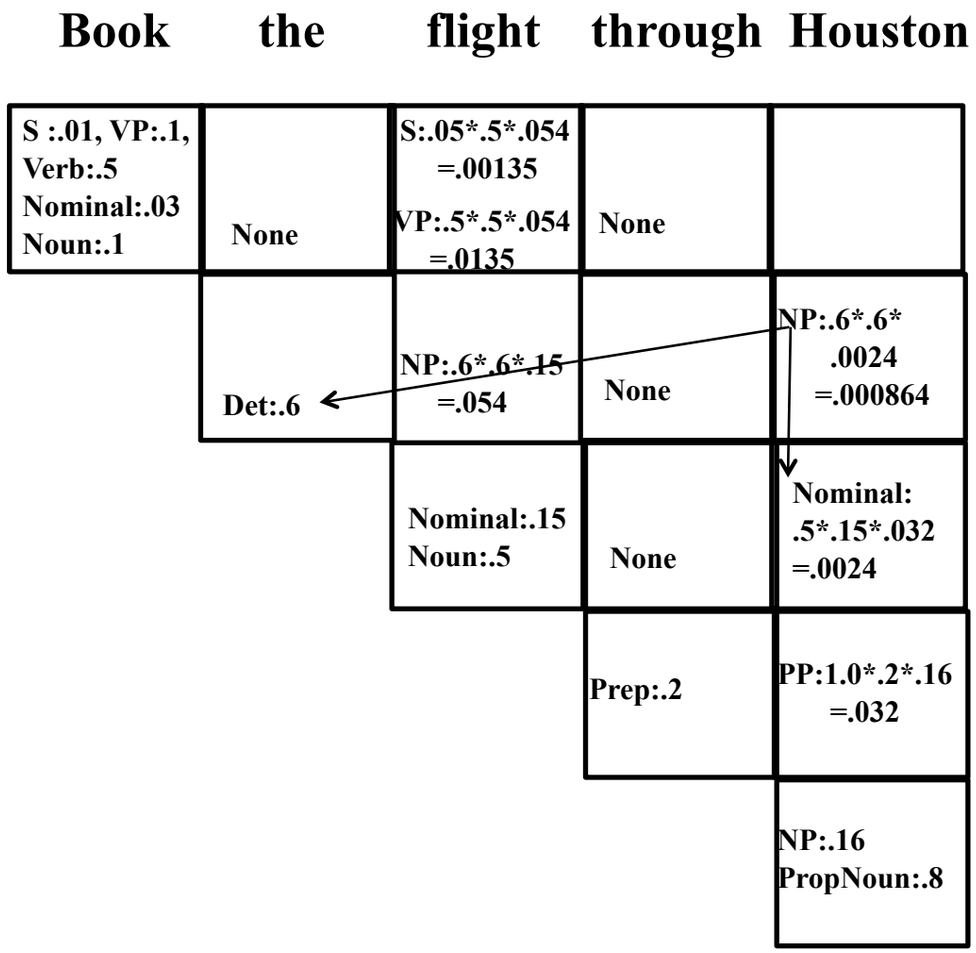
Вероятностная версия алгоритма CKY



Вероятностная версия алгоритма CKY

Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1	None	S:.05*.5*.054 =.00135 VP:.5*.5*.054 =.0135	None	
	Det:.6	NP:.6*.6*.15 =.054	None	
		Nominal:.15 Noun:.5	None	Nominal: .5*.15*.032 =.0024
			Prep:.2	PP:1.0*.2*.16 =.032
				NP:.16 PropNoun:.8

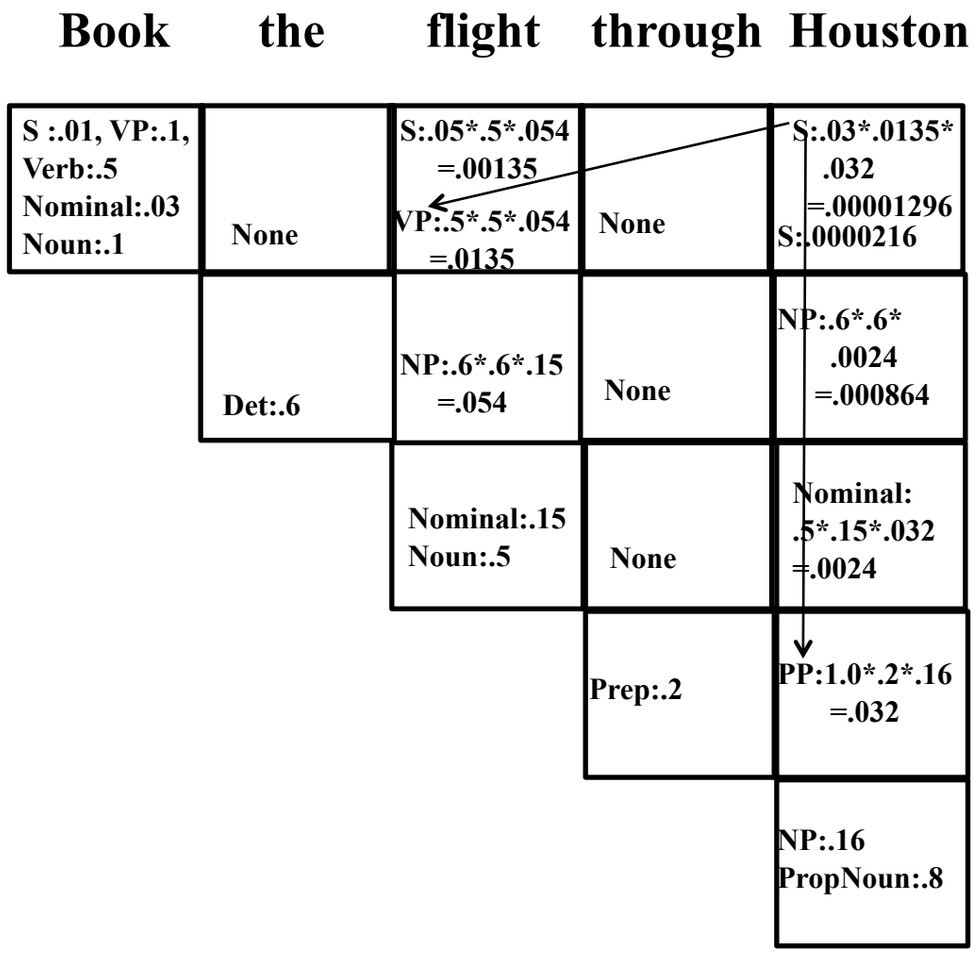
Вероятностная версия алгоритма CKY



Вероятностная версия алгоритма CKY

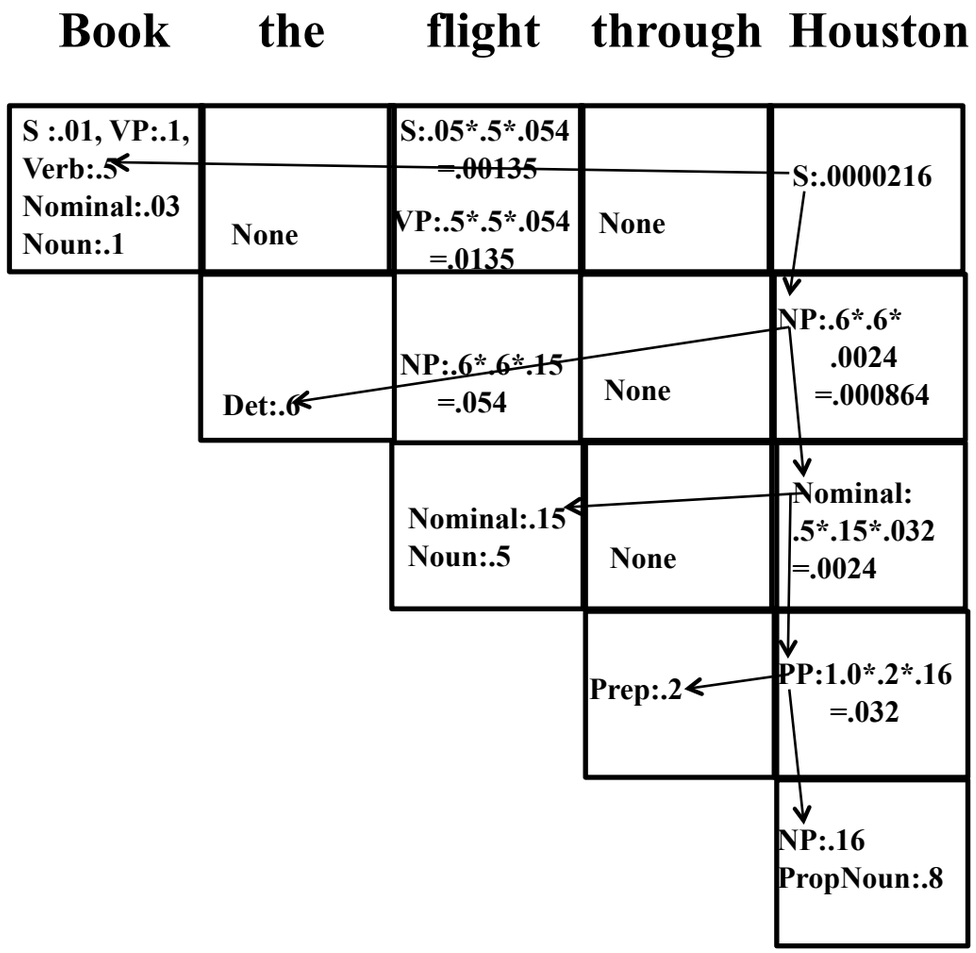
Book	the	flight	through	Houston
S :.01, VP:.1, Verb:.5 Nominal:.03 Noun:.1		S:.05*.5*.054 =.00135		S:.05*.5* .000864 =.0000216
	None	VP:.5*.5*.054 =.0135	None	
	Det:.6	NP:.6*.6*.15 =.054	None	NP:.6*.6* .0024 =.000864
		Nominal:.15 Noun:.5	None	Nominal: .5*.15*.032 =.0024
			Prep:.2	PP:1.0*.2*.16 =.032
				NP:.16 PropNoun:.8

Вероятностная версия алгоритма CKY



Вероятностная версия алгоритма CKY

Выбираем наиболее вероятное дерево разбора



Обучение СКС

- Вычисление вероятности на основе банка деревьев

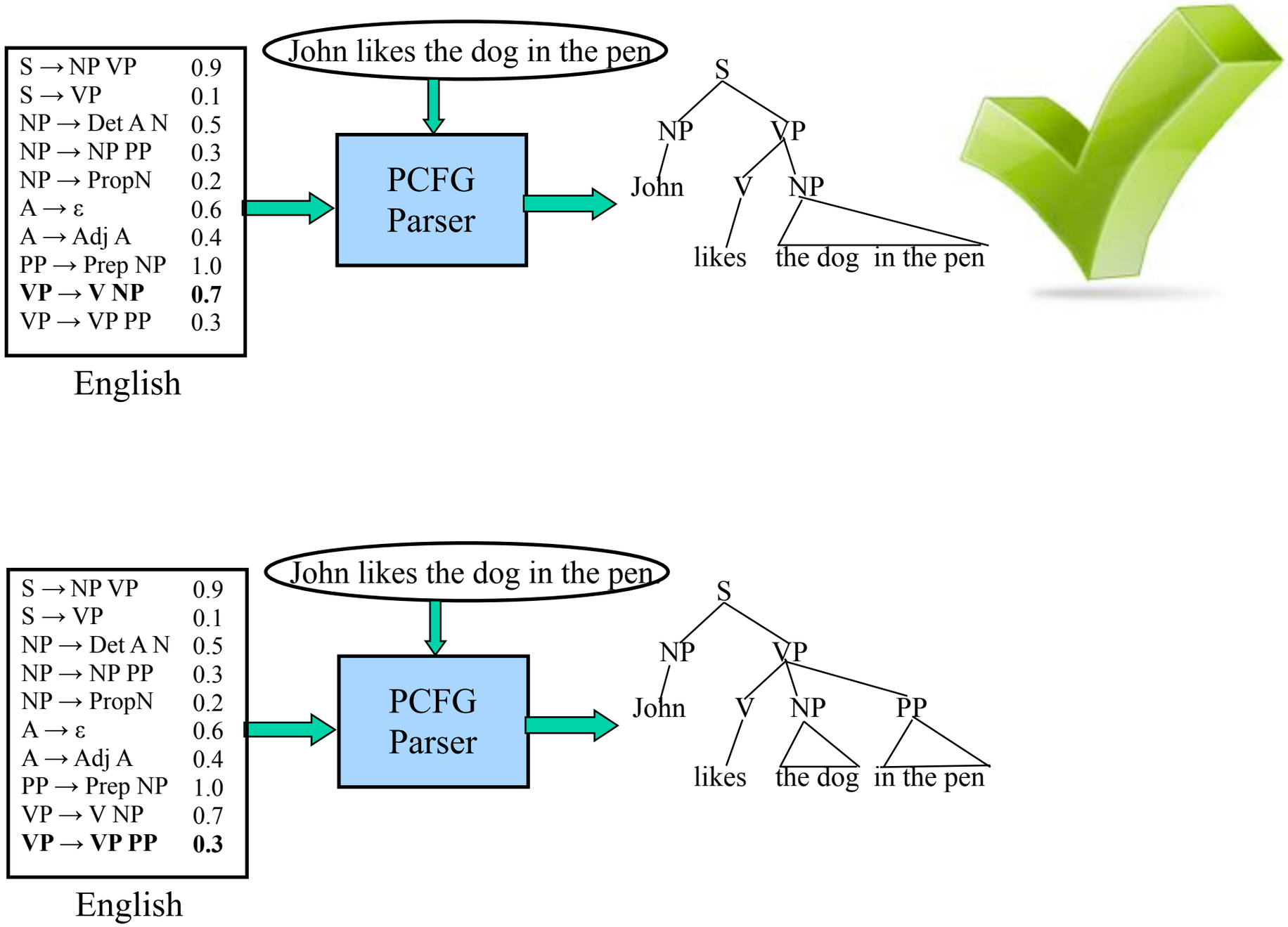
$$P(\alpha \rightarrow \beta|\alpha) = \frac{Count(\alpha \rightarrow \beta)}{\sum_{\gamma} Count(\alpha \rightarrow \beta)} = \frac{Count(\alpha \rightarrow \beta)}{Count(\alpha)}$$

- Вывод без тренировочного множества (EM)
  - –На основе множества предложений построить множество наиболее вероятных синтаксических разборов
  - –Обновить значения вероятностей на осонве полученных данных
  - –(Manning and Schutze 1999)

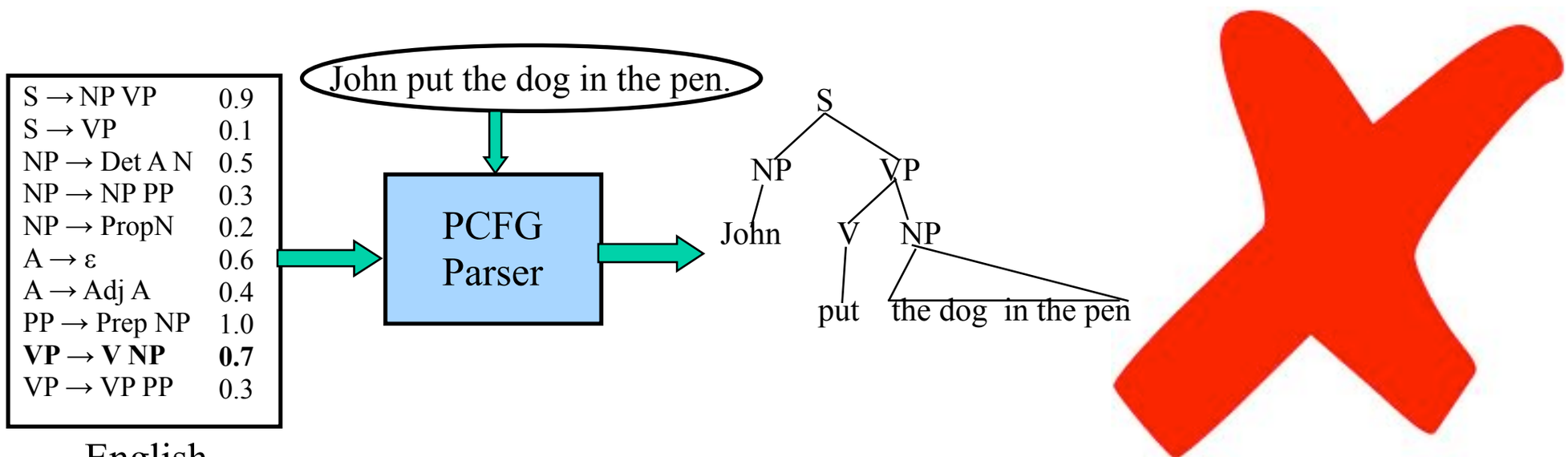
Проблемы СКС

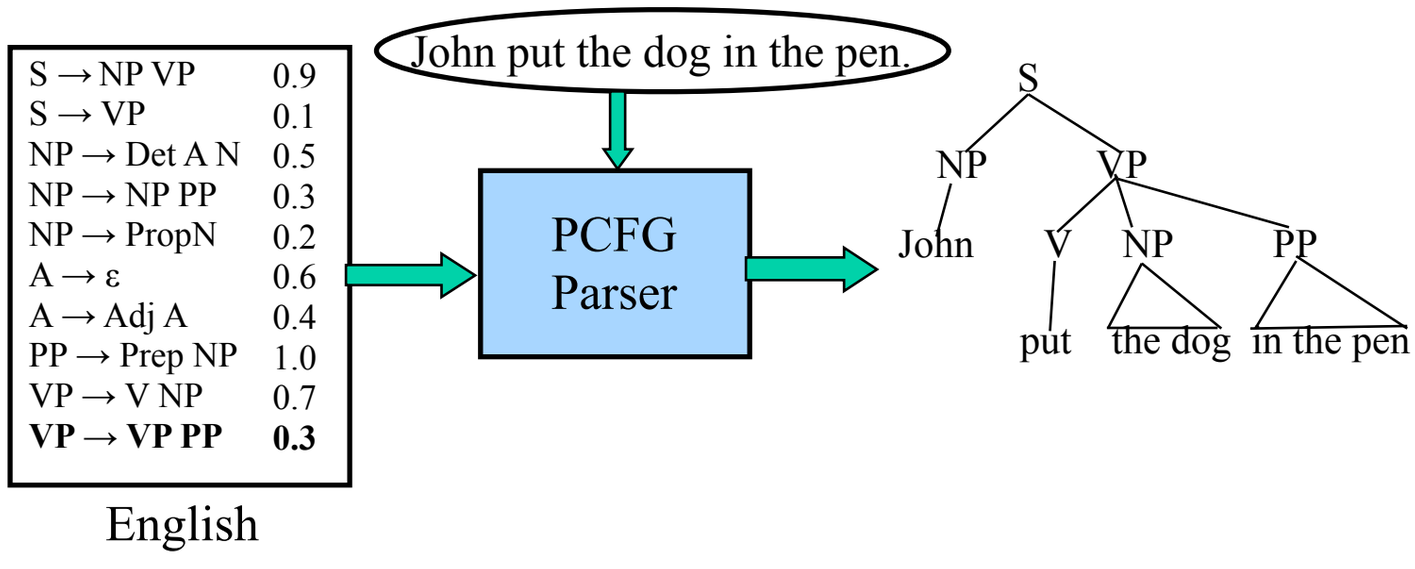
- Предположение о независимости правил, не позволяет хорошо моделировать структурные зависимости в дереве разбора
- СКС не могут моделировать синтаксические факты о конкретных словах

Пример


S → NP VP 0.9
S → VP 0.1
NP → Det A N 0.5
NP → NP PP 0.3
NP → PropN 0.2
A → ε 0.6
A → Adj A 0.4
PP → Prep NP 1.0
VP → V NP 0.7
VP → VP PP 0.3
English
John likes the dog in the pen
PCFG Parser
S
NP
VP
John
V
NP
likes
the dog
in the pen
S → NP VP 0.9
S → VP 0.1
NP → Det A N 0.5
NP → NP PP 0.3
NP → PropN 0.2
A → ε 0.6
A → Adj A 0.4
PP → Prep NP 1.0
VP → V NP 0.7
VP → VP PP 0.3
English
John likes the dog in the pen
PCFG Parser
S
NP
VP
John
V
NP
PP
likes
the dog
in the pen

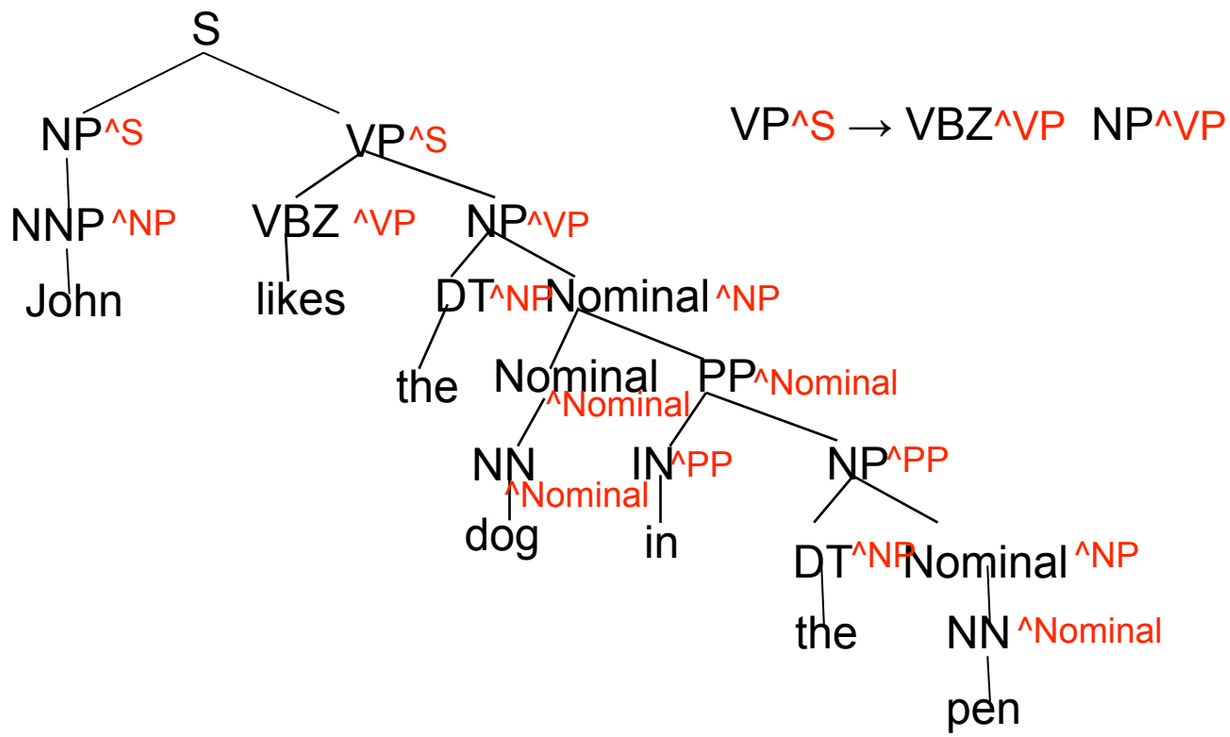
Пример


S → NP VP 0.9
S → VP 0.1
NP → Det A N 0.5
NP → NP PP 0.3
NP → PropN 0.2
A → ε 0.6
A → Adj A 0.4
PP → Prep NP 1.0
VP → V NP 0.7
VP → VP PP 0.3
English
John put the dog in the pen.
PCFG Parser
S
NP
VP
John
V
NP
put
the dog
in the pen


S → NP VP 0.9
S → VP 0.1
NP → Det A N 0.5
NP → NP PP 0.3
NP → PropN 0.2
A → ε 0.6
A → Adj A 0.4
PP → Prep NP 1.0
VP → V NP 0.7
VP → VP PP 0.3
English
John put the dog in the pen.
PCFG Parser
S
NP
VP
John
V
NP
PP
put
the dog
in the pen

Решение проблемы зависимостей

- Для добавления контекстуальной информации нетерминалы можно разделить на несколько, используя родительские узлы в дереве разбора (parent annotation)

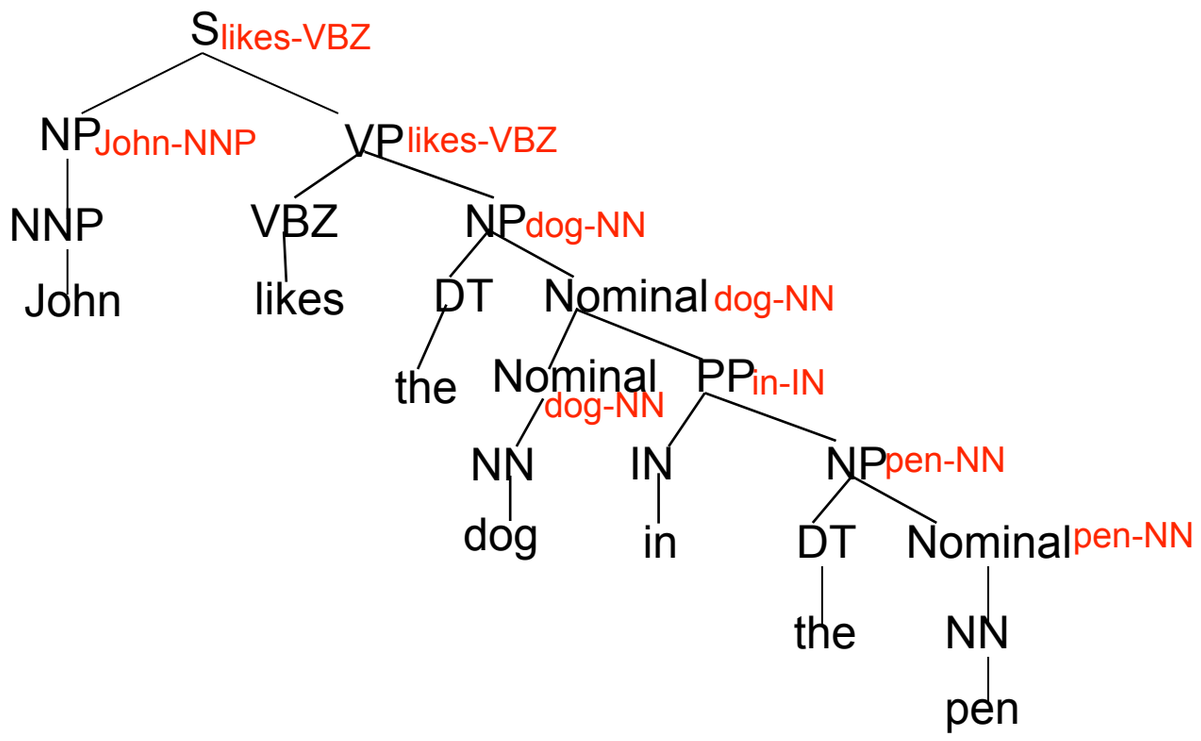


Разделение и слияние

- Разделение нетерминальных символов сильно увеличивает грамматику
- Лучше разделять нетерминалы, только если это приведет к улучшению точности
- Также можно объединять некоторые нетерминальные символы, чтобы достичь болшей точности
- Метод: эвристический поиск наилучшей комбинации разделений и слияний которая будет максимизировать правдоподобие банка деревьев

СКС с поддержкой лексики

- Расширение правил
  - VP → VP PP
  - VP (put) → VP (put) PP (in)
  - VP (put,VDB) → VP (put, VDB) PP (in,IN)



Оценка вероятности

- Точная оценка невозможна, так как не существует достаточного количества деревьев
- Необходимо сделать еще предположения (о независимости), которые помогут оценить эти вероятности
- Алгоритмы (Collins, 1999), (Charniak, 1997)

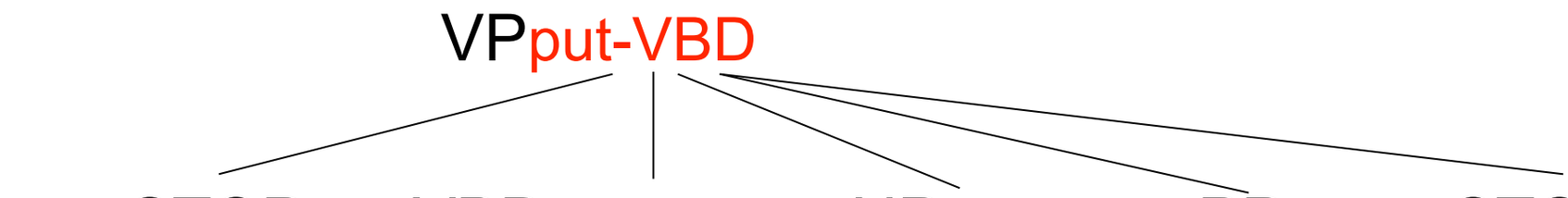
Алгоритм Коллинза

- Использует простую производящую модель
- LHS $\rightarrow L_n L_{n-1} \ldots L_1 H\ R_1 \ldots R_{m-1} R_m$
  - H-вершина группы
  - L - символы слева
  - R - символы справа
  - По краям символы STOP
- Вероятности левых и правых символов зависят только от вершины группы и нетерминала в левой части правила

Пример

VPput-VBD → VBDput-VBD NPdog-NN PPin-IN

VPput-VBD → STOP VBDput-VBD NPdog-NN PPin-IN STOP
L_1 H R_1 R_2 R_3



P(VPput-VBD → VBDput-VBD NPdog-NN PPin-IN) =

$= P_H$(VBDput-VBD| VPput-VBD) *
$* P_L$(STOP | VBDput-VBD, VPput-VBD) *
$* P_R$(NPdog-NN | VBDput-VBD, VPput-VBD) *
$* P_R$(PPin-IN | VBDput-VBD, VPput-VBD) *
$* P_R$(STOP | VBDput-VBD, VPput-VBD)

Оценка вероятностей

- Вероятности можно оценить на основе банка деревьев

$$P_R(\text{PP}_{\text{in-IN}} \mid \text{VP}_{\text{put-VBD}}) = \frac{\text{Count}(\text{PP}_{\text{in-IN}} \text{ справа от вершины в правиле для VP}_{\text{put-VBD}})}{\text{Count}(\text{символы справа от вершины в правиле для VP}_{\text{put-VBD}})}$$

- Сглаживание можно осуществлять через откат или линейную интерполяцию

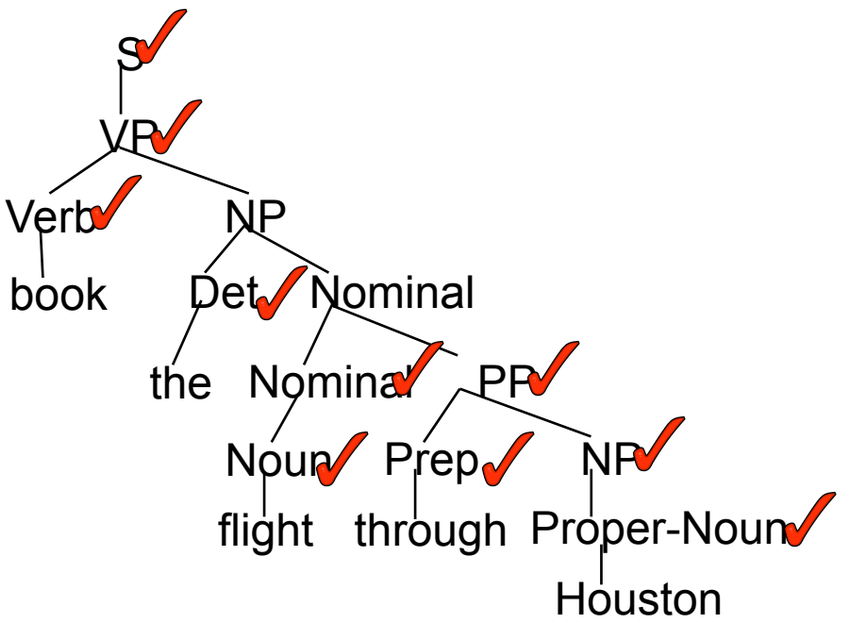
$$\begin{aligned} smP_R(\text{PP}_{\text{in-IN}} \mid \text{VP}_{\text{put-VBD}}) = {} & \lambda_1 P_R(\text{PP}_{\text{in-IN}} \mid \text{VP}_{\text{put-VBD}}) \\ & + (1-\lambda_1)\,(\lambda_2 P_R(\text{PP}_{\text{in-IN}} \mid \text{VP}_{\text{VBD}}) + \\ & (1-\lambda_2) P_R(\text{PP}_{\text{in-IN}} \mid \text{VP})) \end{aligned}$$

Оценка качества алгоритма

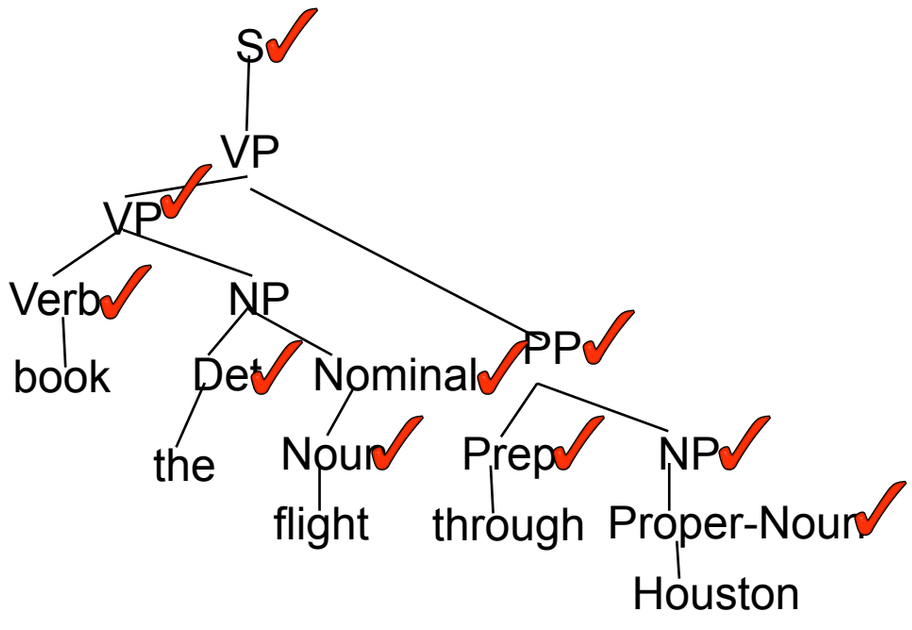
- Метрика PARSEVAL: пусть P - дерево разбора, созданное алгоритмом, T - дерево разбора, созданное экспертами
  - Точность = (# правильных компонент в P) / (# компонент в T)
  - Полнота = (# правильных компонент в P) / (# компонент в P)
  - F-мера = 2PR / (P + R)
- Современные алгоритмы показывают точность и полноту более 90%

Оценка качества алгоритма

T - дерево, размеченное вручную



P - вычисленное дерево



компонент: 12

компонент: 12

правильных компонент: 10

Точность = 10/12= 83.3%

Полнота = 10/12= 83.3%

F_1 = 83.3%

Делают ли люди синтаксический разбор?

- Психолингвистика
- Алгоритмы синтаксического разбора могут быть использованы для предсказания времени, которое потребуется человеку для прочтения каждого слова в предложении
- Чем выше вероятность слова, тем скорость чтения больше
- Для моделирования этого эффекта требуется инкрементальный алгоритм

Предложения с временной неоднозначностью

- Garden path sentence
  - –Complex houses married students
  - –The horse raced past the barn fell
- Инкрементальные парсеры могут найти и объяснить сложность таких предложений

Сложность языка

- Является ли естественный язык регулярным?
  - –контр-пример был на прошлой лекции
- Является ли естественный язык контекстно-свободным?
  - –Диалект немецкого языка в Швейцарии содержит контекстно-зависимые конструкции вида $a^n b^m c^n d^m$
- Сложность понимания людьми
  - –чем проще конструкция, тем легче понимание смысла текста

Заключение

- Статистические модели, такие как СКС позволяют разрешать многозначность
- СКС может можно выучить на основе банка деревьев
- Учет лексики и разделение нетерминальных символов позволяет разрешить дополнительные неоднозначности
- Точность современных алгоритмов синтаксического разбора высока, но не достигает уровня экспертного разбора

Следующая лекция

- Лексическая семантика и разрешение лексической многозначности

Основы обработки текстов

Лекция 8

Лексическая семантика

Возможные взгляды на семантику

- Лексическая семантика
  - –значение индивидуальных слов
- Композиционная семантика
  - –как значения комбинируются и определяют новые значения для словосочетаний
- Дискурс или прагматика
  - –как значения комбинируются между собой и другими знаниями, чтобы задать значение текста или дискурс

План

- Основные понятия
  - слова и отношения между ними
  - словари и тезаурусы
- Вычислительная семантика
  - Разрешение лексической многозначности
  - Семантическая близость слов
  - Некоторые современные направления

Основные понятия

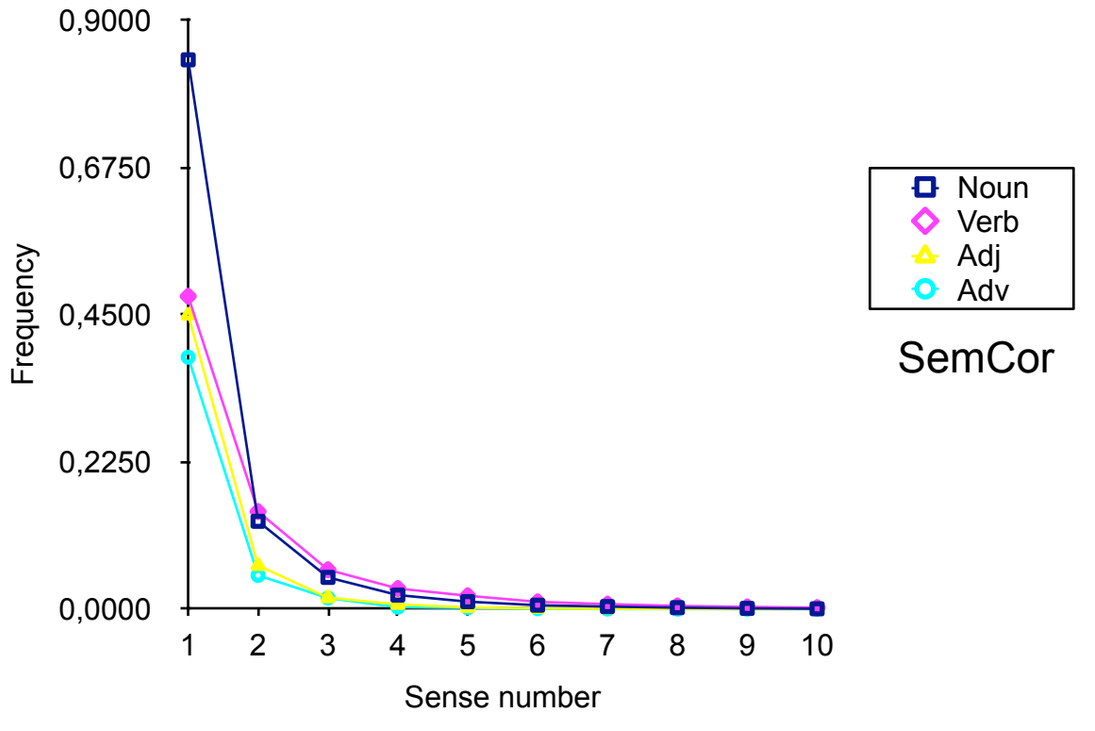
- Значение слова и многозначность
- Омонимия VS многозначность
  - –ключ
  - –платформа
- Метонимия
  - – Я три *тарелки* съел
- Зевгма
  - –За окном шел снег и рота красноармейцев
- Типы омонимов
  - –омофоны (луг-лук, плод-плот)
  - –омографы (м’ука - мук’а, гв’оздик-гвозд’ик)

Отношения между словами

- Синонимия
  - Машина / автомобиль
- Антонимия
  - большой / маленький, вверх / вниз, ложь / истина
- Обобщение и детализация (hyponym and hypernym/superordinate)
  - машина - транспорнтое средство
  - яблоко - фрукт
- Меронимы (партонимы) и холонимы
  - колесо - машина

Многозначность на практике

- Text-to-Speech
  - омографы
- Информационный поиск
- Извлечение информации
- Машинный перевод
- Эмоциональная окраска
- Закон Ципфа (Zipf law)



WordNet

- База лексических отношений
  - содержит иерархии
  - сочетает в себе тезаурус и словарь
  - доступен on-line
  - разрабатываются версии для языков кроме английского (в т.ч. для русского)

Категория	**Уникальных форм**
Существительные	**117,097**
Глаголы	**11,488**
Прилагательные	**22,141**
Наречия	**4,601**

- http://http://wordnet.princeton.edu/
- http://wordnet.ru/

Формат WordNet

The noun "bass" has 8 senses in WordNet.

1. bass[1] - (the lowest part of the musical range)
2. bass[2], bass part[1] - (the lowest part in polyphonic music)
3. bass[3], basso[1] - (an adult male singer with the lowest voice)
4. sea bass[1], bass[4] - (the lean flesh of a saltwater fish of the family Serranidae)
5. freshwater bass[1], bass[5] - (any of various North American freshwater fish with lean flesh (especially of the genus Micropterus))
6. bass[6], bass voice[1], basso[2] - (the lowest adult male singing voice)
7. bass[7] - (the member with the lowest range of a family of musical instruments)
8. bass[8] - (nontechnical name for any of numerous edible marine and freshwater spiny-finned fishes)

The adjective "bass" has 1 sense in WordNet.

1. bass[1], deep[6] - (having or denoting a low vocal or instrumental range)
   "a deep voice"; "a bass voice is lower than a baritone voice"; "a bass clarinet"

WordNet: отношения между словами

Relation	Also called	Definition	Example
Hypernym	Superordinate	From concepts to superordinates	$breakfast^1 \rightarrow meal^1$
Hyponym	Subordinate	From concepts to subtypes	$meal^1 \rightarrow lunch^1$
Member Meronym	Has-Member	From groups to their members	$faculty^2 \rightarrow professor^1$
Has-Instance		From concepts to instances of the concept	$composer^1 \rightarrow Bach^1$
Instance		From instances to their concepts	$Austen^1 \rightarrow author^1$
Member Holonym	Member-Of	From members to their groups	$copilot^1 \rightarrow crew^1$
Part Meronym	Has-Part	From wholes to parts	$table^2 \rightarrow leg^3$
Part Holonym	Part-Of	From parts to wholes	$course^7 \rightarrow meal^1$
Antonym		Opposites	$leader^1 \rightarrow follower^1$

Relation	Definition	Example
Hypernym	From events to superordinate events	$fly^9 \rightarrow travel^5$
Troponym	From a verb (event) to a specific manner elaboration of that verb	$walk^1 \rightarrow stroll^1$
Entails	From verbs (events) to the verbs (events) they entail	$snore^1 \rightarrow sleep^1$
Antonym	Opposites	$increase^1 \Longleftrightarrow decrease^1$

Иерархии WordNet

```
Sense 3
bass, basso --
(an adult male singer with the lowest voice)
=> singer, vocalist, vocalizer, vocaliser
   => musician, instrumentalist, player
      => performer, performing artist
         => entertainer
            => person, individual, someone...
               => organism, being
                  => living thing, animate thing,
                     => whole, unit
                        => object, physical object
                           => physical entity
                              => entity
                  => causal agent, cause, causal agency
                     => physical entity
                        => entity

Sense 7
bass --
(the member with the lowest range of a family of
musical instruments)
=> musical instrument, instrument
   => device
      => instrumentality, instrumentation
         => artifact, artefact
            => whole, unit
               => object, physical object
                  => physical entity
                     => entity
```

Как “зачение” определяется в WordNet

- Множестно синонимов называется синсет
- Пример

```
from nltk.corpus import wordnet
for synset in wordnet.synsets('chick'):
    print synset.definition
    print [lemma.name for lemma in synset.lemmas]
```

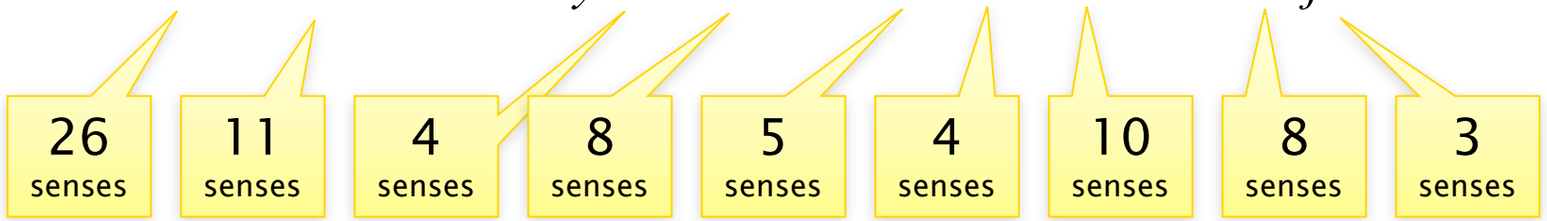
```
young bird especially of domestic fowl
['chick', 'biddy']
informal terms for a (young) woman
['dame', 'doll', 'wench', 'skirt', 'chick', 'bird']
```

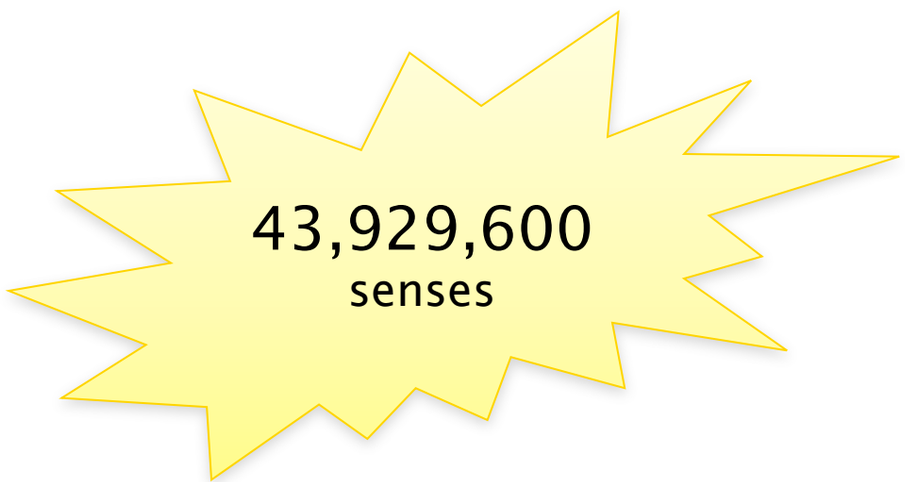
Вычислительная лексическая семантика

- Разрешение лексической многозначности
- Семантическая близость слов

Трудность разрешения лексической многозначности

I saw a man who is 98 years old and can still walk and tell jokes





Разрешение лексической многозначности (РЛМ)

- Word Sense Disambiguation (WSD)
  - определение значения слова в контексте
  - обычно предполагается фиксированный список значений (например WordNet)
- Сводится к задаче классификации
- Отличается от задачи разграничения значений (word sense discrimination)

РЛМ: варианты

- Определение значений только заранее выбранных слов (lexical sample task)
  - line - hard - serve; interest
  - Ранние работы
  - Обучение с учителем
- Определение значений всех слов (all-word task)
  - Проблема разреженности данных
  - Невозможно натренировать отдельный классификатор для каждого слова

Признаки

- Должны описывать контекст
- Предварительная обработка текста
  - –параграфы, предложения, части речи, леммы, синтаксический разбор?
- Признаки в словосочетаниях с позициями
- Множества соседей

- Проблема разреженности языка
  - – Использовать семантическую близость (далее)

Пример

An electric guitar and bass player stand off to one side, not really part of the scene, just as a sort of nod to gringo expectations perhaps.

Collocational features	
word_L3	electric
POS_L3	JJ
word_L2	guitar
POS_L2	NN
word_L1	and
POS_L1	CC
word_R1	player
POS_R1	NN
word_R2	stand
POS_R2	VB
word_R3	off
POS_R3	RB

Bag-of-words features	
fishing	0
big	0
sound	0
player	1
fly	0
rod	0
pound	0
double	0
runs	0
playing	0
guitar	1
band	0

Алгоритмы

- Любые методы классификации
  - (Пример) Наивный байесовский классификатор

Наивный байесовский классификатор

- Выбор наиболее вероятного значения

$$\hat{s} = \arg\max_{s \in S} P(s|f)$$

- По правилу Байеса

$$\hat{s} = \arg\max_{s \in S} \frac{P(s)P(f|s)}{P(f)} = \arg\max_{s \in S} P(s)P(f|s)$$

- Наивное предположение об условной независимости признаков

$$\hat{s} = \arg\max_{s \in S} P(s) \prod_{j=1}^{n} P(f_i|s)$$

Обучение наивного байесовского классификатора

- Метод максимального правдоподобия
- Другими словам, просто считаем

$$P(s_i) = \frac{count(s_i, w_j)}{count(w_j)} \qquad P(f_j \mid s) = \frac{count(f_j, s)}{count(s)}$$

- Алгоритм прост в реализации, но
  - Исчезновение значащих цифр → использовать сумму логарифмов вместо произведения
  - Нулевые вероятности → сглаживание

Вопрос на засыпку

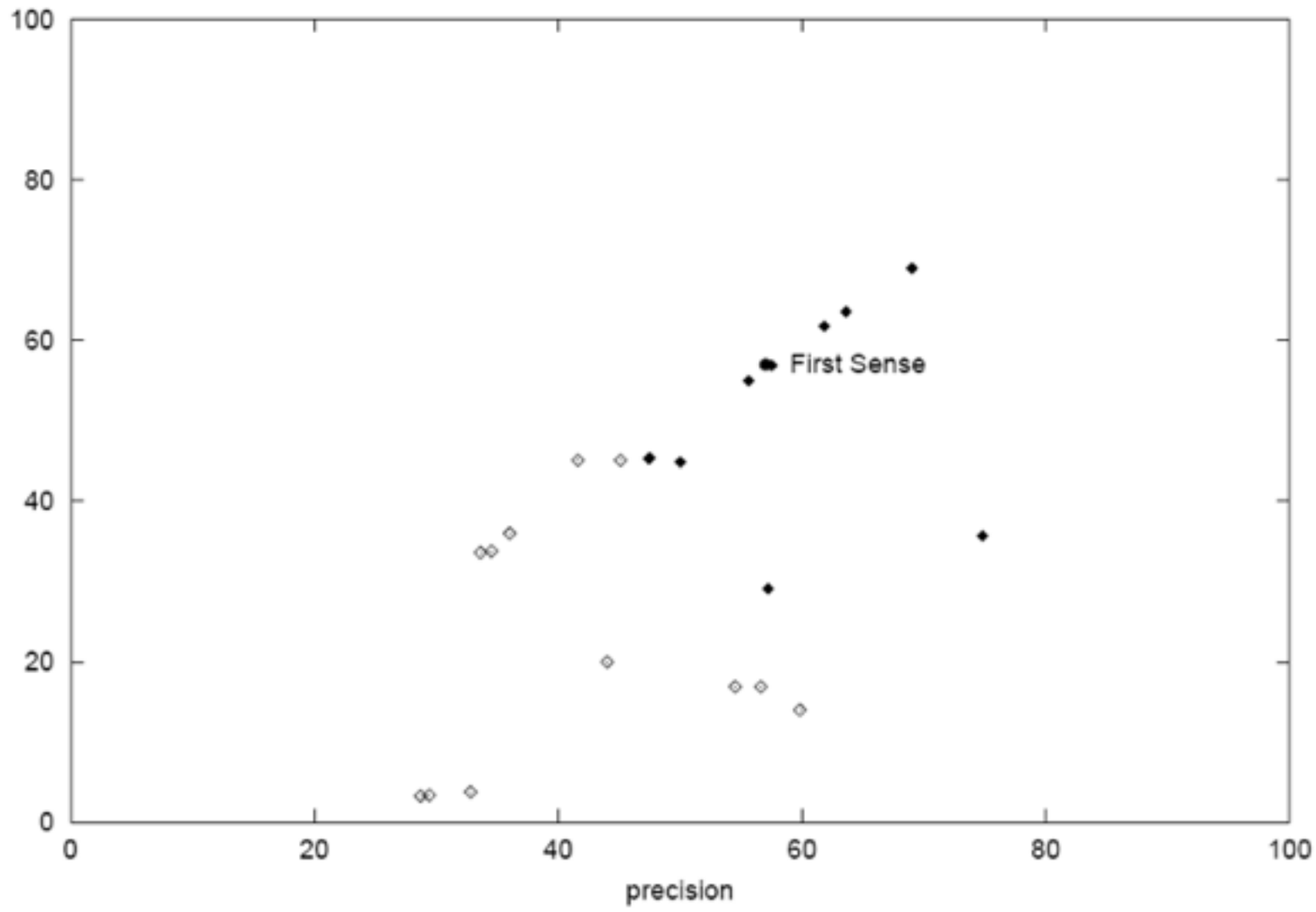
- Как сделать классификатор для задачи определения значений всех слов (all-word task)?

Методы оценки

- Внешние (in vivo)
  - Машинный перевод с/без РЛМ
- Внутренние (in vitro)
  - Применение к размеченным данным (SemCor, SENSEVAL, SEMEVAL)
  - Измерение точности и полноты в сравнении со стандартными значениями
- Нижняя граница
  - Выбор случайных значений работает плохо
  - Более сильные границы: наиболее частое значение, алгоритм Леска
- Верхняя граница: согласие экспертов
  - 75-80 для задачи определения значений всех слов со значениями из WordNet
  - до 90% с менее гранулированными значениями

Наиболее частое значение

- Сравнение методов на SENSEVAL-2



- McCarthy et. al. 2004 ACL - поиск наиболее частого значения по неразмеченному корпусу

Методы основанные на словорях и тезаурусах

- Алгоритм Леска (1986)
  - –Взять все определения целевого слова из словаря
  - –Сравнить с определениями слов в контексте
  - –Выбрать значение с максимальным пересечением
- Пример
  - *pine*
    1. a kind of evergreen tree with needle-shaped leaves
    2. to waste away through sorrow or illness
  - *cone*
    1. A solid body which narrows to a point
    2. Something of this shape, whether solid or hollow
    3. Fruit of certain evergreen trees
  - Определить значение: *pine cone*

Варианты алгоритма Леска

- Упрощенный (Simplified Lesk)
  - Взять все определения целевого слова из словаря
  - Сравнить со ~~определениями~~ словами в контексте
  - Выбрать значение с максимальным пересечением
- Корпусный (Corpus Lesk)
  - Включить предложения из размеченного корпуса в сигнатуру каждого значения
  - Взвесить слова через IDF
  - IDF(w) = -log P(w)
  - Показывает лучшие результаты
  - Использовался как нижняя граница на SENSEVAL

Самонастройка (Bootstrapping)

- Yarowsky (1995)
  - Начать с маленького множества данных, размеченного вручную
  - Натренировать список принятия решений
  - Применить классификатор к неразмеченным данным
  - Переместить примеры в которых мы уверены в тренировочное множество
  - Повторить!
- Требует хорошей метрики уверенности
  - логарифмическое отношение правдоподобия
- Эвристики для получения начальных данных
  - одно значение на словосочетание
  - одно значение на дискурс

Алгоритм Yarowsky

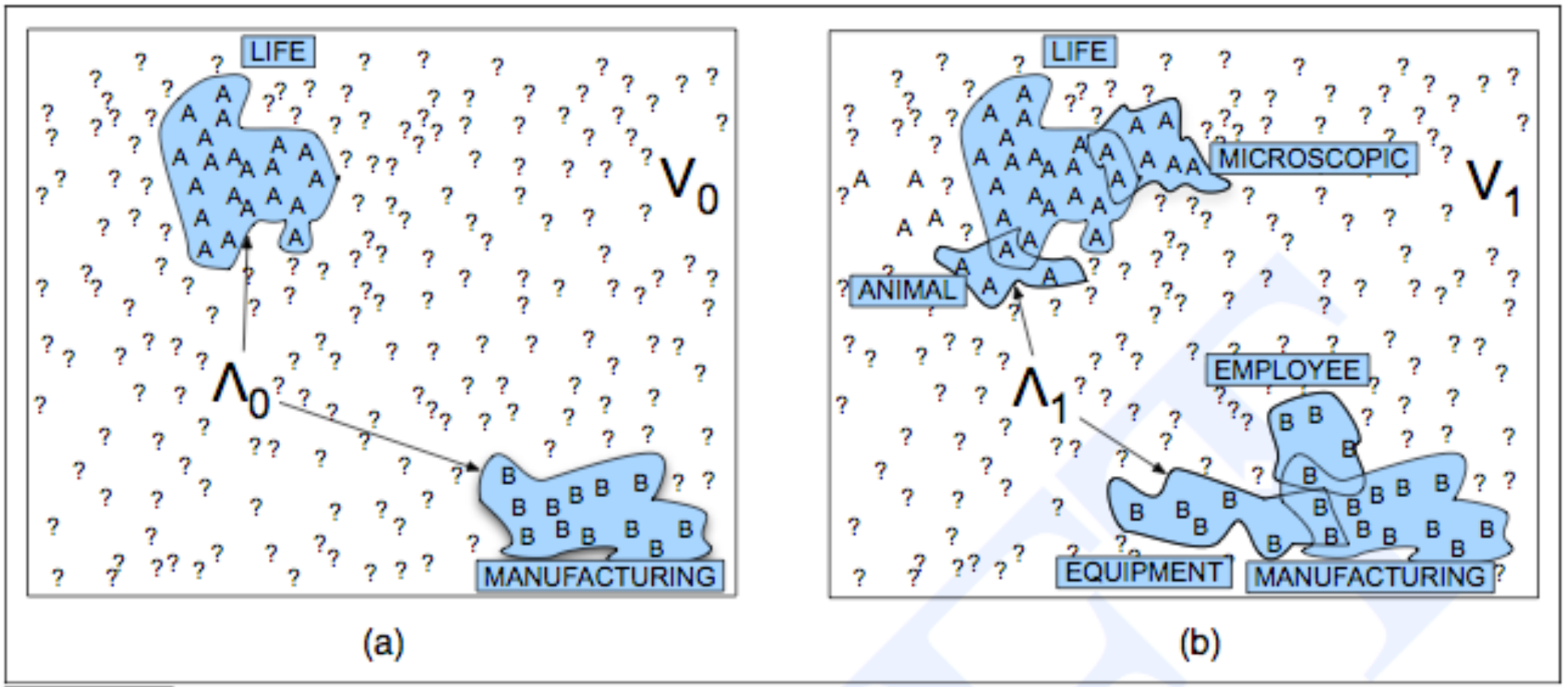


Figure 20.4 The Yarowsky algorithm disambiguating "plant" at two stages; "?" indicates an unlabeled observation, A and B are observations labeled as SENSE-A or SENSE-B. The initial stage (a) shows only seed sentences Λ_0 labeled by collocates ("life" and "manufacturing"). An intermediate stage is shown in (b) where more collocates have been discovered ("equipment", "microscopic", etc.) and more instances in V_0 have been moved into Λ_1, leaving a smaller unlabeled set V_1. Figure adapted from Yarowsky (1995).

Семантическая близость слов

- Подходы на основе тезаурусов
- Подходы на основе статистики

Мотивация

- Хороший признак для многих задач
- Позволяет бороться с разреженностью языка
- Имеет прикладное применение
  - поиск опечаток (с учетом семантики)
  - поиск плагиата
  - извлечение информации

Подход на основе тезаурусов

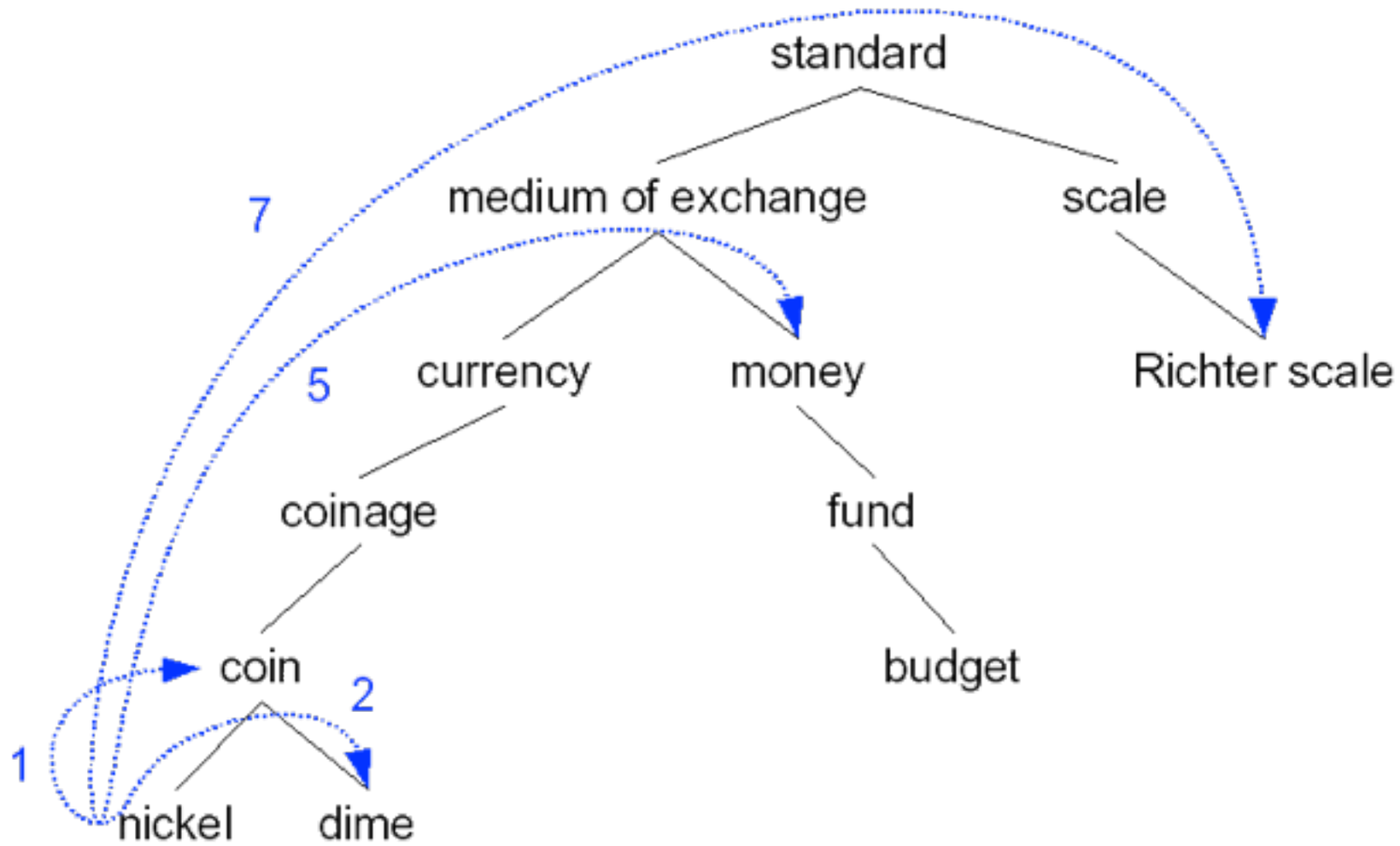
- Близость по пути
- Метод Резника
- Метод Лина
- Расширенный алгоритм Леска

Семантическая близость слов в тезаурусах

- Можно использовать любые отношения между словами
- На практике используется иерархическая структура и иногда описания значений
- Похожесть (similarity) VS связность (relatedness)
  - машина и топливо: не похожи но связаны
  - машина и велосипед: похожи

Близость по пути в иерархии

- Два понятия семантически близки, если они находятся рядом в иерархии



Близость между словами

- Только что мы посчитали близость между понятиями
- Перейдем ко словам

- simpath(c1,c2) = -log (pathlen(c1,c2))
- wordsim(w1,w2) = $\max_{c1 \in senses(w1), c2 \in senses(w2)}$ sim(c1,c2)

Другие методы

- Сначала немного определений...
  - –Информационное содержимое
  - –Наименьший общий предок

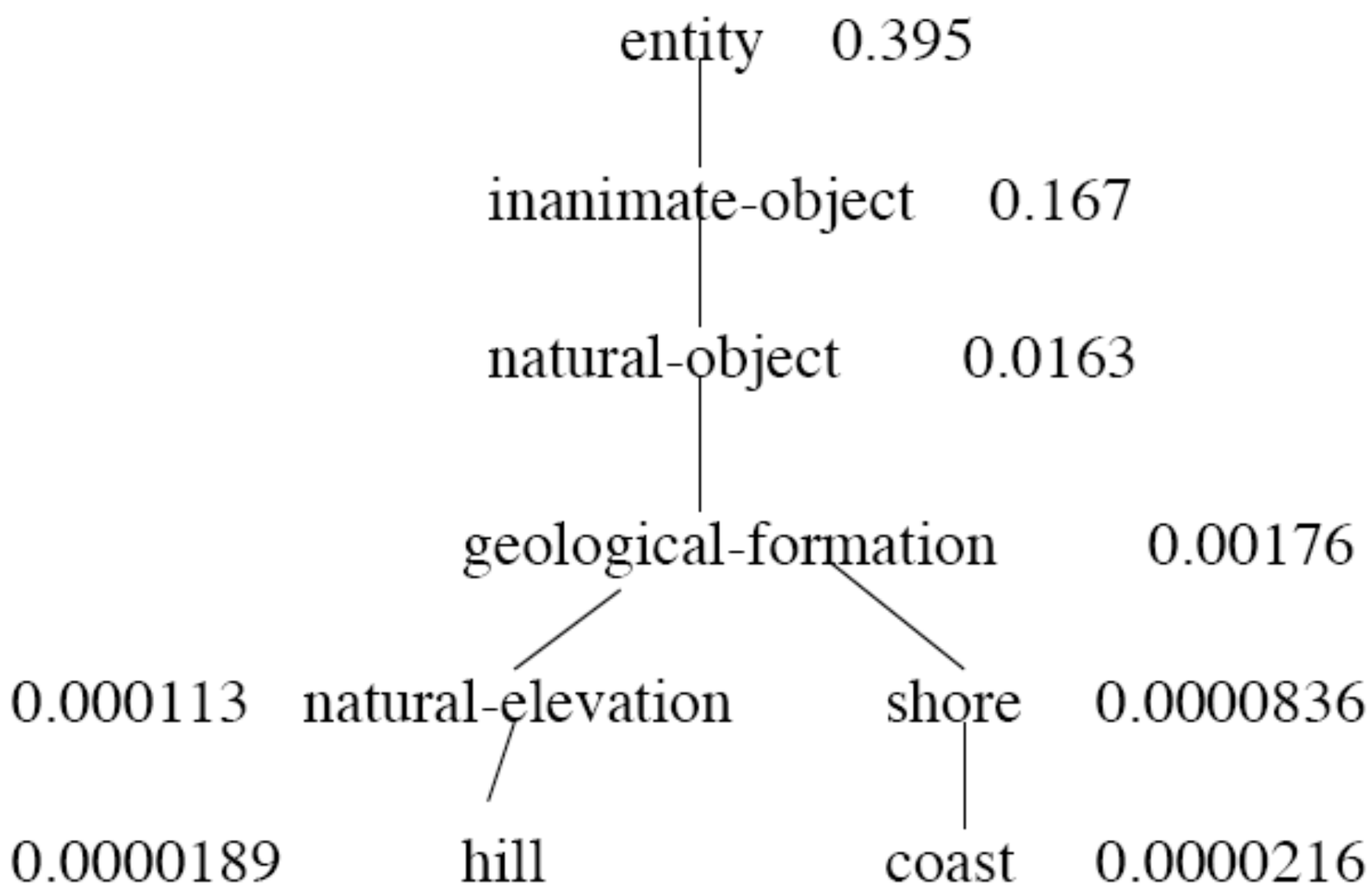
Вероятность класса

- Определим P(C) как:
  - Вероятность, что случайно выбранное слово в корпусе является экземпляром класса C
  - P(root)=1
  - Чем ниже узел в иерархии, тем ниже вероятность

$$P(c) = \frac{\sum_{w \in words(c)} count(w)}{N}$$

Информационное содержимое

- Расширяем иерархию WordNet вероятностями P(C)



Определения

- Информационное содержимое
  - IC(c)=-log(P(c))
- Наименьший общий предок
  - LCS (c1,c2)

Метод Резника

- Resnik (1995)
  - Чем больше общего между понятиями, тем более они похожи
  - $sim_{resnik}(c1,c2) = IC(LCS(c1,c2)) =$

$$= -\log P(LCS(c1,c2))$$

Метод Лина

- Dekang Lin (1998)
  - При вычислении близости также надо учитывать различие между понятиями
- Идея может быть выражена как

$$sim_{Lin}(c_1, c_2) = \frac{2 \times log(P(LCS(c_1, c_2)))}{log(P(c_1)) + log(P(c_2))}$$

$$sim_{Lin}(hill, coast) = \frac{2 \times log(P(geological\_information))}{log(P(hill)) + log(P(coast))} = 0.59$$

Расширенный алгоритм Леска

- Две концепции похожи, если их описания содержат похожие слова
  - *Drawing paper*: **paper** that is **specially prepared** for use in drafting
  - *Decal*: the art of transferring designs from **specially prepared paper** to a wood or glass or metal surface
- Каждому общему словосочетанию длины n назначить вес n^2
- paper + specially prepared: 1+4 = 5

Резюме: методы, основанные на тезаурусах

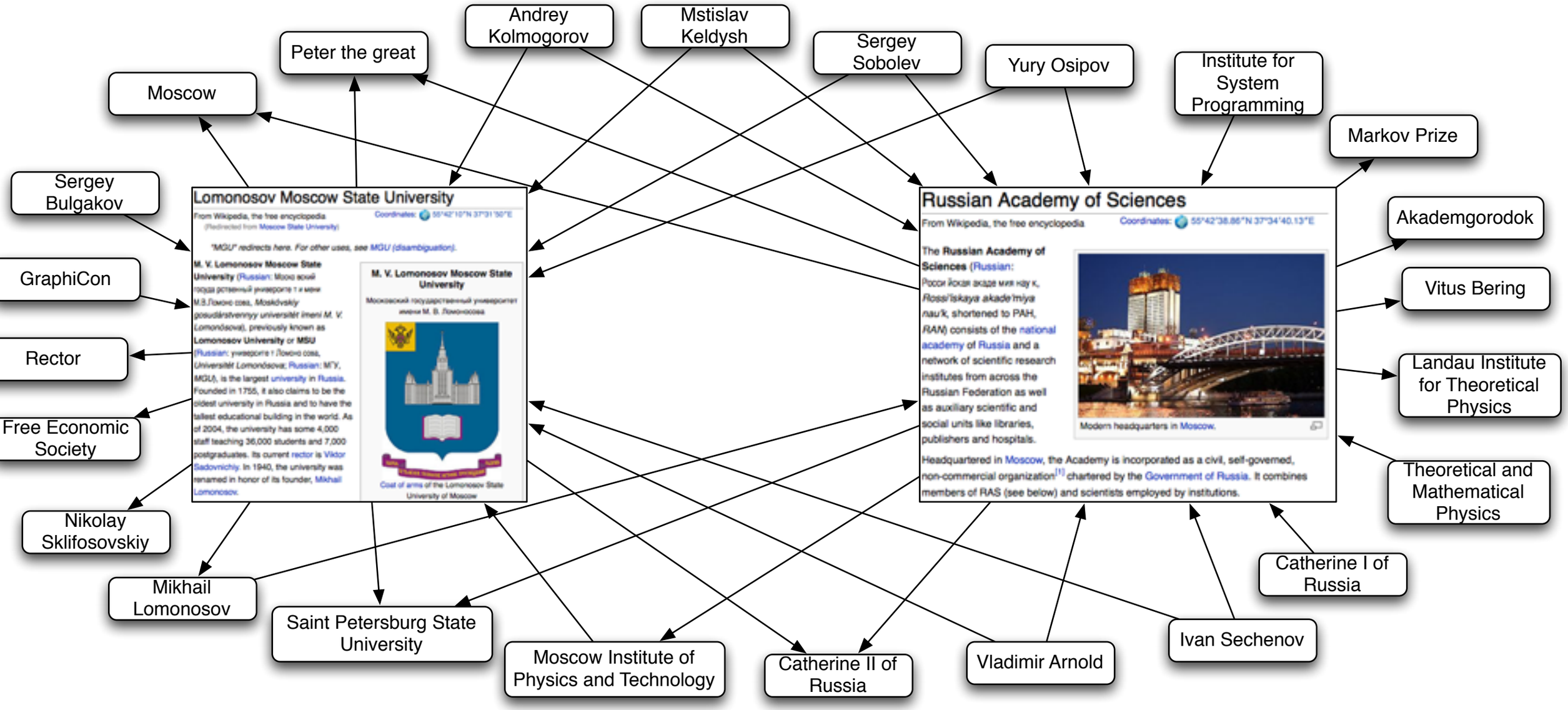
$$\begin{aligned}
\text{sim}_{\text{path}}(c_1, c_2) &= -\log \text{pathlen}(c_1, c_2) \\
\text{sim}_{\text{Resnik}}(c_1, c_2) &= -\log P(\text{LCS}(c_1, c_2)) \\
\text{sim}_{\text{Lin}}(c_1, c_2) &= \frac{2 \times \log P(LCS(c_1, c_2))}{\log P(c_1) + \log P(c_2)} \\
\text{sim}_{\text{jc}}(c_1, c_2) &= \frac{1}{2 \times \log P(\text{LCS}(c_1, c_2)) - (\log P(c_1) + \log P(c_2))} \\
\text{sim}_{\text{eLesk}}(c_1, c_2) &= \sum_{r,q \in \text{RELS}} \text{overlap}(\text{gloss}(r(c_1)), \text{gloss}(q(c_2)))
\end{aligned}$$

Проблемы с подходом, основанном на тезаурусе

- Не доступен для многих языков
- Много слов пропущено
- Используются только обобщения и детализация
  - Хорошо работает для имен существительных
  - Для прилагательных и глаголов намного хуже

Семантическая близость

- Нормализованное количество общих соседей



- Близкие концепции чаще встречаются вместе

Статистический подход к оценки близости слов

- Firth (1957): “You shall know a word by the company it keeps!”
- Пример

Бутылка **tezgüino** стоит на столе

Все любят **tezgüino**

Tezgüino делает тебя пьяным

Мы делаем **tezgüino** из кукурузы

- Идея:
  - из контекста можно понять значение слова
  - надо взять контекст и посмотреть, какие еще слова имеют такой же контекст

Векторное представление контекста

- Для каждого слова из словаря определим бинарный признак, показывающий встречаемость вместе с целевым словом **w**
- $w=(f_1,f_2,f_3,\ldots,f_N)$
- w= tezgüino, v1=бутылка, v2=кукуруза, v3=матрица
- w = (1,1,0,...)

Идея

- Задать два слова через разреженный вектор признаков
- Применить метрику близости векторов
- Два слова близки, если векторы близки

	arts	boil	data	function	large	sugar	summarized	water
apricot	0	1	0	0	1	1	0	1
pineapple	0	1	0	0	1	1	0	1
digital	0	0	1	1	1	0	1	0
information	0	0	1	1	1	0	1	0

Статистический подход к оценки близости слов

- Необходимо определить 3 вещи:
  - совместная встречаемость
  - вес термина
  - близость между векторами

Совместная встречаемость

- Проблема разреженности
  - Нужны большие корпуса

Вес термина

- Manning and Schuetze (1999)

$$\begin{aligned}
\text{assoc}_{\text{prob}}(w,f) &= P(f|w) \\
\text{assoc}_{\text{PMI}}(w,f) &= \log_2 \frac{P(w,f)}{P(w)P(f)} \\
\text{assoc}_{\text{Lin}}(w,f) &= \log_2 \frac{P(w,f)}{P(w)P(r|w)P(w'|w)} \\
\text{assoc}_{\text{t-test}}(w,f) &= \frac{P(w,f)-P(w)P(f)}{\sqrt{P(f)P(w)}}
\end{aligned}$$

Близость между векторами

$$\text{sim}_{\text{cosine}}(\vec{v},\vec{w}) = \frac{\vec{v}\cdot\vec{w}}{|\vec{v}||\vec{w}|} = \frac{\sum_{i=1}^{N} v_i \times w_i}{\sqrt{\sum_{i=1}^{N} v_i^2}\sqrt{\sum_{i=1}^{N} w_i^2}}$$

$$\text{sim}_{\text{Jaccard}}(\vec{v},\vec{w}) = \frac{\sum_{i=1}^{N} \min(v_i,w_i)}{\sum_{i=1}^{N} \max(v_i,w_i)}$$

$$\text{sim}_{\text{Dice}}(\vec{v},\vec{w}) = \frac{2\times\sum_{i=1}^{N} \min(v_i,w_i)}{\sum_{i=1}^{N} (v_i+w_i)}$$

$$\text{sim}_{\text{JS}}(\vec{v}||\vec{w}) = D(\vec{v}|\frac{\vec{v}+\vec{w}}{2}) + D(\vec{w}|\frac{\vec{v}+\vec{w}}{2})$$

Современное направление

- Использование нейронных сетей для получения векторного представления слов
  - –word2vec, GloVe
  - –http://code.google.com/p/word2vec/
- Близость к слову france ->
- Задача поиска аналогий
  - – v(king)-v(man)+v(woman)=?
- Gensim: реализация на Python

Word	Cosine distance
spain	0.678515
belgium	0.665923
netherlands	0.652428
italy	0.633130
switzerland	0.622323
luxembourg	0.610033
portugal	0.577154
russia	0.571507
germany	0.563291
catalonia	0.534176

Оценка качества

- Внутренняя
  - –Коэффициент корреляции между
    - результатами алгоритма и
    - значениями, поставленными людьми
- Внешняя
  - –Встроить в приложение
    - Поиск опечаток
    - Поиск плагиата
    - Разрешение лексической многозначности

Заключение

- **Лексическая семантика** изучает значения отдельных слов
- **WordNet** содержит различные отношения между словами, синсеты задают **значения** слов
- **Разрешение лексической многозначности** - задача определения значений слов
- **Семантическая близость** между словами - полезный инструмент для монгих приложений

Что не было рассказано

- Композиционная семантика
- Представление знаний
- Семантические поля и семантические роли
  - PropBank
  - FrameNet
- Задача разграничения значений
- Автоматическое извлечение отношений между словами
- ...

Следующая лекция

- Информационный поиск
- Вопросно-ответные системы
- Автоматическое реферирование

Основы обработки текстов

Лекция 9

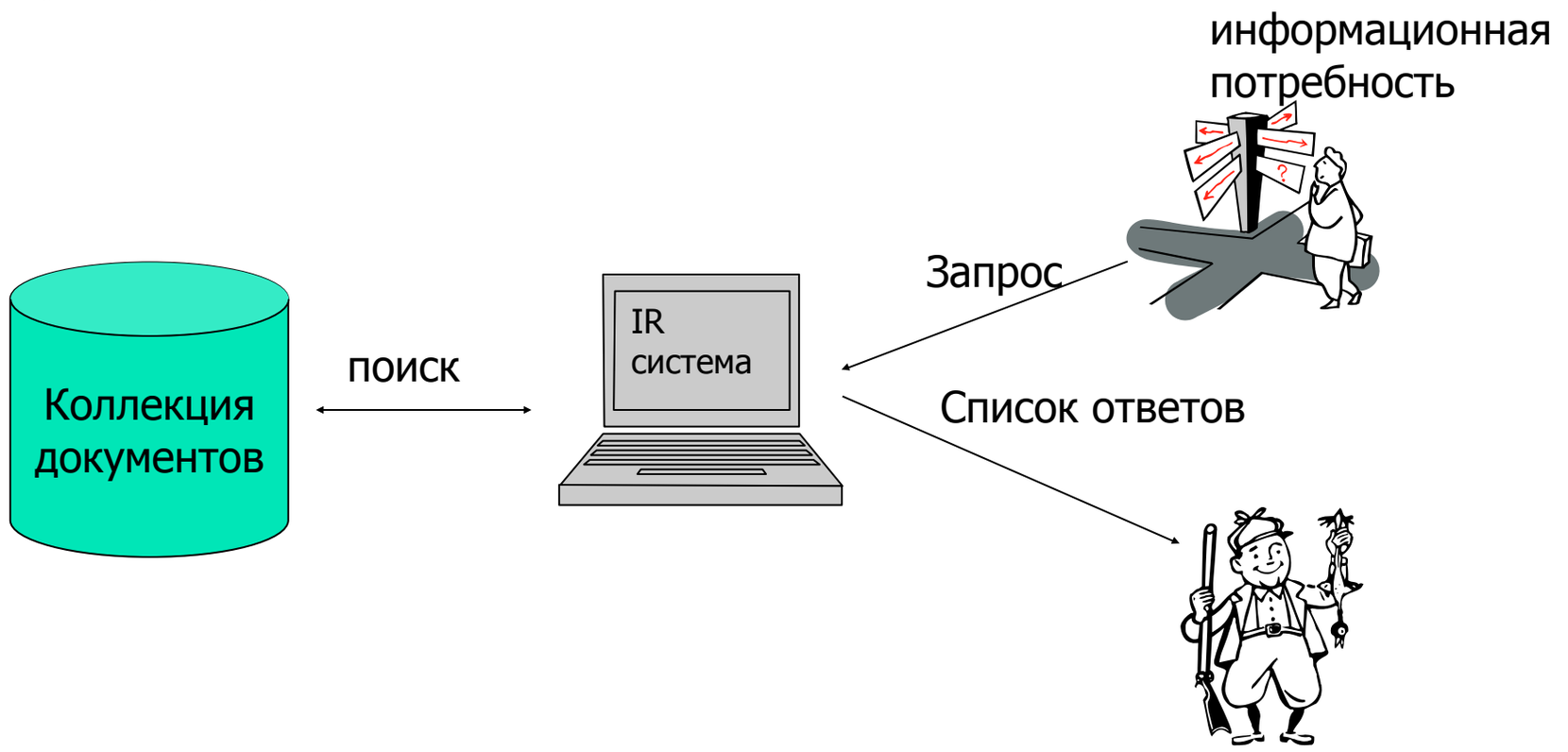
Приложения обработки текстов

План

- Информационный поиск
- Вопросно-ответные системы
- Автоматическое реферирование

Информационный поиск

- Information retrieval (IR)
- Поиск всех документов из заданного множества, отвечающих запросам пользователя



Проблема информационного поиска

- Первое приложение в библиотечном деле

ISBN: 0-201-12227-8

Author: Salton, Gerard

Title: Automatic text processing: the transformation, analysis, and retrieval of information by computer

Editor: Addison-Wesley

Date: 1989

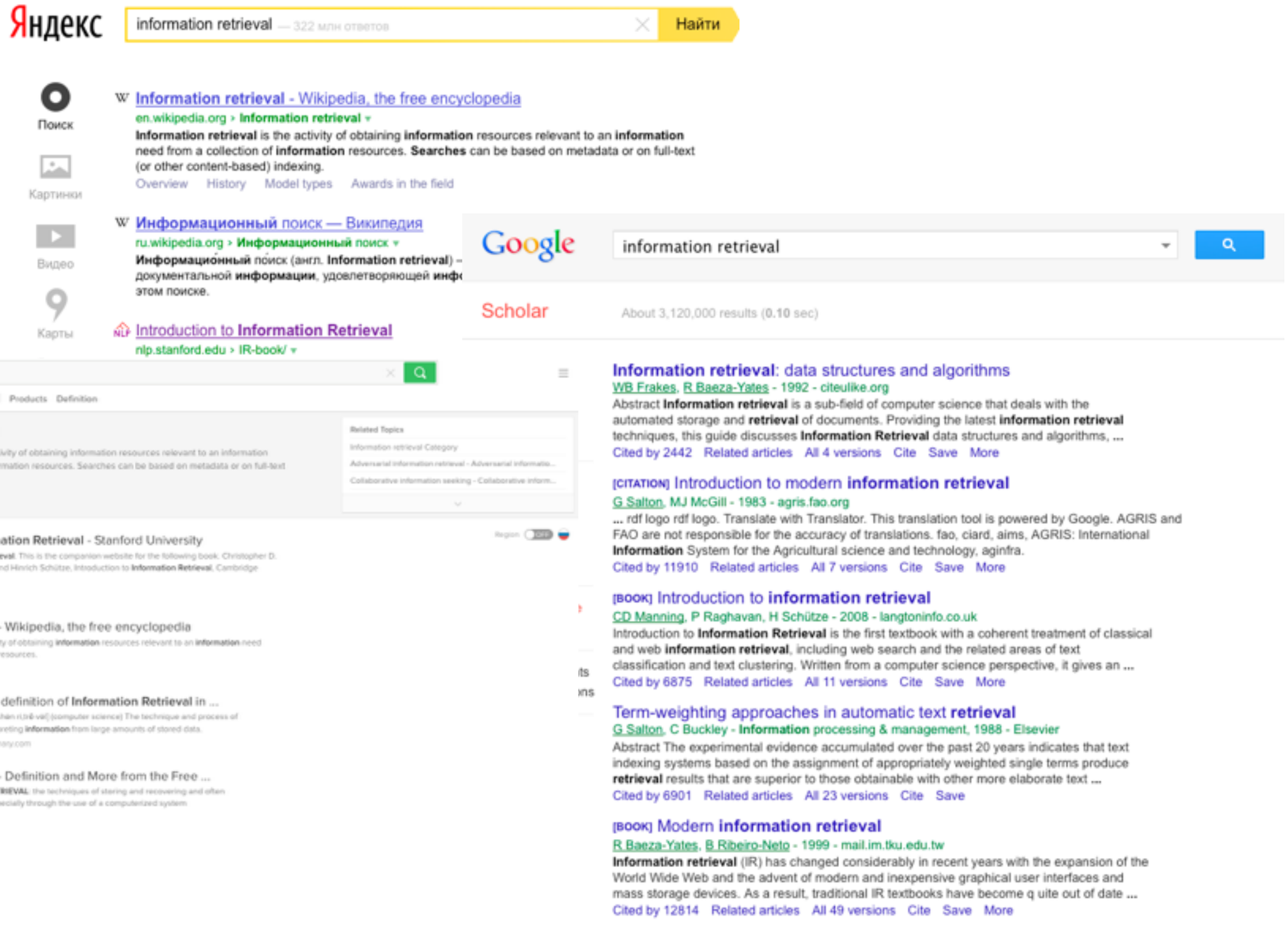
Content: <Text>

- Поиск по внешним аттрибутам - поиск в БД
- IR: поиск по контенту

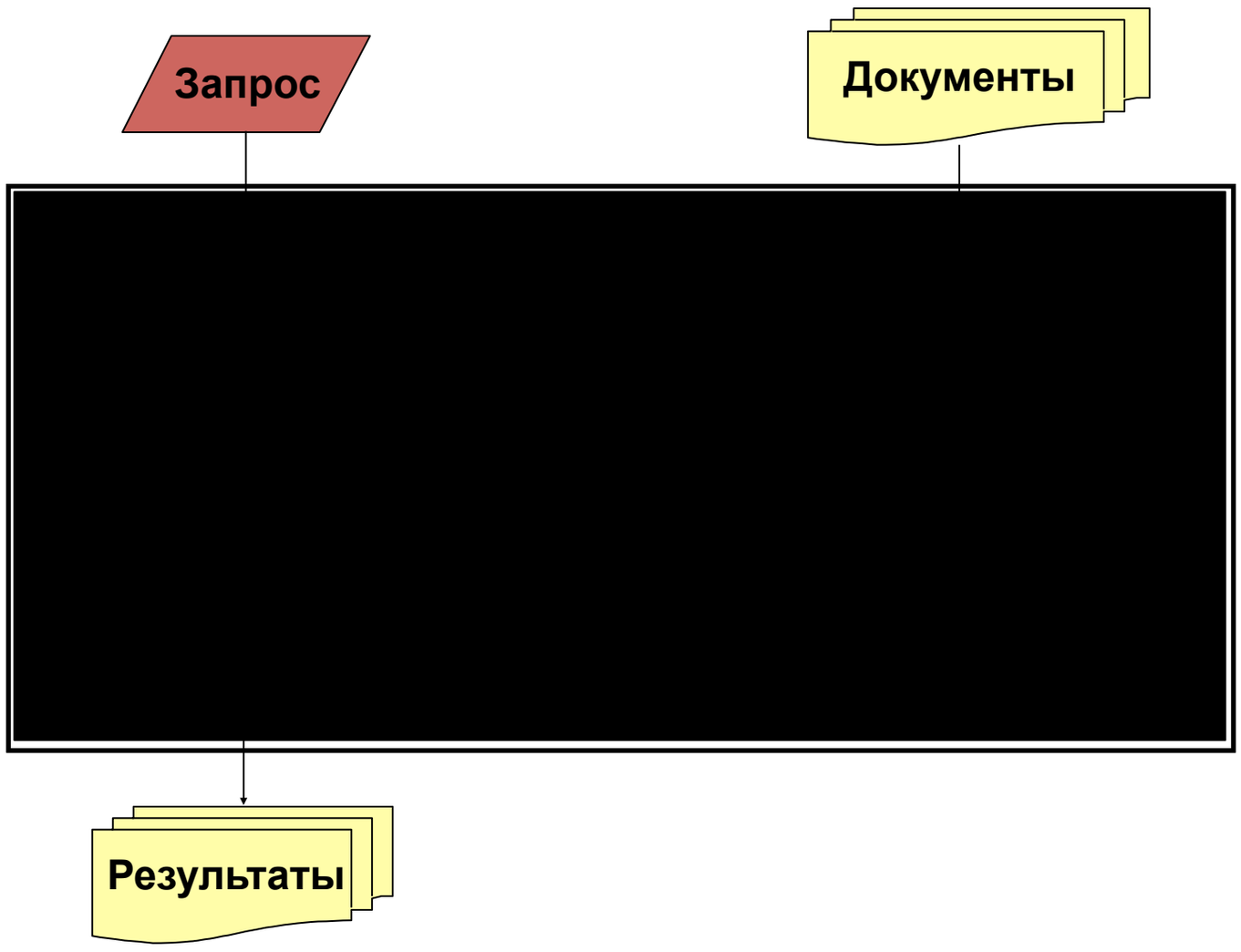
Возможные подходы

- Поиск близких строк
  - Медленно
  - Тяжело улучшать
- Индексирование
  - Быстро
  - Возможности для улучшений

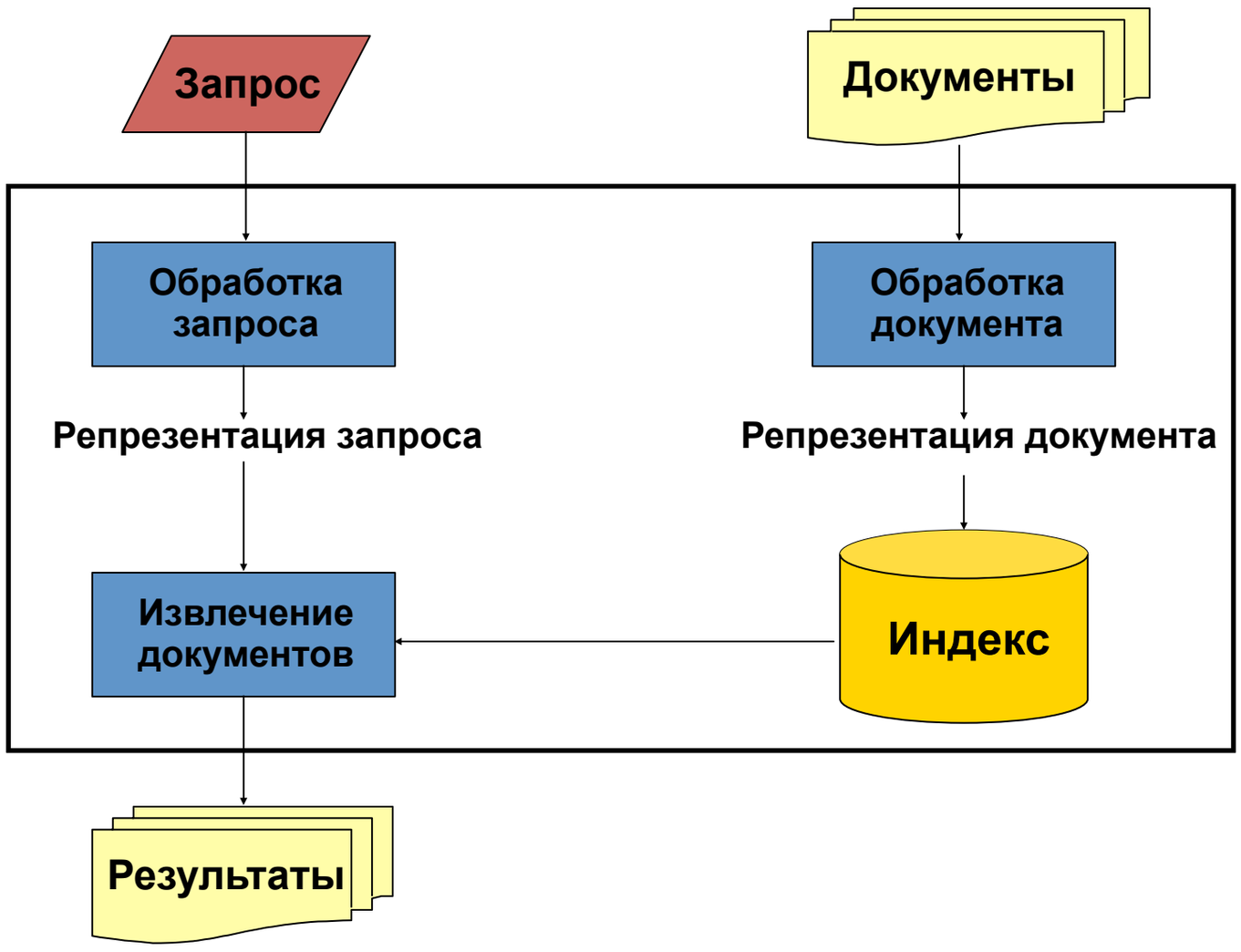
Примеры систем



Архитектура систем



Архитектура систем



Основные проблемы

- Обработка запроса и документа
  - –Какой наилучший способ представления запроса и документа
- Извлечение документов
  - –Как понять какой документ наилучшим образом удовлетворяет запросу
- Оценка систем
  - –Как понять что система работает хорошо

Представление документа

- Модель мешка слов (bag-of-words)
- 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) * idf(t)

Варианты tf-idf

- tf(t, D)=freq(t,D)
- tf(t, D)=log[freq(t,D)]
- tf(t, D)=log[freq(t,D)]+1
- tf(t, D)=freq(t,d)/Max[f(t,d)]

idf(t) = log(N/n)

n = # документов содержит t

N = # документов в корпусе

Предварительная обработка

- Стоп-слова - Функциональные слова не несут полезной инорфмации для IR систем
  - Удаление стоп-слов часто улучшает качество IR систем
  - Часто используются “стандартные” списки стоп-слов
- Стемминг
- Лемматизация

Результат индексирования

- Инвертированный индекс

Doc 1

This is a sample document with one sample sentence

Doc 2

This is another sample document

Term	# docs	Total freq
This	2	2
is	2	2
sample	2	3
another	1	1
…	…	…

Doc id	Freq
1	1
2	1
1	1
2	1
1	2
2	1
2	1
…	…
…	…

Извлечение документов

- Запрос из одного слова
  - Берем инвертированый список для слова
- Запрос из нескольких слов
  - Комбинирование нескольких списков
  - Как интерпретировать вес?
  - Модель информационного поиска

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 - логическая конъюнкция слов
- Запрос - Булево выражение

$$R(Q, D) = Q \rightarrow D$$

$D = t_1 \wedge t_2 \wedge \ldots \wedge t_n$
$Q = (t_1 \wedge t_2) \vee (t_3 \wedge \neg t_4)$
$Q \rightarrow D$, то есть $R(D, Q) = 1$.

- Проблемы
  - R - либо 0, либо 1 (неупорядоченное множество документов
  - Сложно писать запросы

Векторная модель

- Векторное пространство всех слов

$<t_1, t_2, t_3, \ldots, t_n>$

- Документ

$D = < a_1, a_2, a_3, \ldots, a_n>$

a_i = вес t_i в D

- Запрос

$Q = < b_1, b_2, b_3, \ldots, b_n>$

b_i = вес t_i в Q

Матричное представление

Пространство документов	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

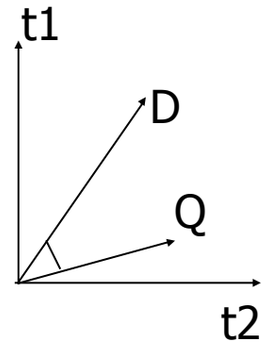
Пространство терминов

Разреженная матрица!

Подсчет близости

Скалярное произведение

$$Sim(D,Q) = \sum (a_i * b_i)$$



Косинус

$$Sim(D,Q) = \frac{\sum_i (a_i * b_i)}{\sqrt{\sum_i a_i^{\,2} * \sum_i b_i^{\,2}}}$$

Мера Дайса

$$Sim(D,Q) = \frac{2\sum_i (a_i * b_i)}{\sum_i a_i^{\,2} + \sum_i b_i^{\,2}}$$

Мера Жаккара

$$Sim(D,Q) = \frac{\sum_i (a_i * b_i)}{\sum_i a_i^{\,2} + \sum_i b_i^{\,2} - \sum_i (a_i * b_i)}$$

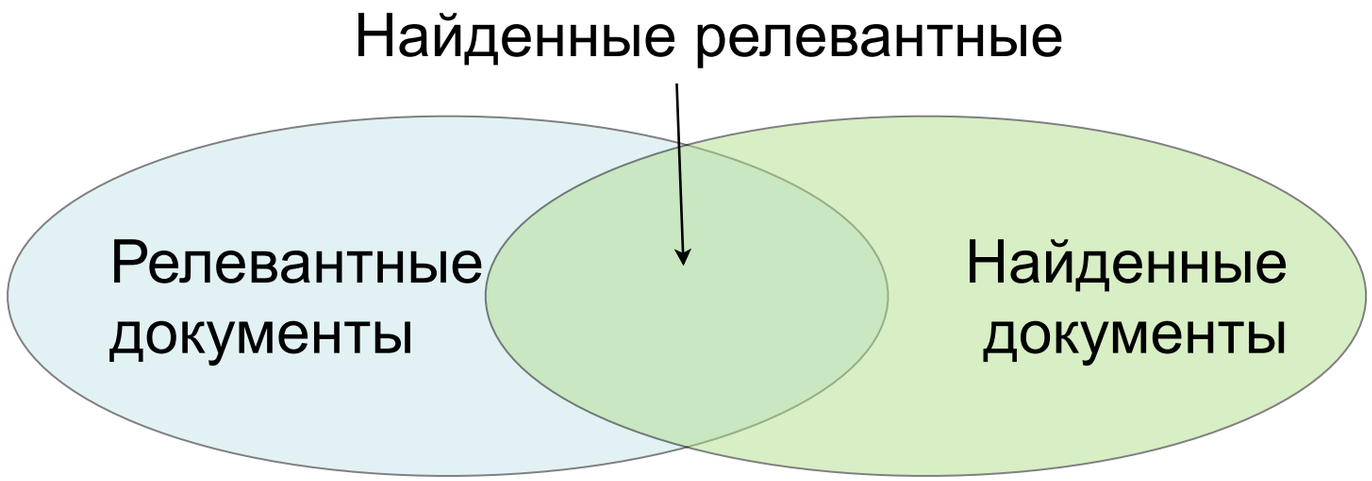
Какие еще бывают модели

Properties of the Model / Mathematical Basis	without term-interdependencies	with term-interdependencies: immanent term-dependencies	with term-interdependencies: transcendent term-interdependencies
set-theoretic	Standard Boolean, Extended Boolean		Fuzzy Set
algebraic	Vector Space, Extended Boolean	Generalised Vector Space, Latent Semantic, Spread. Activation Neuronal Network	Topic-based Vector Space, Balanced Topic-based Vector Space, Backpropagation Neuronal Network
probabilistic	Binary Interdependence, Inference Network, Belief Network	Language	Retrieval by Logical Imaging

*http://en.wikipedia.org/wiki/Information_retrieval

Оценка систем

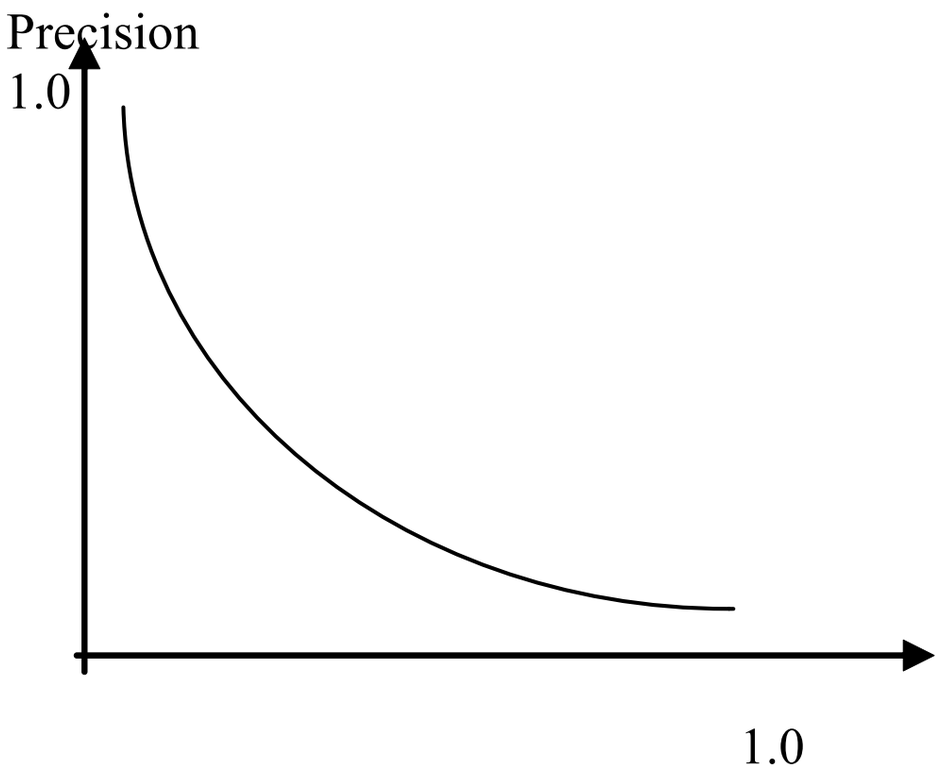
- Точность и полнота



  - Точность = найденные релевантные / найденные документы
  - Полнота = найденные релевантные / релевантные документы

Точность и полнота

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



$$\mathrm{AveP} = \int_0^1 p(r)dr$$

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

$$\mathrm{AveP} = \frac{\sum_{k=1}^{n}(P(k) \times \mathrm{rel}(k))}{\text{number of relevant documents}}$$

$rel(k) \in \{0,1\} = 1$, если k-й 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 = # тестовых запросов

Ранг		
1	4	1ый рел. док.
5	8	2ой рел. док.
10		3ий рел. док.

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

Темы для дальнейшего изучения

- Ранжирование
  - PageRank (Google), HITS, ...
- Семантический поиск
  - ключевые слова VS ключевые понятия
- IR для (полу-) структурированных данных
- Сбор данных в Вебе
- Мультимедийный поиск
- Исследовательский поиск
- Многоязычный поиск
- Сжатие и хранение данных
- Нечеткий поиск
- Учет обратной связи от пользователей
- Персонализация
- Инструменты: Apache Lucene, ElasticSearch, Apache Nutch

Вопросно-ответные системы

Какой национальности бывший папа римский Бенедикт XVI?

Ватикан выступил во вторник, 12 мая, с опровержением информации о том, что Папа Римский Бенедикт XVI в юности состоял в гитлерюгенде. "Йозеф Рацингер (имя понтифика, **немца по национальности**) никогда не состоял в гитлерюгенде - идеологической нацистской организации.

Короткий фрагмент текста, не URL

Ответ: Немец

Примеры систем

WolframAlpha computational... knowledge engine

Enter what you want to calculate or know about:

How far is San-Francisco from Moscow?

Examples Random

Assuming Moscow (Russia) | Use Moscow (Idaho, USA) or more instead

Input interpretation:

Moscow to San Francisco, California, United States

Distance: Show non-metric units

9472 km (kilometers)

Direct travel times: More

aircraft (550 mph)	10 hours 40 minutes
sound	7 hours 40 minutes
light in fiber	44.3 ms (milliseconds)
light in vacuum	31.6 ms (milliseconds)

(assuming constant–speed great–circle path)

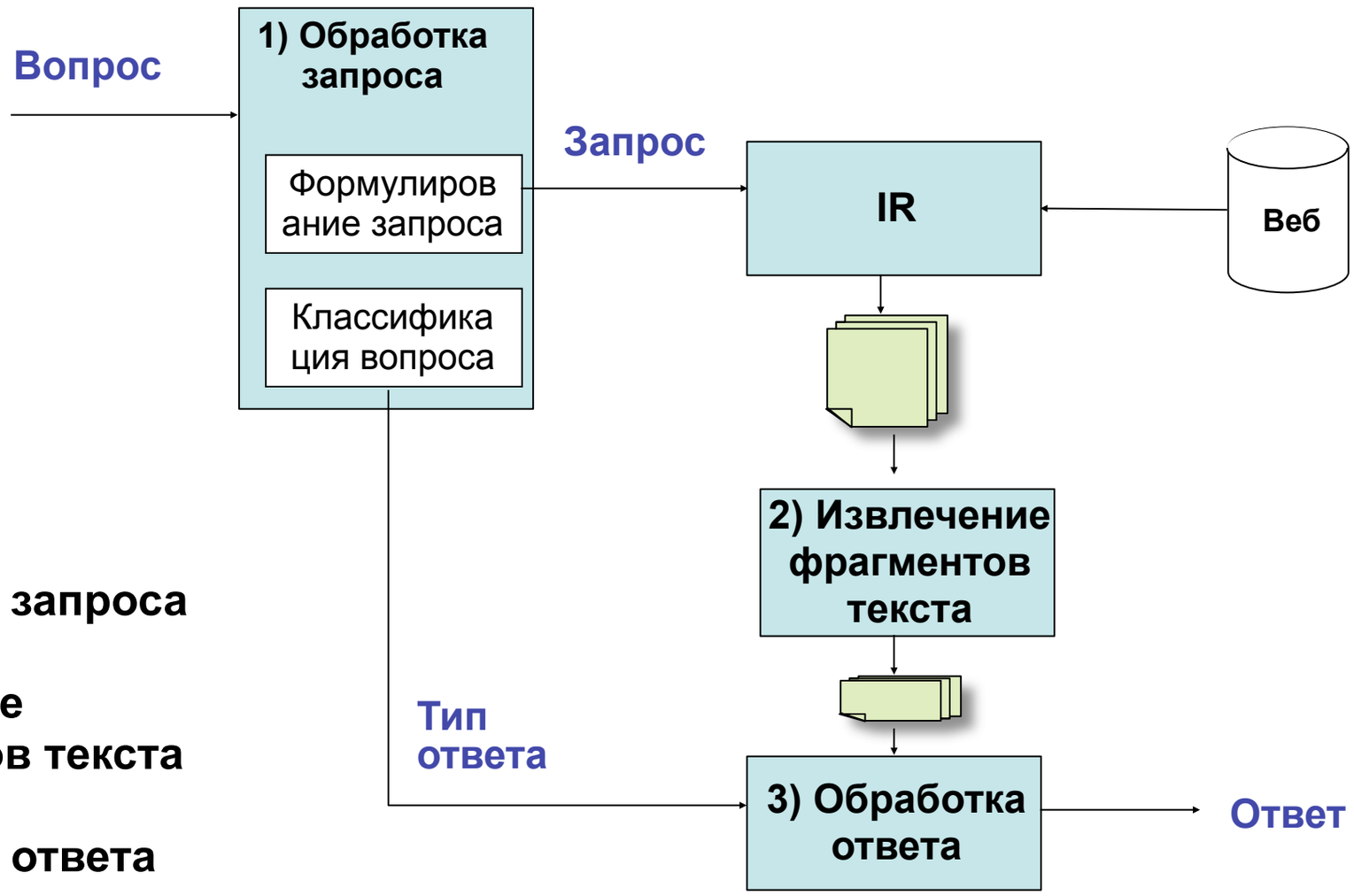
Типы вопросов

О фактах	Какая обычная высота жирафа? Где расположен главный офис Google ?
Списки	Какие страны экспортируют нефть? Какие названия имеют штаты США?
Определения	Кто такой Франсуа Томбалбай? Что такое квазар?

Вопросы о фактах

- Ответом служит простой факт
  - Примеры:
    - Где расположен Лувр?
    - Какая называется валюта Китая?
    - Какой официальный язык Алжира?
- Существует большая разница между постановкой вопроса и описанием ответа в тексте
  - Какая компания является лидером по производству открыток?
  - Компания "Арт и Дизайн" десять лет назад создала в России практически новый рынок. Теперь она является лидером среди отечественных производителей поздравительных открыток.

Типичная архитектура QA-систем



Три этапа:

1. **Обработка запроса**
2. **Извлечение фрагментов текста**
3. **Обработка ответа**

Обработка запроса

- Из вопроса на естественном языке извлекаем:
  - ключевые слова для запроса к информационно-поисковой системе (Формулирование запроса)
  - Тип ответа, специфицирующий класс сущности, возвращаемой в качестве ответа (Классификация вопроса)

Формулирование запроса

- Извлечь ключевые термины из вопроса
  - возможно расширить вопрос лексически/ семантически близкими словами
- Вопрос моделируется как множество ключевых слов

Question (from TREC QA track)	Lexical terms
Q002: What was the monetary value of the Nobel Peace Prize in 1989?	monetary, value, Nobel, Peace, Prize, 1989
Q003: What does the Peugeot company manufacture?	Peugeot, company, manufacture
Q004: How much did Mercury spend on advertising in 1993?	Mercury, spend, advertising, 1993
Q005: What is the name of the managing director of Apricot Computer?	name, managing, director, Apricot, Computer

Формулирование запроса

- Применение правил для переформулирования вопроса
  - к форме подстроки декларативного ответа
  - “когда был придуман лазер” → “лазер был придуман”
- Послать переформулированный запрос информационно-поисковой системе
- Правила (Lin 07)
  - wh-word did A verb B → A verb-ed B
  - Where is A → A is located in

Классификация вопросов

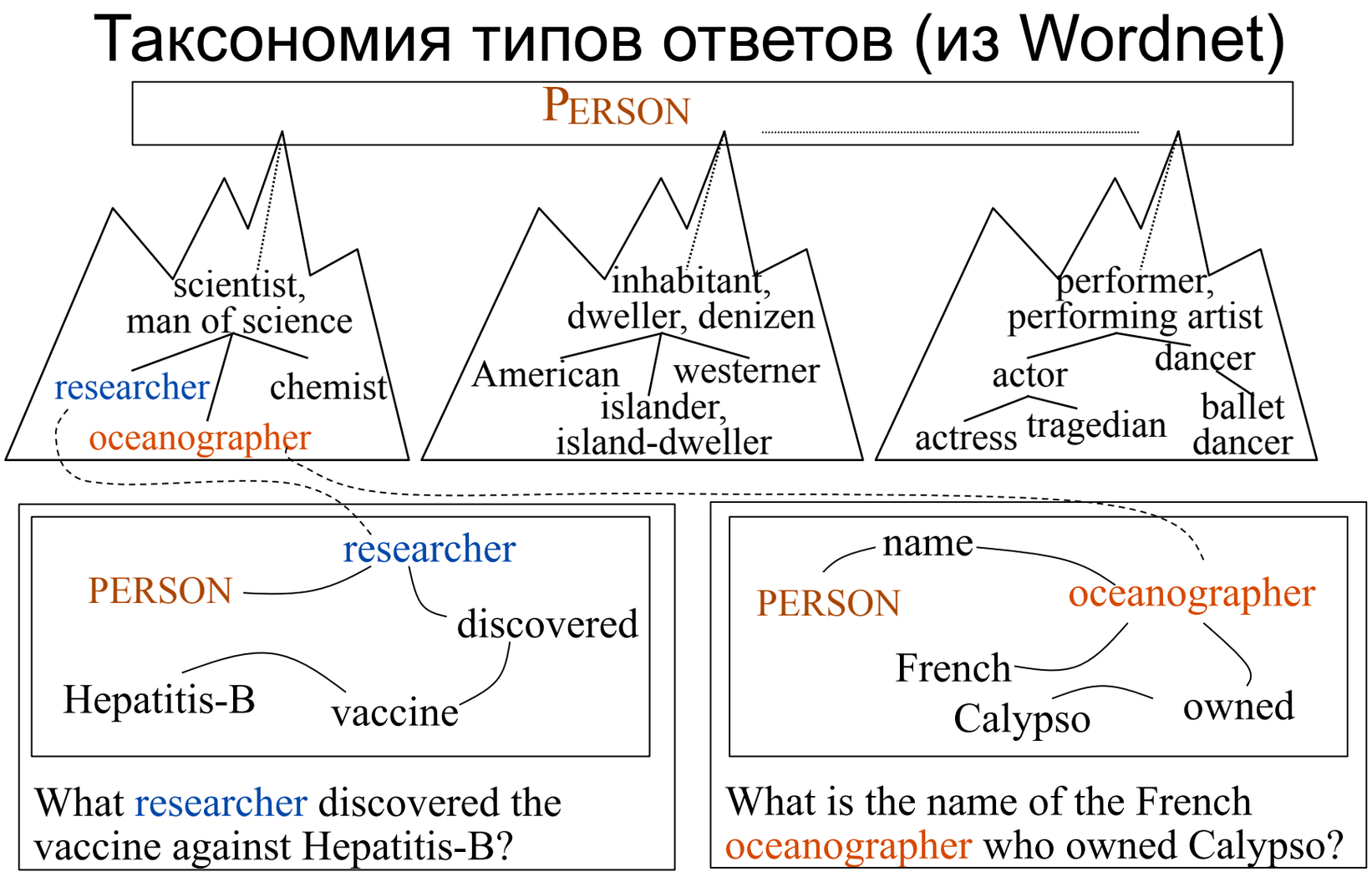
- Классификация вопросов по ожидаемому ответу

Вопрос	Основа вопроса	Тип ответа
Q555: What was the name of Titanic's captain?	What	Person
Q654: What U.S. Government agency registers trademarks?	What	Organization
Q162: What is the capital of Kosovo?	What	City
Q661: How much does one ton of cement cost?	How much	Quantity

Определение типа ответа

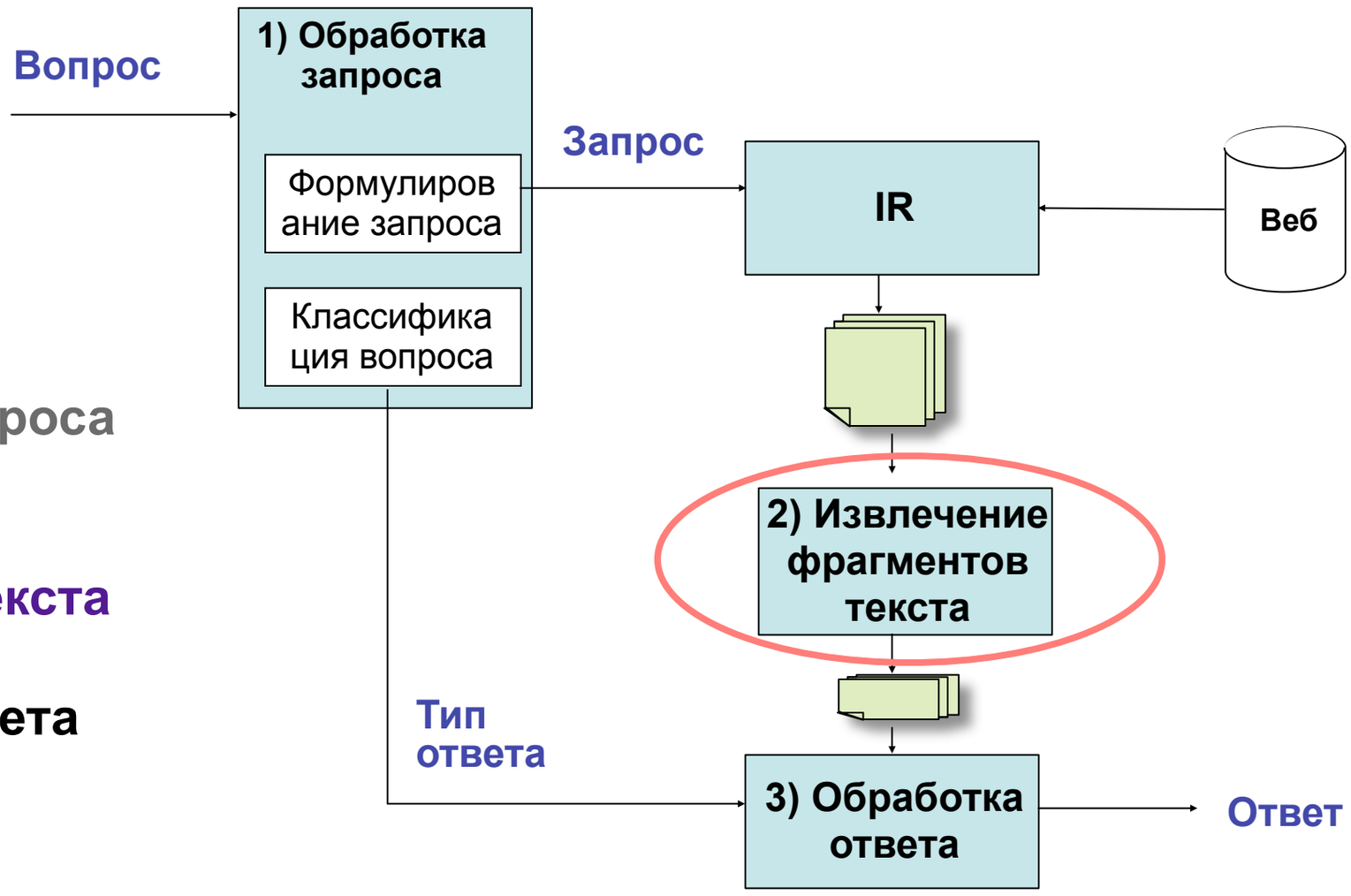
- В некоторых случаях тип ответа можно определить по вопросу
  - Почему → Причина
  - Когда → Дата
- Для многозначных вопросов использовать дополнительные понятия в вопросе
  - What was the name of Titanic's captain?
  - What U.S. Government agency registers trademarks?
  - What is the capital of Kosovo?
- Машинное обучение (если есть размеченный корпус

Определение типов ответов



Типичная архитектура QA-систем

1. **Обработка запроса**
2. **Извлечение фрагментов текста**
3. **Обработка ответа**



Извлечение фрагментов текста

- IR-система возвращает список документов
- Необходимым фрагментом может быть предложение или параграф
- Необходимо выбрать фрагменты, потенциально содержащие ответ

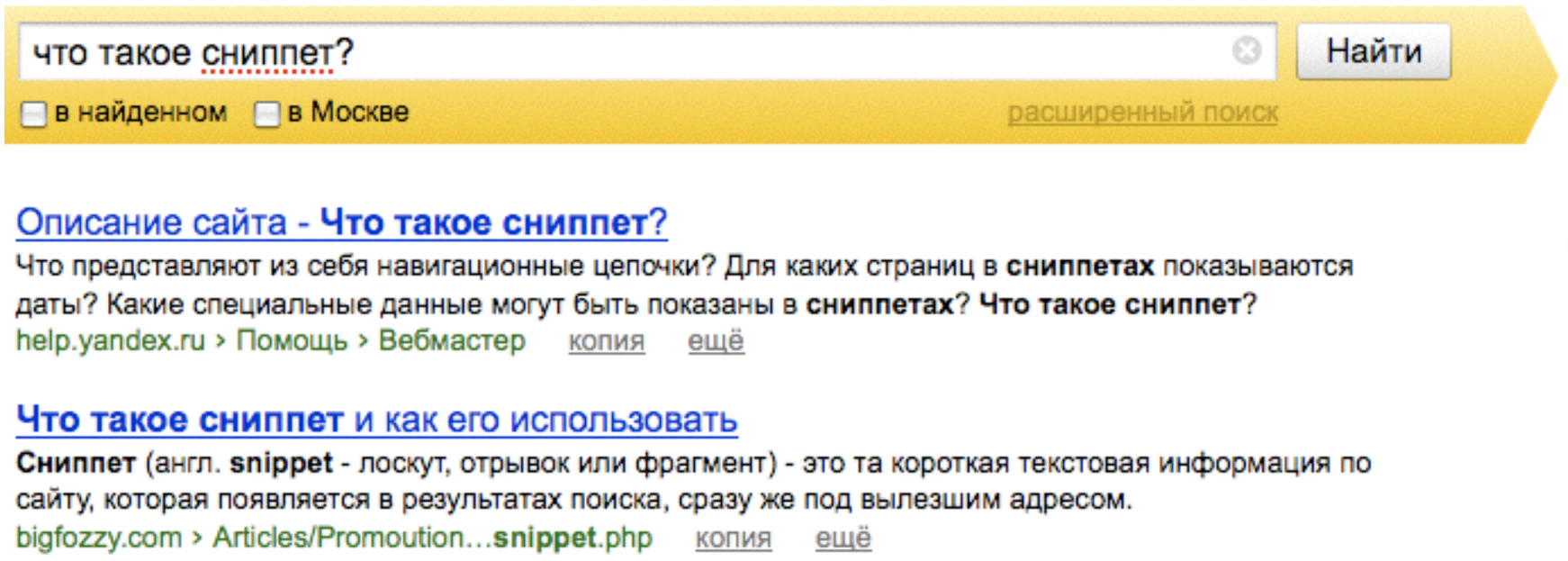
1. Отсеять фрагменты не содержащие ответ
   – распознавание именованных сущностей и классификация ответов
2. Отранжировать оставшиеся фрагменты
   – Правила, составленные вручную
   – Машинное обучение

Извлечение фрагментов текста (ранжирование)

- Признаки
  - Число именованных сущностей правильного типа в фрагменте
  - Число ключевых слов из вопроса в фрагменте
  - Наиболее длинная последовательность ключевых слов запроса в фрагменте
  - Ранг документа (IR), содержащего фрагмент
  - Плотность ключевых слов из вопроса в фрагменте
  - Пересечение N-грамм вопроса и фрагмента

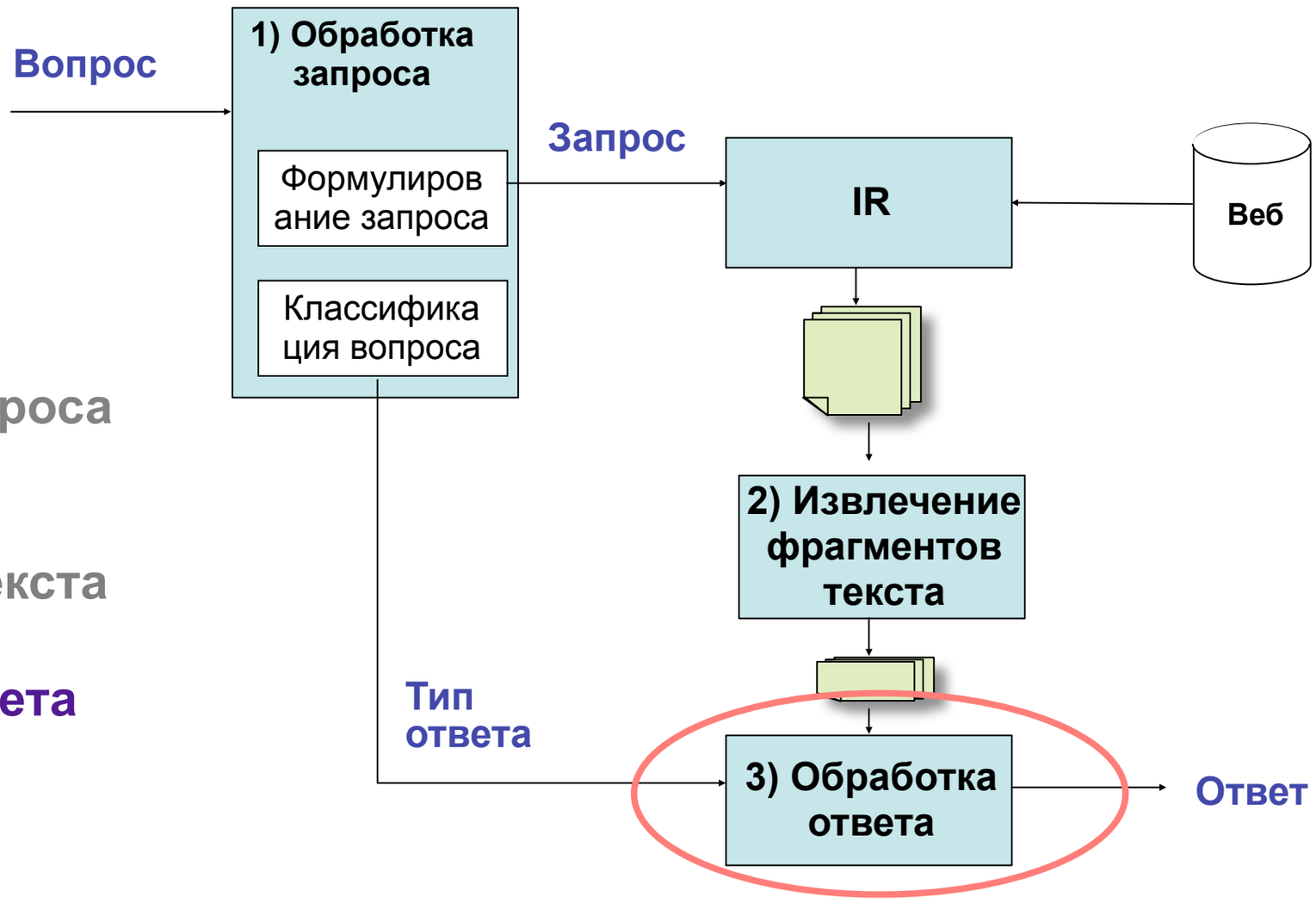
Извлечение фрагментов

- Для извлечения ответа из Веба можно пропустить шаг извлечения фрагмента и использовать сниппеты, возвращаемые информационно-поисковыми системами



Типичная архитектура QA-систем

1. **Обработка запроса**
2. **Извлечение фрагментов текста**
3. **Обработка ответа**



Обработка ответа

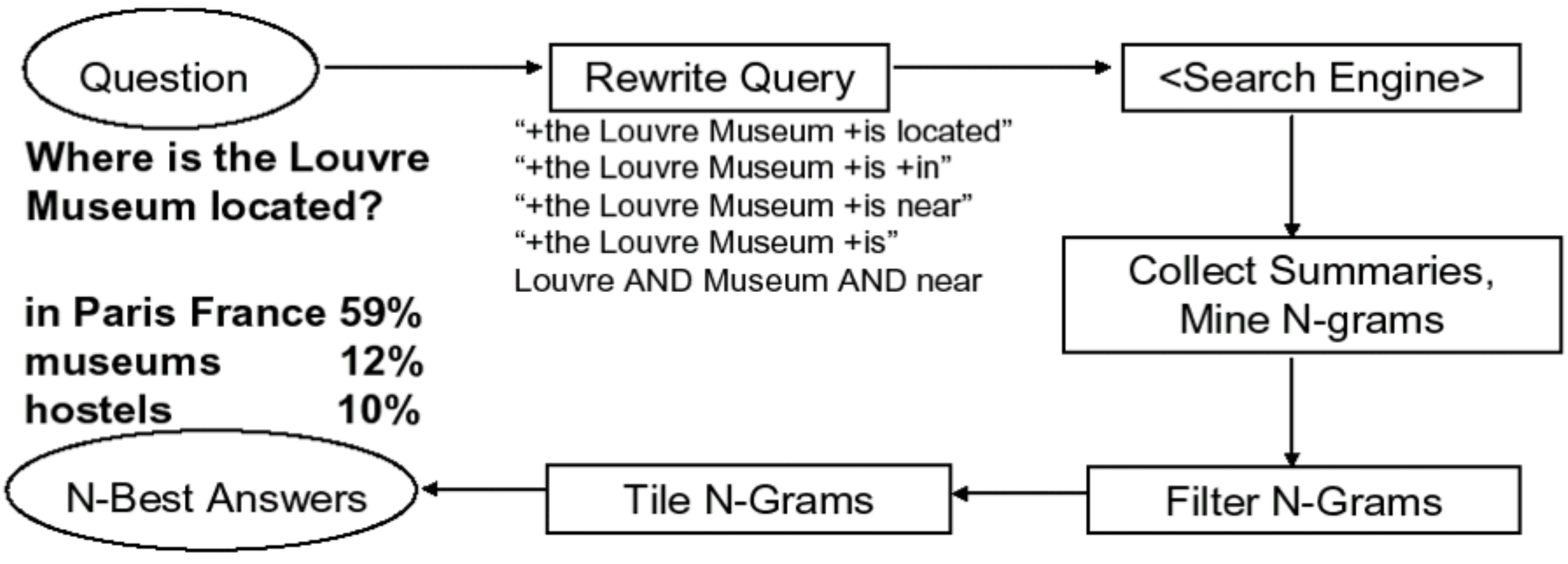
- Извлечение специфичного ответа из фрагмента
- Два основных класса алгоритмов
  - Основанные на шаблонах
  - Сбор ответа из N-грамм (N-gramm tiling)

Алгоритмы на основе шаблонов

- Использование информации о типе в регулярных выражениях
  - Если тип ответа ЧЕЛОВЕК, извлечь именованные сущности ЧЕЛОВЕК из фрагмента
- Некоторые типы ответов (например, определения) не подразумевают конкретного типа именованной сущности в ответе
  - Использовать регулярные выражения (созданные вручную или автоматически)

Pattern	Question	Answer
<AP> such as <QP>	*What is autism?*	*developmental disorders such as autism*

Сбор ответа из N-грамм
Архитектура AskMSR

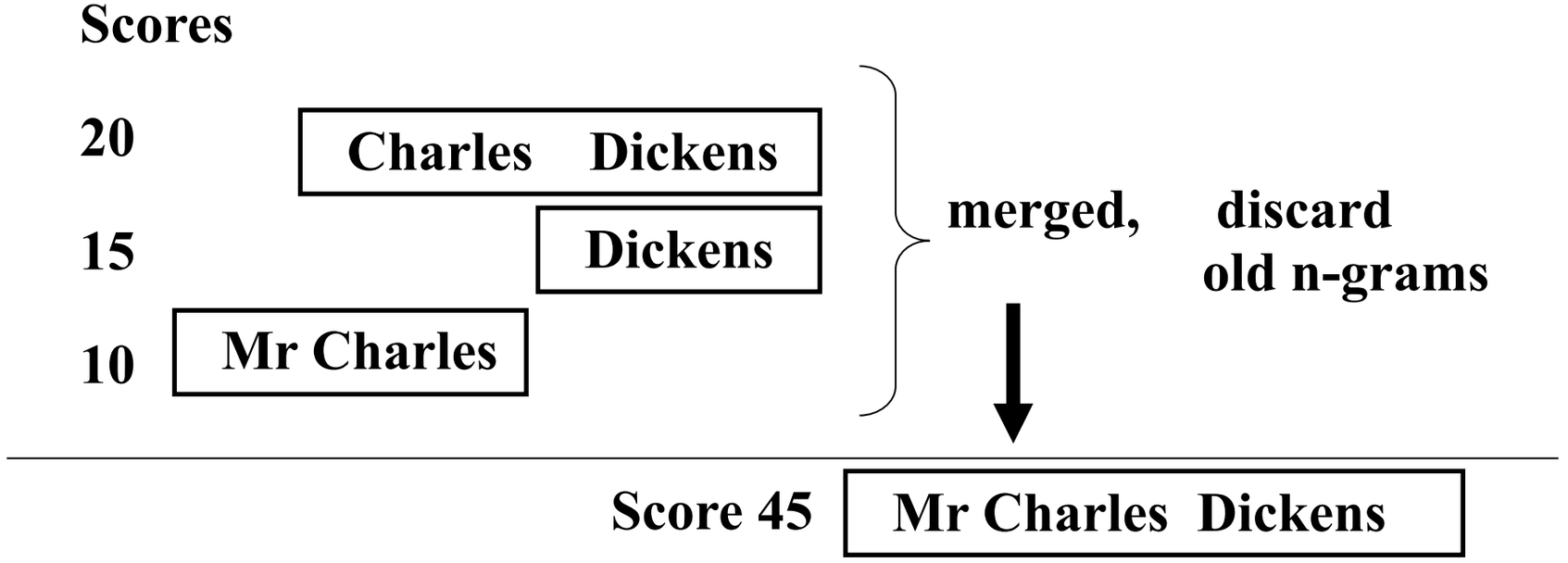


Сбор N-грамм

- Назначить вес N-грамме равный количеству сниппетов, в которых она встретилась
- Пример: “Who created the character of Scrooge?”
  - Dickens 117
  - Christmas Carol 78
  - Charles Dickens 75
  - Disney 72
  - Carl Banks 54
  - A Christmas 41
  - Christmas Carol 45
  - Uncle 31

Фильтрация и сбор ответа

- Заново взвесить N-граммы с учетом типа ответа
- Собрать ответ



Автоматическое реферирование

- Часто ответом на вопрос должен быть текст
- Пример:
  - Кто такой Франсуа Томбалбай?
- Извлечение короткого фрагмента текста является задачей автоматического реферирования

Аннотирование и реферирование

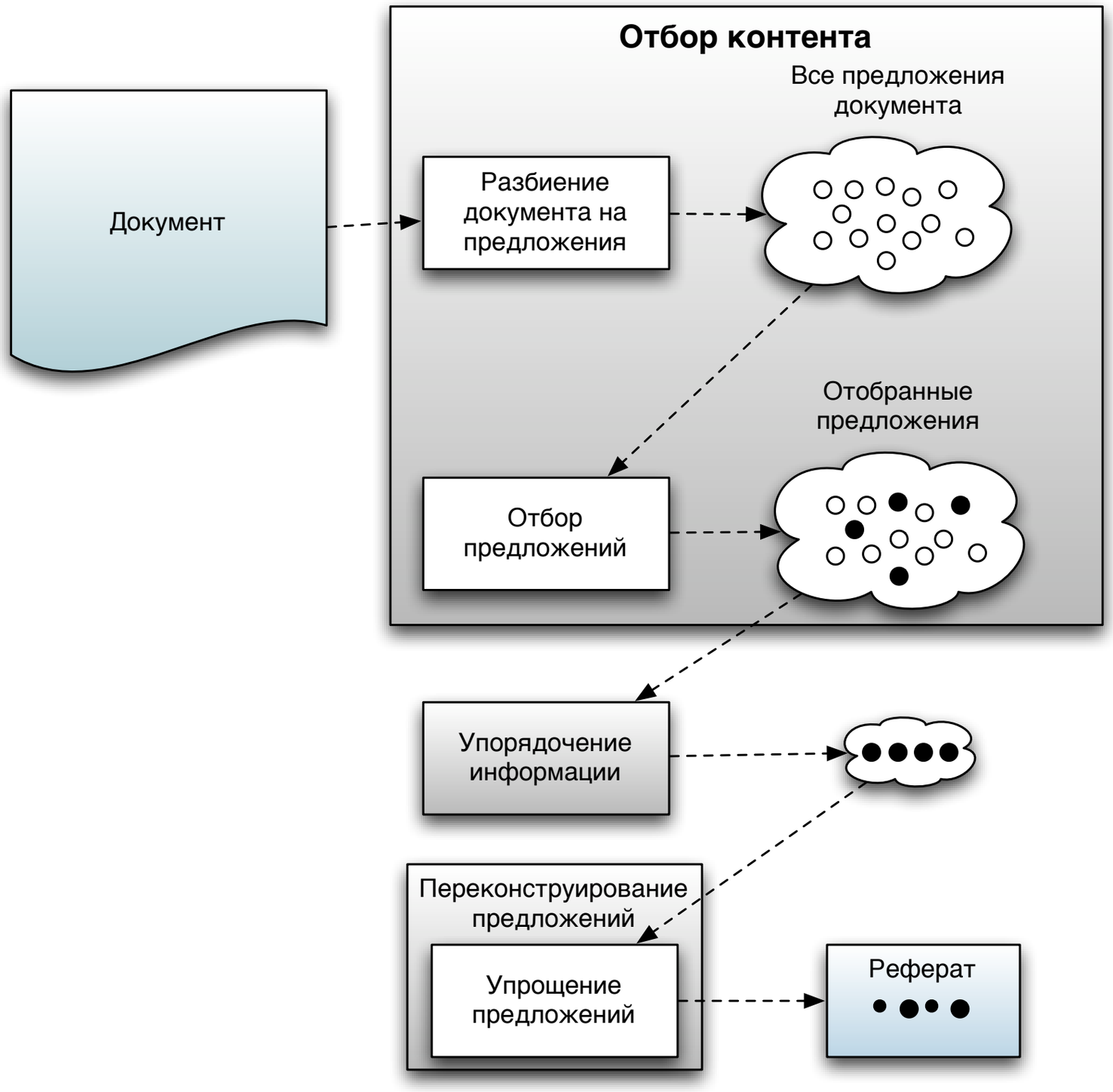
- **Реферат** состоит из частей оригинального текста
- **Аннотация** - главная мысль документа, сформулированная своими словами
  - Более компактная
  - Предполагает генерацию текста

Автоматическое реферирование

Приложения

- Аннотации и рефераты к научным и другим статьям
- Реферированное новостей (несколько документов)
- Создание сниппетов
- Текст для мобильных устройств
- Реферат встречи
- ...

Типичная архитектура



Отбор контента

- **Без учителя**
  - выбор предложений с ключевыми словами (tf-idf, логарифмическое отношение правдоподобия, ...)
  - Центральность
    - пример $centrality(x) = \frac{1}{K} \sum_{y} \text{tf-idf-cos}(x, y)$
- **С учителем**
  - бинарная классификация предложений
  - признаки: позиция, обобщающие фразы (“in summary”, “in conclusion”,...), информативность слов, длина предложения, связность

Упорядочение

- **Для одного документа**
  - –Использовать порядок внутри документа
- Для коллекции документов
  - – более сложные методы
  - кластеризация предложений

Переконструирование предложения

- Упрощение предложений
  - ~~When it arrives sometime new year in new TV sets~~, the V-chip will give parents a ~~new and potentially revolutionary~~ device to block out programs they don’t want their children to see.
- Использование синтаксического разбора и удаление неинформативных частей
  - Zajic et al. 2007, Conroy et al. 2006

Заключение

- Информационный поиск
  - Обработка запроса и документа
  - Извлечение документов
  - Оценка систем
- Вопросно-ответные системы
  - Обработка запроса
  - Извлечение фрагментов текста
  - Обработка ответа
- Системы автоматического реферирования
  - Отбор контента
  - Упорядочение информации
  - Переконструирование предложений

Следующая лекция

- Машинный перевод

Основы обработки текстов

Лекция 10

Машинный перевод

План

- Применение машинного перевода
- Сложности перевода
  - Типология
  - Различия языков
- Классический подход
- Статистический подход
  - Модель зашумленного канала
  - Выравнивание
  - Тренировка моделей
  - Декодирование
  - Методы оценки

Применение машинного перевода

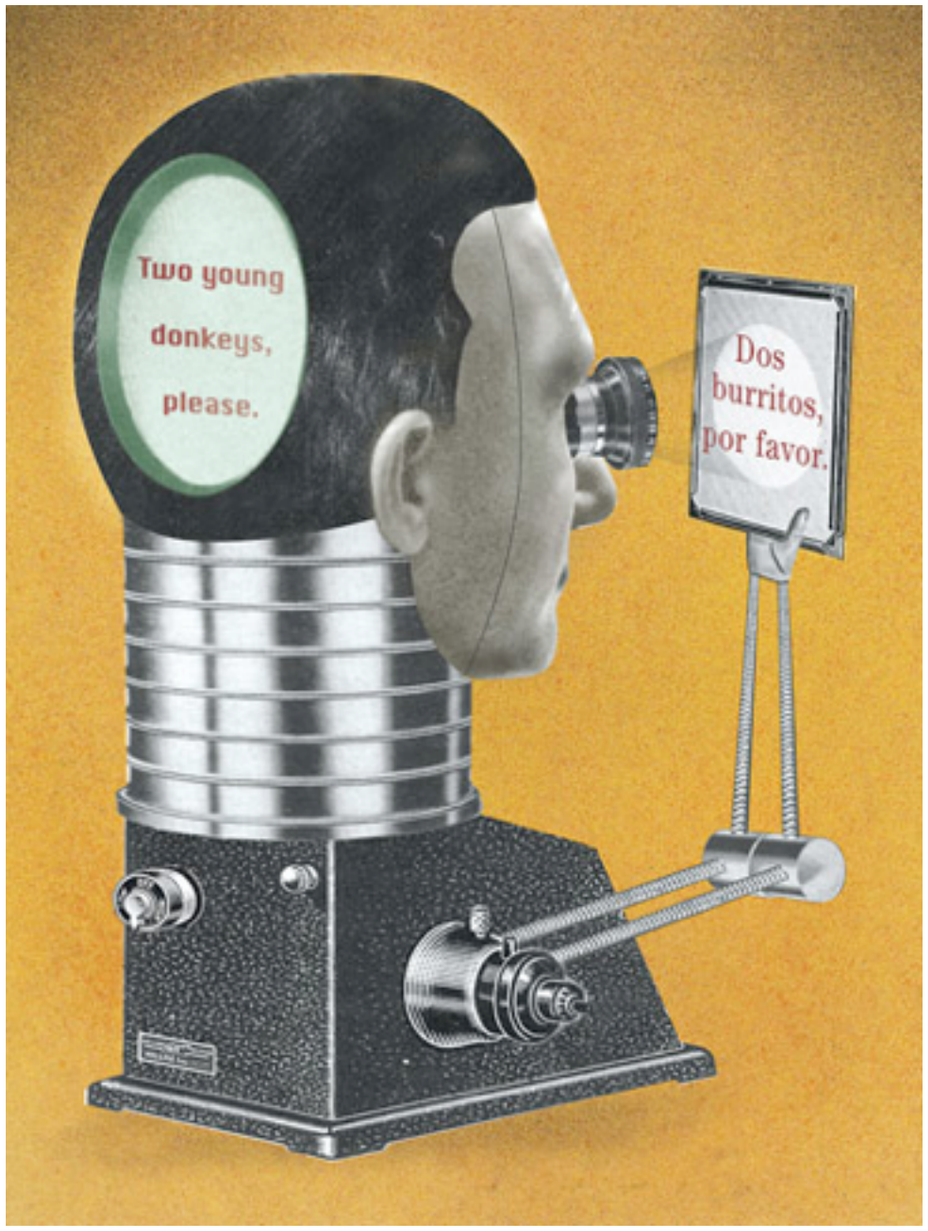
- Задачи, где достаточно грубого перевода
  - Задачи извлечения информации
  - Перевод Веб-страниц
  - e-mail
- Задачи, где результат перевода может быть исправлен
  - Помощь переводчику
- Перевод подмножеств языка
  - FAHQT (Fully Automatic High Quality Translation)

Где машинный перевод недостаточно хорош

- Художественная литература
- Разговорный язык
- Медицинский перевод в больницах
- Звонки в службу спасения

Сложность перевода

- Некоторые аспекты языков схожи, некоторые различны
- Различия в
  - –морфологии
  - –лексике
  - –структуре



Морфология

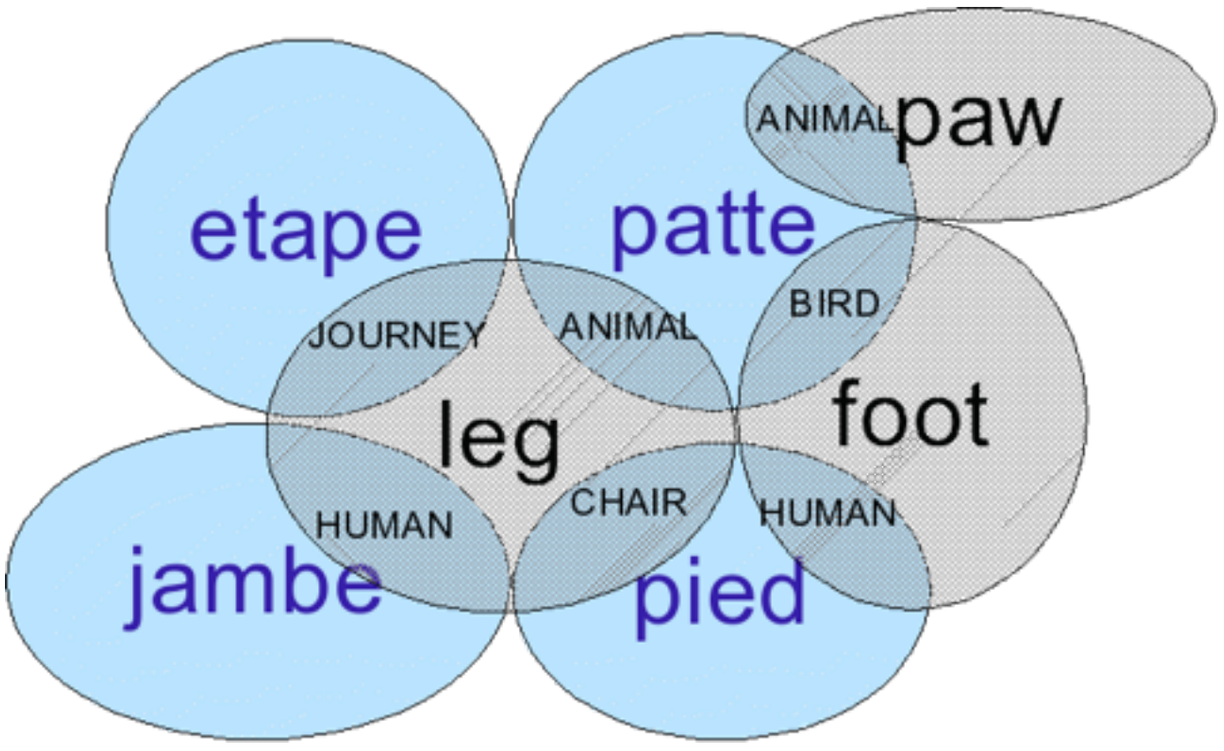
- Морфема
  - минимальная значимая единица языка
  - слово = морфема + морфема + морфема + ...
- Аффиксы
  - Префикс: undo
  - Суффикс: looking
  - Инфикс: hingi (занимать) - humingi (заемщик) (Тагальский язык)
  - Циркумфикс: sagen (сказать) - gesagt (сказал) (Немецкий)

Морфологические различия

- Изолирующие языки
  - –Каждое слово состоит из одной морфемы (Вьетнамский)
- Полисинтетические языки
  - –слово состоит из множества морфем (Чукотский: Тымэйӈылевтпыгтыркын - У меня сильно болит голова)
- Аглютинативные
  - – Морфемы несут определенные значения (Турецкий)
- Флективные
  - –Морфемы имеют несколько значений (Русский: “хороший” - им. падеж, ед. число, муж. род)

Лексические различия

- Семантические особенности:
  - В корейском нет слов брат/сестра, есть старший/младший брат/сестра
  - В чукотском около 20 слов для снега
- Английский vs французский



Синтаксические различия

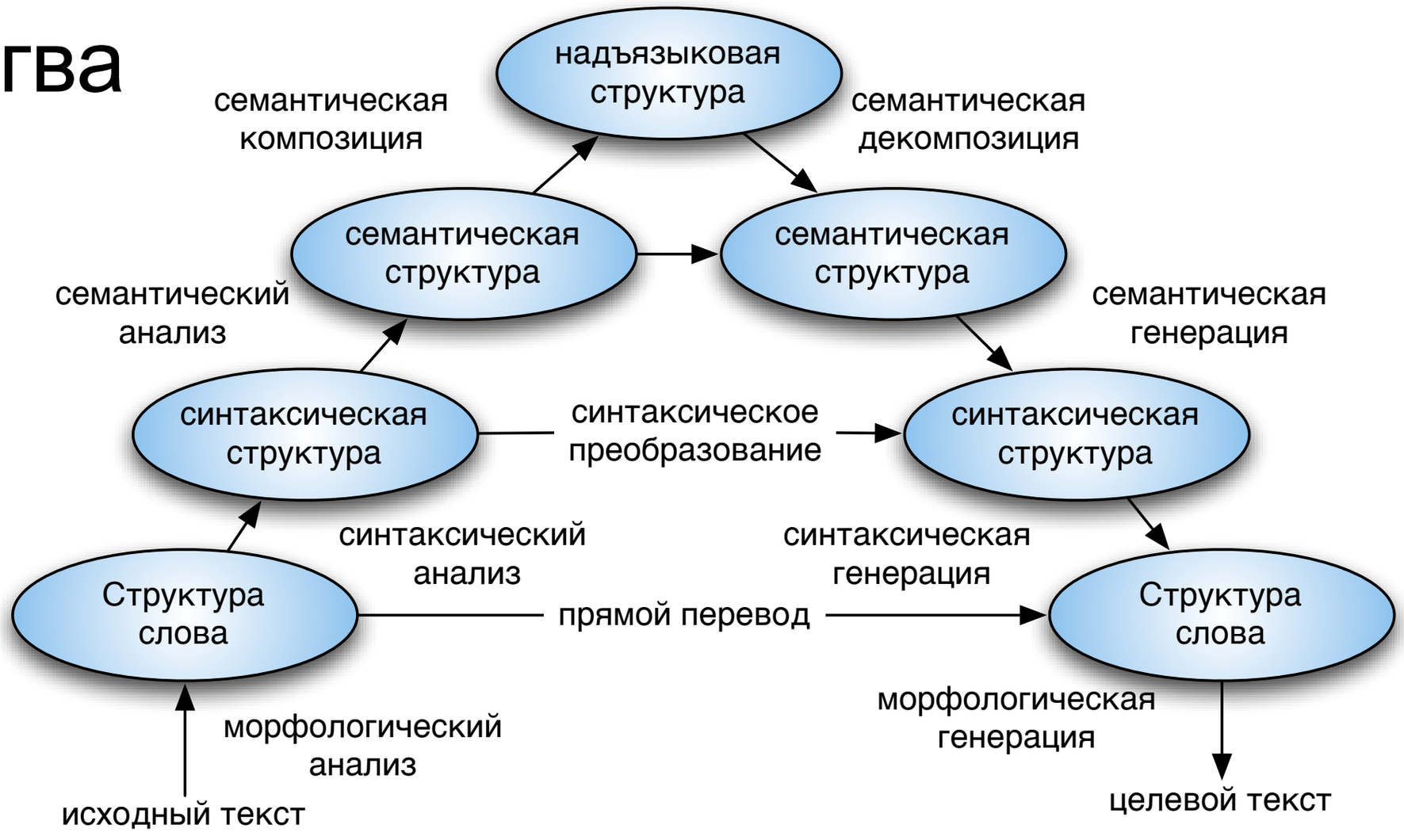
- СГО (Субъект-Глагол-Объект)
  - Английский, Немецкий
  - I am in Moscow
- СОГ
  - Японский, Корейский
  - 저는 모스크바에 있습니다 (Я в Москве нахожусь)
- ГСО
  - Ирландский, классический Арабский

Границы

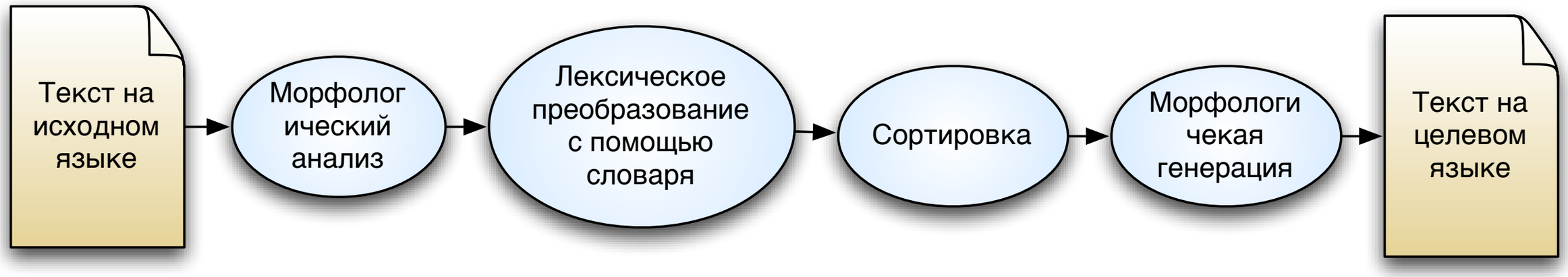
- Языки в которых не выделены границы слов:
  - –Китайский, Японский, Тайский, Въетнамский
- Предложения в некоторых языках больше похожи на параграфы
  - –Китайский, современный Арабский

Классические подходы

- Прямой перевод
- Преобразование
- Интерлингва



Подход 1: Прямой перевод



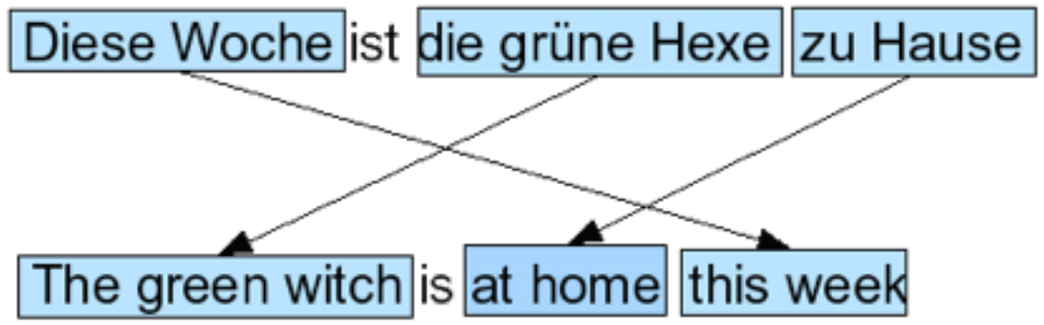
- Последовательный перевод каждого слова
- Не используется никакие структуры кроме морфологии
- После перевода слов, делается сортировка

Пример

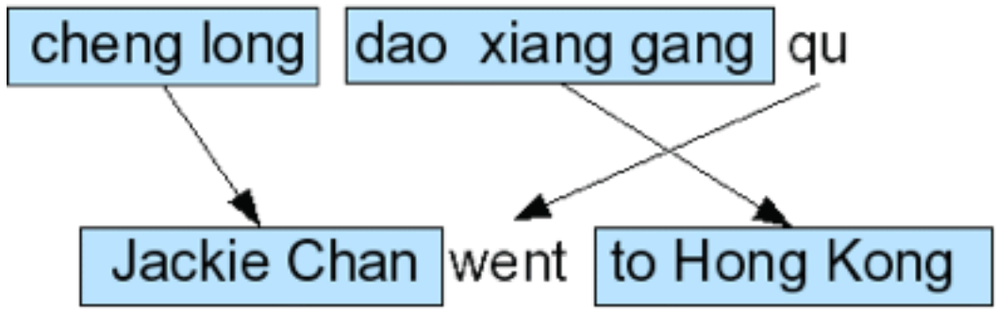
Input:	Mary didn't slap the green witch
After 1: Morphology	Mary DO-PAST not slap the green witch
After 2: Lexical Transfer	Maria PAST no dar una bofetada a la verde bruja
After 3: Local reordering	Maria no dar PAST una bofetada a la bruja verde
After 4: Morphology	Maria no dió una bofetada a la bruja verde

Проблемы

- Сложные перестановки
  - –термины
  - –длинные дистанции
- Немецкий



- Китайский

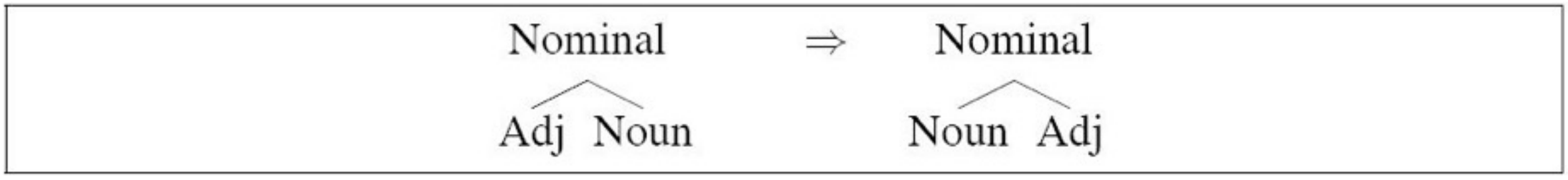


Подход 2: Преобразование

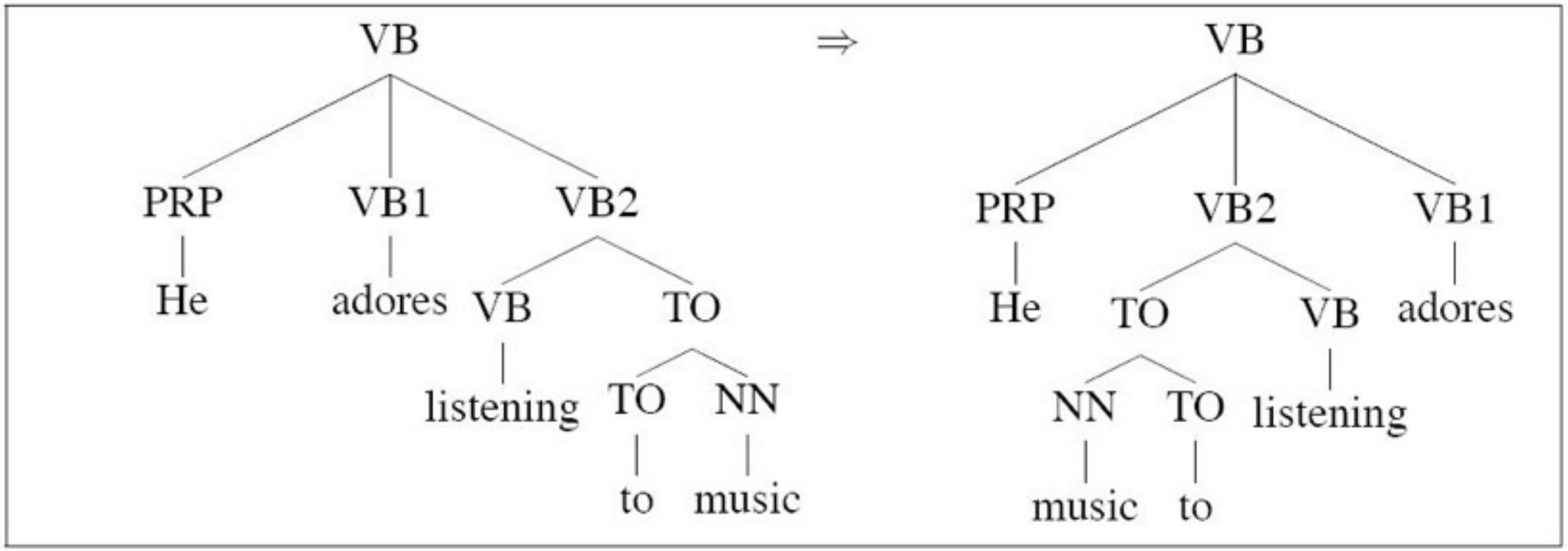
- Применение знаний о различиях в языках
- Шаги
  - Анализ: синтаксический разбор исходного предложения
  - Преобразование: правила преобразования разбора в разбор на целевом языке
  - Генерация предложения на целевом языке

Пример

- Английский: прилагательное существительное
- Французский: существительное прилагательное
- Не всегда
- Правило



Правила преобразования



	English to Spanish:		
1.	NP → Adjective$_1$ Noun$_2$	⇒	NP → Noun$_2$ Adjective$_1$
	Chinese to English:		
2.	VP → PP[+Goal] V	⇒	VP → V PP[+Goal]
	English to Japanese:		
3.	VP → V NP	⇒	VP → NP V
4.	PP → P NP	⇒	PP → NP P
5.	NP → NP$_1$ Rel. Clause$_2$	⇒	NP → Rel. Clause$_2$ NP$_1$

Systran: комбинирование подходов

- Анализ
  - –Морфологический, определение частей речи
  - –Группировка (chunking)
  - –Разбор некоторых зависимостей
- Преобразование
  - –перевод идиом
  - –Разрешение лексической многозначности
  - –назначение предлогов на основе моделей управления глаголов
- Синтез
  - –Применения большого двуязычного словаря
  - –сортировка
  - –морфологическая генерация

Проблемы

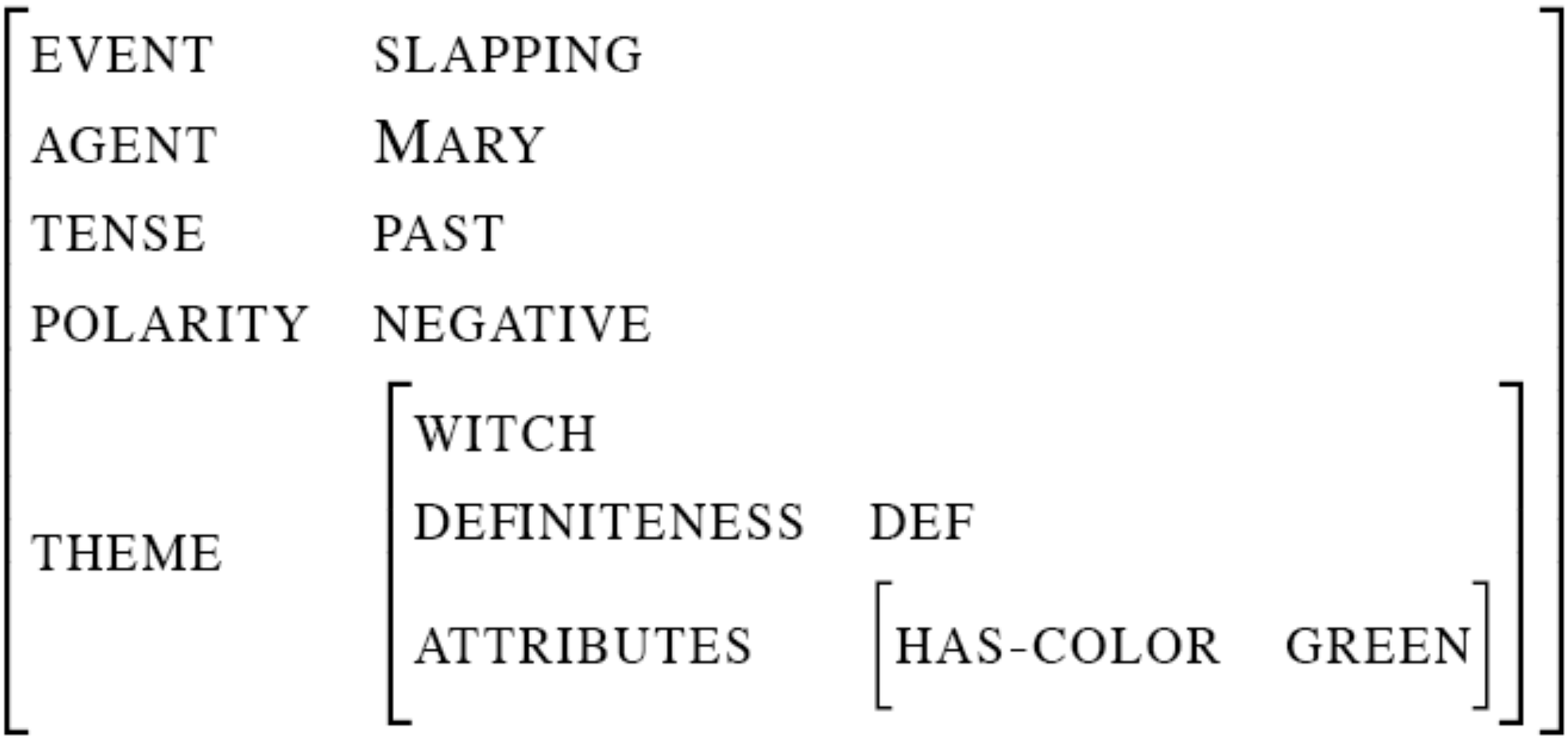
- Грамматика и лексика содержат много специфики
- Трудно сделать и еще труднее поддерживать

Интерлингва

- Пример системы: ABBYY Compreno
- Идея: Вместо использования правил преобразования между языками использовать значение предложения
- Шаги
  - Перевести исходное предложение в представление его значения
  - Сгенерировать целевое предложение из значения

Интерлингва

Mary did not slap the green witch



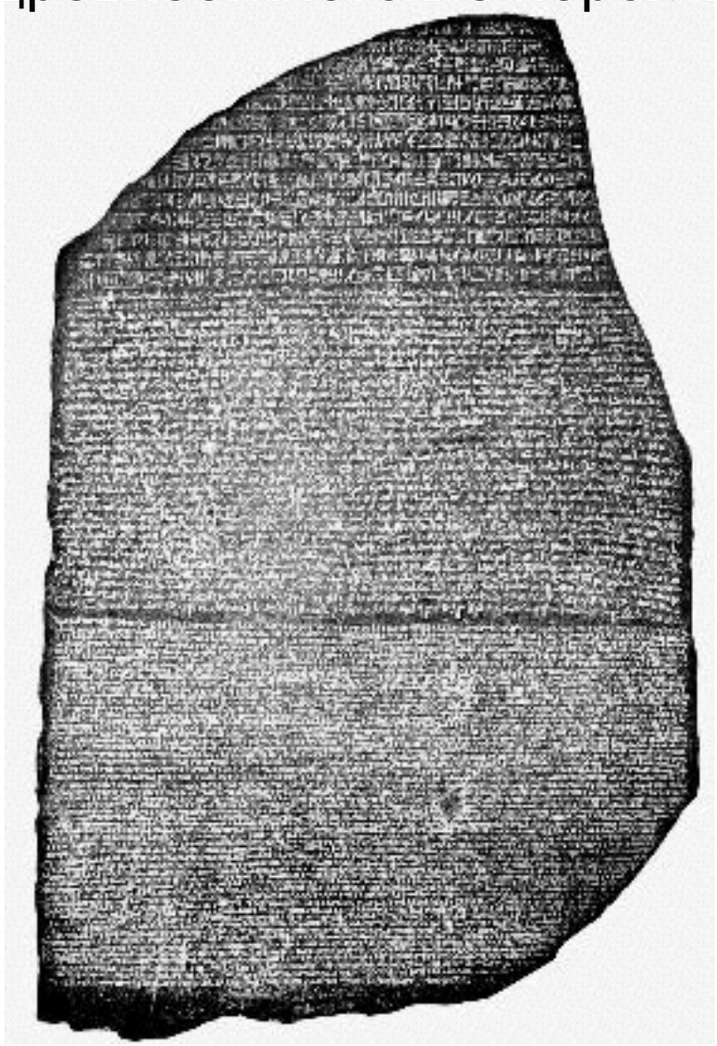
Проблемы

- Разные понятия в языках
  - 20 типов снега в Чукотском
  - Не нужны для англо-русского перевода
- Всесторонний анализ семантики и представление знаний
  - Возможно сделать только для специфичных подмножеств языка

Статистический машинный перевод

- Идеи:
  - Использование параллельных текстов
  - Перевод по фразам
  - Сортировка результата

Розеттский камень:
- древнегреческий
- древнеегипетский
- древнеегипетские иероглифы



Перевод на основе параллельных корпусов

Перевод с Центаврианского на Арктуранский [Knight, 1997]

Перевести: **farok crrrok hihok yorok clok kantok ok-yurp**

1a. ok-voon ororok sprok . 1b. at-voon bichat dat .	7a. lalok farok ororok lalok sprok izok enemok . 7b. wat jjat bichat wat dat vat eneat .
2a. ok-drubel ok-voon anok plok sprok . 2b. at-drubel at-voon pippat rrat dat .	8a. lalok brok anok plok nok . 8b. iat lat pippat rrat nnat .
3a. erok sprok izok hihok ghirok . 3b. totat dat arrat vat hilat .	9a. wiwok nok izok kantok ok-yurp . 9b. totat nnat quat oloat at-yurp .
4a. ok-voon anok drok brok jok . 4b. at-voon krat pippat sat lat .	10a. lalok mok nok yorok ghirok clok . 10b. wat nnat gat mat bat hilat .
5a. wiwok farok izok stok . 5b. totat jjat quat cat .	11a. lalok nok crrrok hihok yorok zanzanok . 11b. wat nnat arrat mat zanzanat .
6a. lalok sprok izok jok stok . 6b. wat dat krat quat cat .	12a. lalok rarok nok izok hihok mok . 12b. wat nnat forat arrat vat gat .

Перевод на основе параллельных корпусов

Перевод с Центаврианского на Арктуранский [Knight, 1997]

Перевести: **farok crrrok hihok yorok clok kantok ok-yurp**

1a. ok-voon ororok sprok . 1b. at-voon bichat dat .	7a. lalok farok ororok lalok sprok izok enemok . 7b. wat jjat bichat wat dat vat eneat .
2a. ok-drubel ok-voon anok plok sprok . 2b. at-drubel at-voon pippat rrat dat .	8a. lalok brok anok plok nok . 8b. iat lat pippat rrat nnat .
3a. erok sprok izok hihok ghirok . 3b. totat dat arrat vat hilat .	9a. wiwok nok izok kantok ok-yurp . 9b. totat nnat quat oloat at-yurp .
4a. ok-voon anok drok brok jok . 4b. at-voon krat pippat sat lat .	10a. lalok mok nok yorok ghirok clok . 10b. wat nnat gat mat bat hilat .
5a. wiwok farok izok stok . 5b. totat jjat quat cat .	11a. lalok nok crrrok hihok yorok zanzanok . 11b. wat nnat arrat mat zanzanat .
6a. lalok sprok izok jok stok . 6b. wat dat krat quat cat .	12a. lalok rarok nok izok hihok mok . 12b. wat nnat forat arrat vat gat .

Перевод на основе параллельных корпусов

Перевод с Центаврианского на Арктуранский [Knight, 1997]

Перевести: **farok crrrok hihok yorok clok kantok ok-yurp**

1a. ok-voon ororok sprok . 1b. at-voon bichat dat .	7a. lalok farok ororok lalok sprok izok enemok . 7b. wat jjat bichat wat dat vat eneat .
2a. ok-drubel ok-voon anok plok sprok . 2b. at-drubel at-voon pippat rrat dat .	8a. lalok brok anok plok nok . 8b. iat lat pippat rrat nnat .
3a. erok sprok izok hihok ghirok . 3b. totat dat arrat vat hilat .	9a. wiwok nok izok kantok ok-yurp . 9b. totat nnat quat oloat at-yurp .
4a. ok-voon anok drok brok jok . 4b. at-voon krat pippat sat lat .	10a. lalok mok nok yorok ghirok clok . 10b. wat nnat gat mat bat hilat .
5a. wiwok farok izok stok . 5b. totat jjat quat cat .	11a. lalok nok crrrok hihok yorok zanzanok . ??? 11b. wat nnat arrat mat zanzanat .
6a. lalok sprok izok jok stok . 6b. wat dat krat quat cat .	12a. lalok rarok nok izok hihok mok . 12b. wat nnat forat arrat vat gat .

Перевод на основе параллельных корпусов

Перевод с Центаврианского на Арктуранский [Knight, 1997]

Перевести: **farok crrrok hihok yorok clok kantok ok-yurp**

1a. ok-voon ororok sprok . 1b. at-voon bichat dat .	7a. lalok farok ororok lalok sprok izok enemok . 7b. wat jjat bichat wat dat vat eneat .
2a. ok-drubel ok-voon anok plok sprok . 2b. at-drubel at-voon pippat rrat dat .	8a. lalok brok anok plok nok . 8b. iat lat pippat rrat nnat .
3a. erok sprok izok hihok ghirok . 3b. totat dat arrat vat hilat .	9a. wiwok nok izok kantok ok-yurp . 9b. totat nnat quat oloat at-yurp .
4a. ok-voon anok drok brok jok . 4b. at-voon krat pippat sat lat .	10a. lalok mok nok yorok ghirok clok . 10b. wat nnat gat mat bat hilat .
5a. wiwok farok izok stok . 5b. totat jjat quat cat .	11a. lalok nok crrrok hihok yorok zanzanok . 11b. wat nnat arrat mat zanzanat .
6a. lalok sprok izok jok stok . 6b. wat dat krat quat cat .	12a. lalok rarok nok izok hihok mok . 12b. wat nnat forat arrat vat gat .

Перевод на основе параллельных корпусов

Перевод с Центаврианского на Арктуранский [Knight, 1997]

Перевести: **farok crrrok hihok yorok clok kantok ok-yurp**

1a. ok-voon ororok sprok . 1b. at-voon bichat dat .	7a. lalok farok ororok lalok sprok izok enemok . 7b. wat jjat bichat wat dat vat eneat .
2a. ok-drubel ok-voon anok plok sprok . 2b. at-drubel at-voon pippat rrat dat .	8a. lalok brok anok plok nok . 8b. iat lat pippat rrat nnat .
3a. erok sprok izok hihok ghirok . 3b. totat dat arrat vat hilat .	9a. wiwok nok izok kantok ok-yurp . 9b. totat nnat quat oloat at-yurp .
4a. ok-voon anok drok brok jok . 4b. at-voon krat pippat sat lat .	10a. lalok mok nok yorok ghirok clok . 10b. wat nnat gat mat bat hilat .
5a. wiwok farok izok stok . 5b. totat jjat quat cat .	11a. lalok nok crrrok hihok yorok zanzanok . 11b. wat nnat arrat mat zanzanat .
6a. lalok sprok izok jok stok . 6b. wat dat krat quat cat .	12a. lalok rarok nok izok hihok mok . 12b. wat nnat forat arrat vat gat .

Перевод на основе параллельных корпусов

Перевод с Центаврианского на Арктуранский [Knight, 1997]

Перевести: **farok crrrok hihok yorok clok kantok ok-yurp**

1a. ok-voon ororok sprok . 1b. at-voon bichat dat .	7a. lalok farok ororok lalok sprok izok enemok . 7b. wat jjat bichat wat dat vat eneat .
2a. ok-drubel ok-voon anok plok sprok . 2b. at-drubel at-voon pippat rrat dat .	8a. lalok brok anok plok nok . 8b. iat lat pippat rrat nnat .
3a. erok sprok izok hihok ghirok . 3b. totat dat arrat vat hilat .	9a. wiwok nok izok kantok ok-yurp . 9b. totat nnat quat oloat at-yurp .
4a. ok-voon anok drok brok jok . 4b. at-voon krat pippat sat lat .	10a. lalok mok nok yorok ghirok clok . 10b. wat nnat gat mat bat hilat .
5a. wiwok farok izok stok . 5b. totat jjat quat cat .	11a. lalok nok crrrok hihok yorok zanzanok . 11b. wat nnat arrat mat zanzanat .
6a. lalok sprok izok jok stok . 6b. wat dat krat quat cat .	12a. lalok rarok nok izok hihok mok . 12b. wat nnat forat arrat vat gat .

Перевод на основе параллельных корпусов

Перевод с Центаврианского на Арктуранский [Knight, 1997]

Перевести: **farok crrrok hihok yorok clok kantok ok-yurp**

1a. ok-voon ororok sprok . 1b. at-voon bichat dat .	7a. lalok farok ororok lalok sprok izok enemok . 7b. wat jjat bichat wat dat vat eneat .
2a. ok-drubel ok-voon anok plok sprok . 2b. at-drubel at-voon pippat rrat dat .	8a. lalok brok anok plok nok . 8b. iat lat pippat rrat nnat .
3a. erok sprok izok hihok ghirok . 3b. totat dat arrat vat hilat .	9a. wiwok nok izok kantok ok-yurp . 9b. totat nnat quat oloat at-yurp .
4a. ok-voon anok drok brok jok . 4b. at-voon krat pippat sat lat .	10a. lalok mok nok yorok ghirok clok . 10b. wat nnat gat mat bat hilat . ???
5a. wiwok farok izok stok . 5b. totat jjat quat cat .	11a. lalok nok crrrok hihok yorok zanzanok . 11b. wat nnat arrat mat zanzanat .
6a. lalok sprok izok jok stok . 6b. wat dat krat quat cat .	12a. lalok rarok nok izok hihok mok . 12b. wat nnat forat arrat vat gat .

Перевод на основе параллельных корпусов

Перевод с Центаврианского на Арктуранский [Knight, 1997]

Перевести: **farok crrrok hihok yorok clok kantok ok-yurp**

1a. ok-voon ororok sprok . 1b. at-voon bichat dat .	7a. lalok farok ororok lalok sprok izok enemok . 7b. wat jjat bichat wat dat vat eneat .
2a. ok-drubel ok-voon anok plok sprok . 2b. at-drubel at-voon pippat rrat dat .	8a. lalok brok anok plok nok . 8b. iat lat pippat rrat nnat .
3a. erok sprok izok hihok ghirok . 3b. totat dat arrat vat hilat .	9a. wiwok nok izok kantok ok-yurp . 9b. totat nnat quat oloat at-yurp .
4a. ok-voon anok drok brok jok . 4b. at-voon krat pippat sat lat .	10a. lalok mok nok yorok ghirok clok . 10b. wat nnat gat mat bat hilat .
5a. wiwok farok izok stok . 5b. totat jjat quat cat .	11a. lalok nok crrrok hihok yorok zanzanok . 11b. wat nnat arrat mat zanzanat .
6a. lalok sprok izok jok stok . 6b. wat dat krat quat cat .	12a. lalok rarok nok izok hihok mok . 12b. wat nnat forat arrat vat gat .

Перевод на основе параллельных корпусов

Перевод с Центаврианского на Арктуранский [Knight, 1997]

Перевести: **farok crrrok hihok yorok clok kantok ok-yurp**

1a. ok-voon ororok sprok . 1b. at-voon bichat dat .	7a. lalok farok ororok lalok sprok izok enemok . 7b. wat jjat bichat wat dat vat eneat .
2a. ok-drubel ok-voon anok plok sprok . 2b. at-drubel at-voon pippat rrat dat .	8a. lalok brok anok plok nok . 8b. iat lat pippat rrat nnat .
3a. erok sprok izok hihok ghirok . 3b. totat dat arrat vat hilat .	9a. wiwok nok izok kantok ok-yurp . 9b. totat nnat quat oloat at-yurp .
4a. ok-voon anok drok brok jok . 4b. at-voon krat pippat sat lat .	10a. lalok mok nok yorok ghirok clok . 10b. wat nnat gat mat bat hilat .
5a. wiwok farok izok stok . 5b. totat jjat quat cat .	11a. lalok nok crrrok hihok yorok zanzanok . 11b. wat nnat arrat mat zanzanat .
6a. lalok sprok izok jok stok . 6b. wat dat krat quat cat .	12a. lalok rarok nok izok hihok mok . 12b. wat nnat forat arrat vat gat .

методом исключения

Перевод на основе параллельных корпусов

Перевод с Центаврианского на Арктуранский [Knight, 1997]

Перевести: **farok crrrok hihok yorok clok kantok ok-yurp**

1a. ok-voon ororok sprok . 1b. at-voon bichat dat .	7a. lalok farok ororok lalok sprok izok enemok . 7b. wat jjat bichat wat dat vat eneat .
2a. ok-drubel ok-voon anok plok sprok . 2b. at-drubel at-voon pippat rrat dat .	8a. lalok brok anok plok nok . 8b. iat lat pippat rrat nnat .
3a. erok sprok izok hihok ghirok . 3b. totat dat arrat vat hilat .	9a. wiwok nok izok kantok ok-yurp . 9b. totat nnat quat oloat at-yurp .
4a. ok-voon anok drok brok jok . 4b. at-voon krat pippat sat lat .	10a. lalok mok nok yorok ghirok clok . 10b. wat nnat gat mat bat hilat .
5a. wiwok farok izok stok . 5b. totat jjat quat cat .	11a. lalok nok crrrok hihok yorok zanzanok . 11b. wat nnat arrat mat zanzanat .
6a. lalok sprok izok jok stok . 6b. wat dat krat quat cat .	12a. lalok rarok nok izok hihok mok . 12b. wat nnat forat arrat vat gat .

Родственное слово?

Перевод на основе параллельных корпусов

Перевод с Центаврианского на Арктуранский [Knight, 1997]

Задание: упорядочить: {jjat, arrat, mat, bat, oloat, at-yurp}

1a. ok-voon ororok sprok . 1b. at-voon bichat dat .	7a. lalok farok ororok lalok sprok izok enemok . 7b. wat jjat bichat wat dat vat eneat .
2a. ok-drubel ok-voon anok plok sprok . 2b. at-drubel at-voon pippat rrat dat .	8a. lalok brok anok plok nok . 8b. iat lat pippat rrat nnat .
3a. erok sprok izok hihok ghirok . 3b. totat dat arrat vat hilat .	9a. wiwok nok izok kantok ok-yurp . 9b. totat nnat quat oloat at-yurp .
4a. ok-voon anok drok brok jok . 4b. at-voon krat pippat sat lat .	10a. lalok mok nok yorok ghirok clok . 10b. wat nnat gat mat bat hilat .
5a. wiwok farok izok stok . 5b. totat jjat quat cat .	11a. lalok nok crrrok hihok yorok zanzanok . 11b. wat nnat arrat mat zanzanat .
6a. lalok sprok izok jok stok . 6b. wat dat krat quat cat .	12a. lalok rarok nok izok hihok mok . 12b. wat nnat forat arrat vat gat .

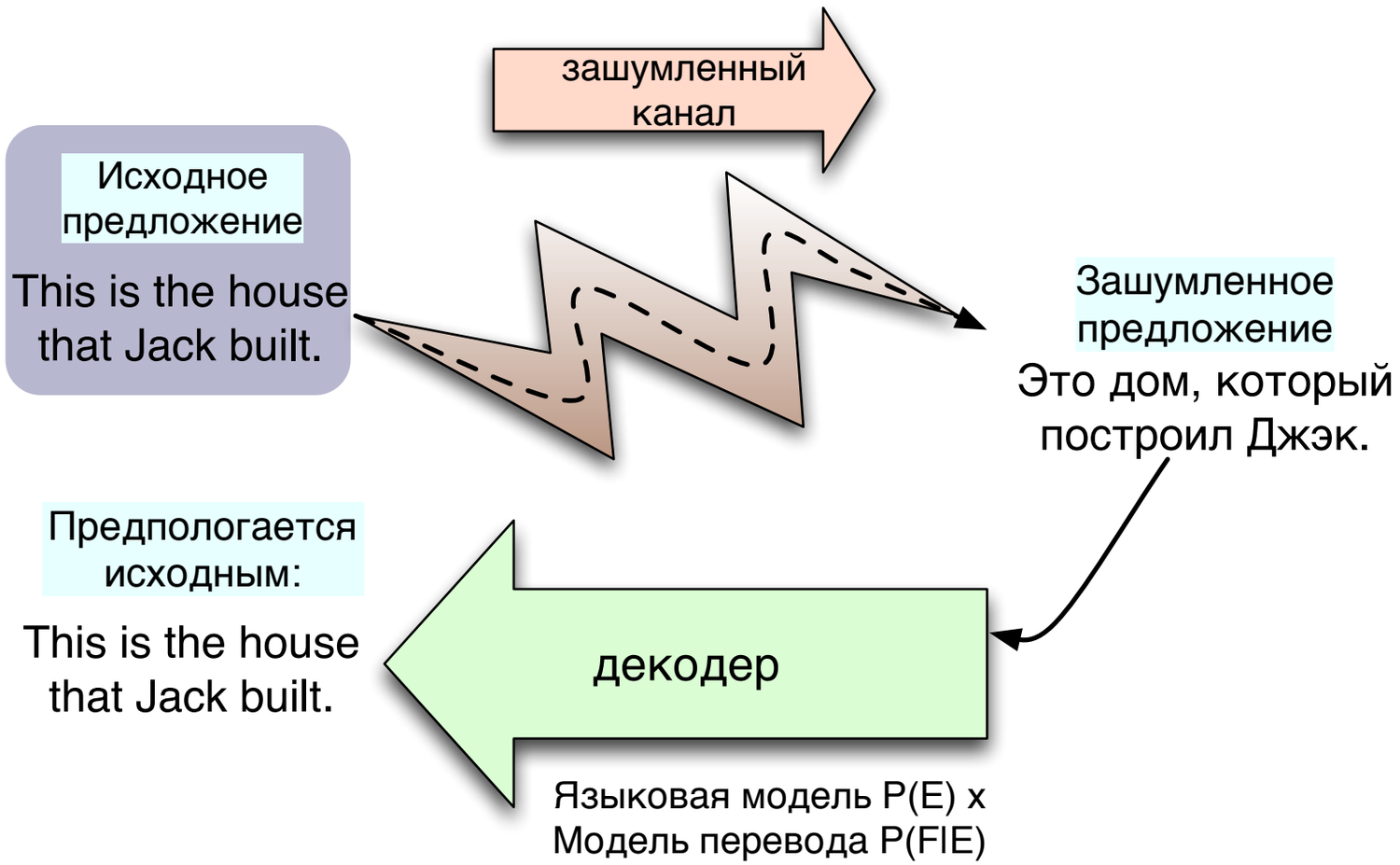
Перевод на основе параллельных корпусов

В действительности это англо-испанский перевод

Clients do not sell pharmaceuticals in Europe => Clientes no venden medicinas en Europa

1a. Garcia and associates . 1b. Garcia y asociados .	7a. the clients and the associates are enemies . 7b. los clients y los asociados son enemigos .
2a. Carlos Garcia has three associates . 2b. Carlos Garcia tiene tres asociados .	8a. the company has three groups . 8b. la empresa tiene tres grupos .
3a. his associates are not strong . 3b. sus asociados no son fuertes .	9a. its groups are in Europe . 9b. sus grupos estan en Europa .
4a. Garcia has a company also . 4b. Garcia tambien tiene una empresa .	10a. the modern groups sell strong pharmaceuticals . 10b. los grupos modernos venden medicinas fuertes .
5a. its clients are angry . 5b. sus clientes estan enfadados .	11a. the groups do not sell zenzanine . 11b. los grupos no venden zanzanina .
6a. the associates are also angry . 6b. los asociados tambien estan enfadados .	12a. the small groups are not modern . 12b. los grupos pequenos no son modernos .

Модель зашумленного канала



- Байесовская модель

$$\hat{E} = \underset{E \in English}{\arg\max}\ P(F|E)P(E)$$

Модель перевода

Языковая модель

Машинный перевод

- Языковая модель
  - –N-граммы
  - –СКС грамматики
- Модель перевода
- Декодер

Модель перевода на основе фраз

- P(F|E)
- Разбиваем E на фразы
- Переводим каждую фразу из E во фразу на другом языке, запоминая **вероятность перевода**
- Находим наиболее вероятную последовательность фраз F

Вероятность перевода

- Пусть есть параллельные тексты с указанием соответствия между фразами в E и F (см. далее).
- Тогда вероятность перевода можно оценить на основе метода максимального правдоподобия

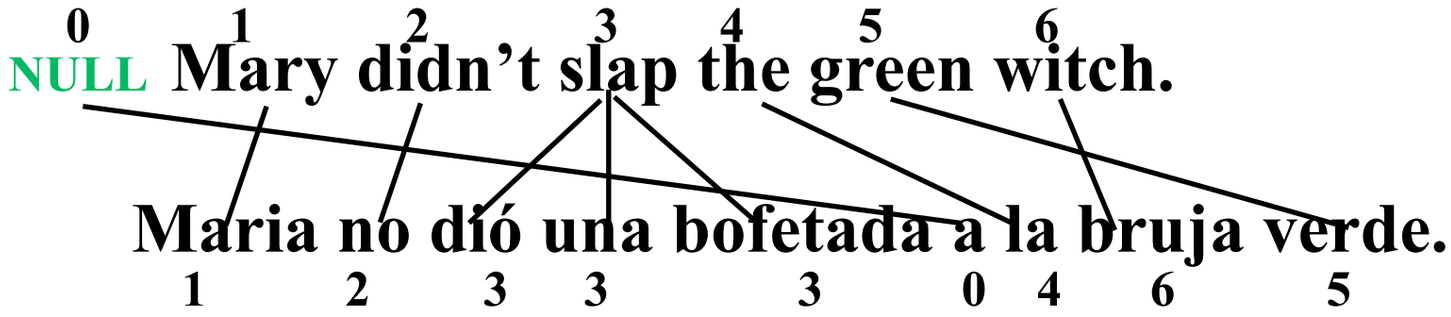
$$\phi(\overline{f}, \overline{e}) = \frac{count(\overline{f}, \overline{e})}{\sum_{\overline{f}}(\overline{f}, \overline{e})}$$

Выравнивание слов

- Сначала выравнивают слова
- Вход: пары предложение-перевод

Выравнивание один ко многим

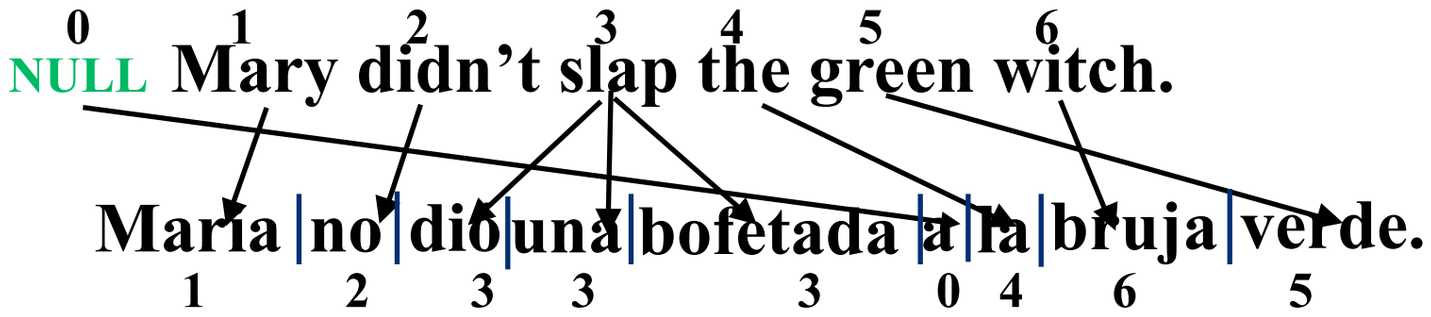
- Для простоты предположим что
  - слово из F соответствует одному слову в E
  - но слово из E может соответствовать нескольким словам в F
- Некоторые слова в F могут соответствовать элементу NULL в E
- Тогда выравнивание можно задавать вектором



Модель IBM Model 1

- Первая самая простая модель предложенная в основополагающей статье [Brown et. al. 1993]
- Предлагает простую генерирующую модель для получения F из E=$e_1, e_2, \ldots e_I$
  - Выбрать длину J предложения F=$f_1, f_2, \ldots f_J$
  - Выбрать выравнивание $A=a_1, a_2, \ldots a_J$
  - Получить F из E

Пример


0 1 2 3 4 5 6
NULL Mary didn't slap the green witch.
Maria no dió una bofetada a la bruja verde.
1 2 3 3 3 0 4 6 5

Подсчет P(F|E)

- Обозначения
  - –Длина предложения на языке E равна I
  - –Длина предложения на языке F равна J
  - –Вероятность длины предложения в F равна P(J|E)
- Model 1: Предположим что все выравнивания A равновероятны (их $(I+1)^J$)
- Тогда условная вероятность (выравниваний):

$$P(A \mid E) = P(A \mid E, J)P(J \mid E) = \frac{P(J \mid E)}{(I+1)^J}$$

Подсчет P(F|E)

- Пусть $t(f_x,e_y)$ вероятность перевода слова e_y в слово f_x
- Определим **P(F|E)**

$$P(F \mid E) = \sum_A P(F \mid E, A)P(A \mid E) = \sum_A \frac{P(J \mid E)}{(I+1)^J} \prod_{j=1}^{J} t(f_j, e_{a_j})$$

Декодирование

- Цель: найти наиболее вероятное выравнивание

$$\hat{A} = \underset{\mathrm{A}}{\operatorname{argmax}}\, P(F, A \mid E)$$
$$= \underset{A}{\operatorname{argmax}}\, \frac{P(J \mid E)}{(I+1)^J} \prod_{j=1}^{J} t(f_j, e_{a_j})$$
$$= \underset{A}{\operatorname{argmax}} \prod_{j=1}^{J} t(f_j, e_{a_j})$$

- Так как различные переводы для каждой позиции j независимы, максимум произведения достигается при максимуме каждого термина

$$a_j = \underset{0 \le i \le I}{\operatorname{argmax}}\, t(f_j, e_i) \quad 1 \le j \le J$$

Тренировка моделей выравнивания

- Если есть выровненный вручную корпус, то можно оценить параметры модели IBM 1 через метод максимального правдоподобия
- Часто такого корпуса нет. В этом случае применяют EM-алгоритм

EM-алгоритм для выравнивания

- Выбираем начальные параметры
- Пока не сойдется выполняем:
  - E-шаг: Вычисляем вероятность всех выравниваний с помощью текущей модели
  - M-шаг: Используем эти вероятности для переоценки значений всех параметров модели

Для сокращения времени работы используется алгоритм динамического программирования

Пример

Тренировочный корпус

green house
casa verde

the house
la casa

Вероятности перевода

	verde	casa	la
green	1/3	1/3	1/3
house	1/3	1/3	1/3
the	1/3	1/3	1/3

Предполагаем начальные вероятности равными

Вычисляем вероятности выравнивания P(A, F | E)

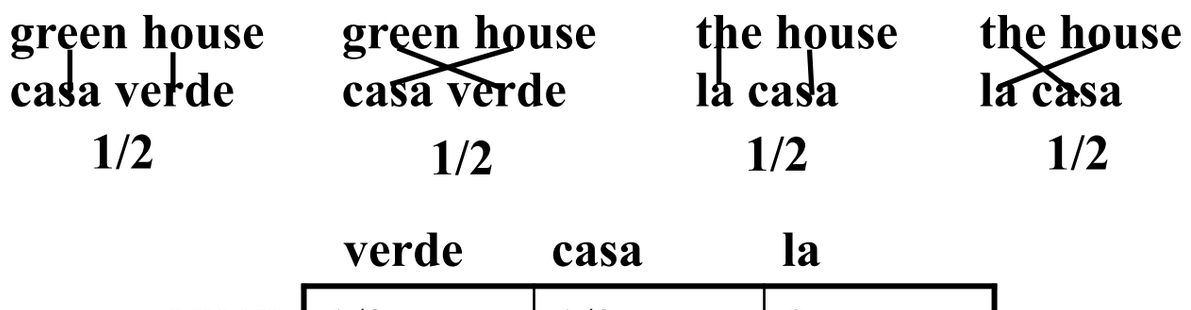
green house **casa verde**	**green house** **casa verde**	**the house** **la casa**	**the house** **la casa**
1/3 X 1/3 = 1/9	**1/3 X 1/3 = 1/9**	**1/3 X 1/3 = 1/9**	**1/3 X 1/3 = 1/9**

Нормализуем P(A | F, E)

$\frac{1/9}{2/9} = \frac{1}{2}$ $\frac{1/9}{2/9} = \frac{1}{2}$ $\frac{1/9}{2/9} = \frac{1}{2}$ $\frac{1/9}{2/9} = \frac{1}{2}$

$$P(A|E,F) = \frac{P(A,F|E)}{\sum_A P(A,F|E)}$$

Пример



Считаем веса переводов

	verde	casa	la
green	1/2	1/2	0
house	1/2	1/2 + 1/2	1/2
the	0	1/2	1/2

Нормализуем и получаем P(f | e)

	verde	casa	la
green	1/2	1/2	0
house	1/4	1/2	1/4
the	0	1/2	1/2

Пример

Вероятности перевода

	verde	**casa**	**la**
green	1/2	1/2	0
house	1/4	1/2	1/4
the	0	1/2	1/2

Пересчитываем вероятности выравнивания P(A, F | E)

green house **casa verde**	**green house** **casa verde**	**the house** **la casa**	**the house** **la casa**
1/2 X 1/4=1/8	**1/2 X 1/2=1/4**	**1/2 X 1/2=1/4**	**1/2 X 1/4=1/8**

$$P(A, F|E) = \prod_{j=1}^{J} t(f_j|e_{a_j})$$

Нормализуем и получаем P(A | F, E)

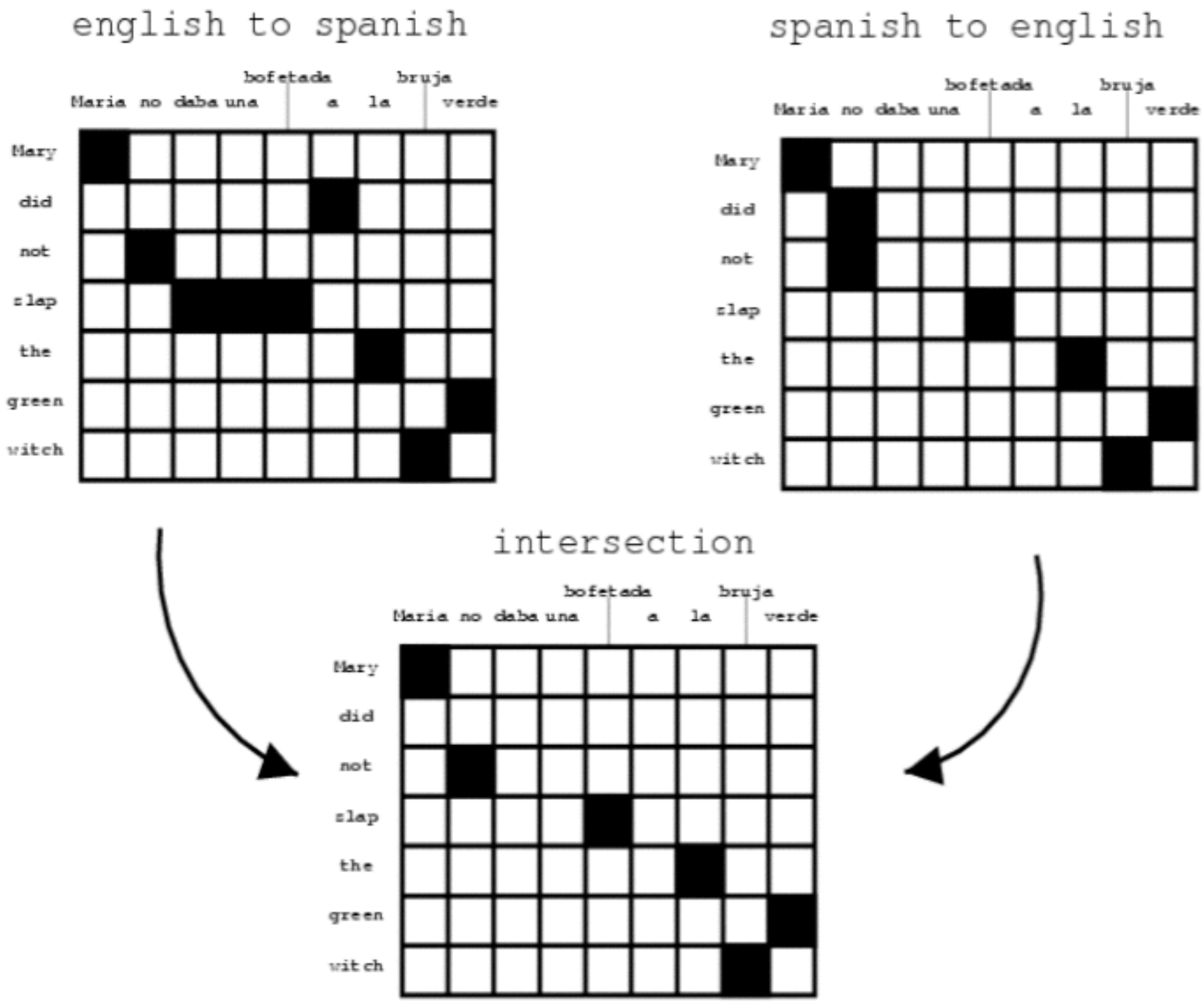
$$\frac{1/8}{3/8} = \frac{1}{3} \qquad \frac{1/4}{3/8} = \frac{2}{3} \qquad \frac{1/4}{3/8} = \frac{2}{3} \qquad \frac{1/8}{3/8} = \frac{1}{3}$$

Продолжаем алгоритм до сходимости или ограниченное число шагов

Выравнивание фраз

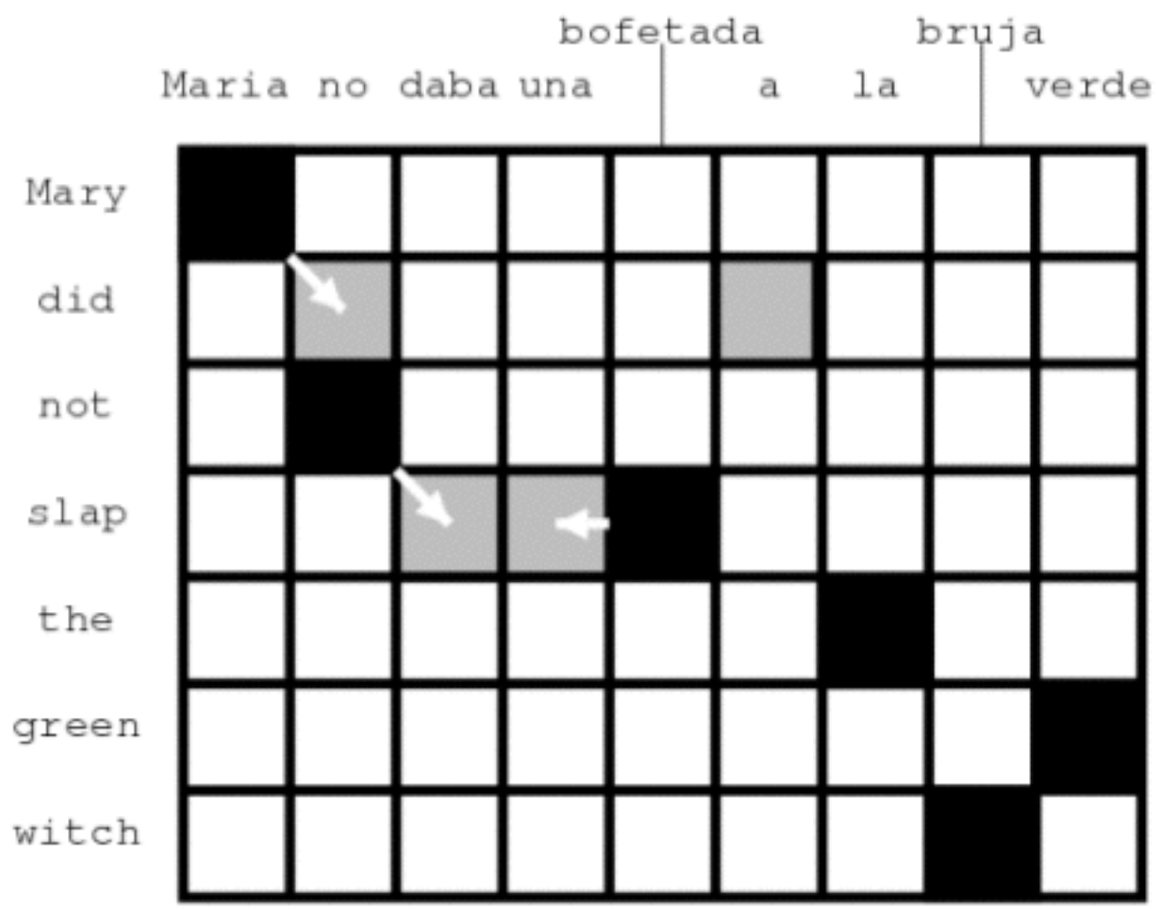
- Мы обсудили как выравнивать слова и переводить текст по словам
- Теперь перейдем к фразам

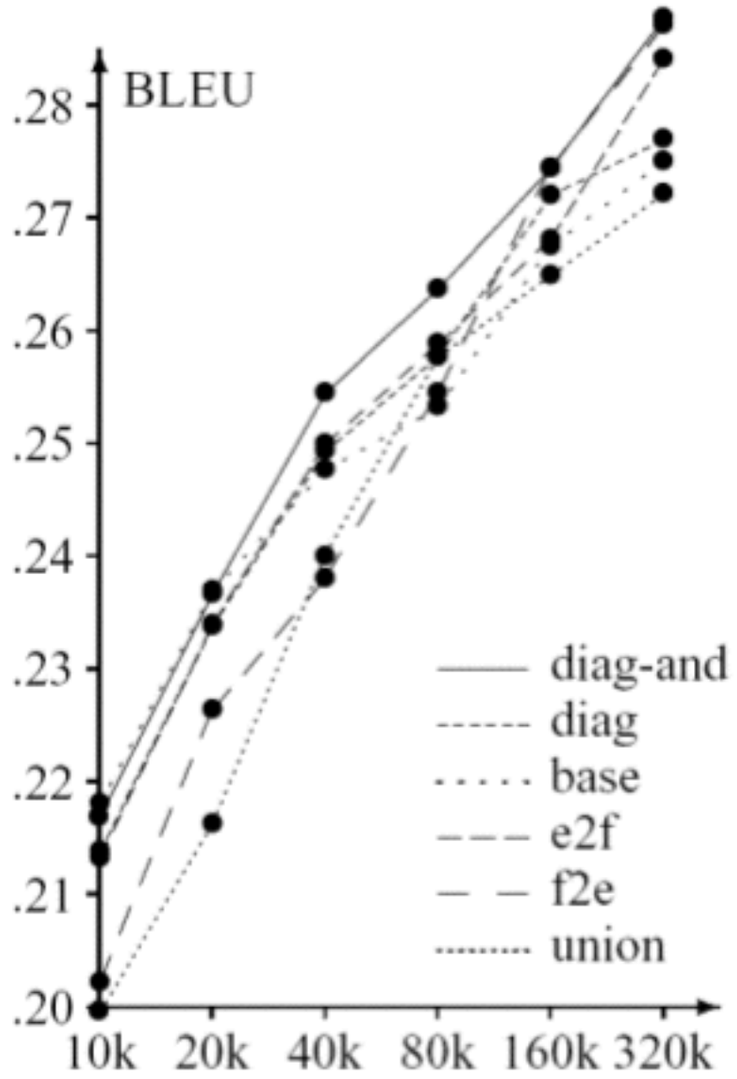
Выравнивание фраз



Выравнивание фраз

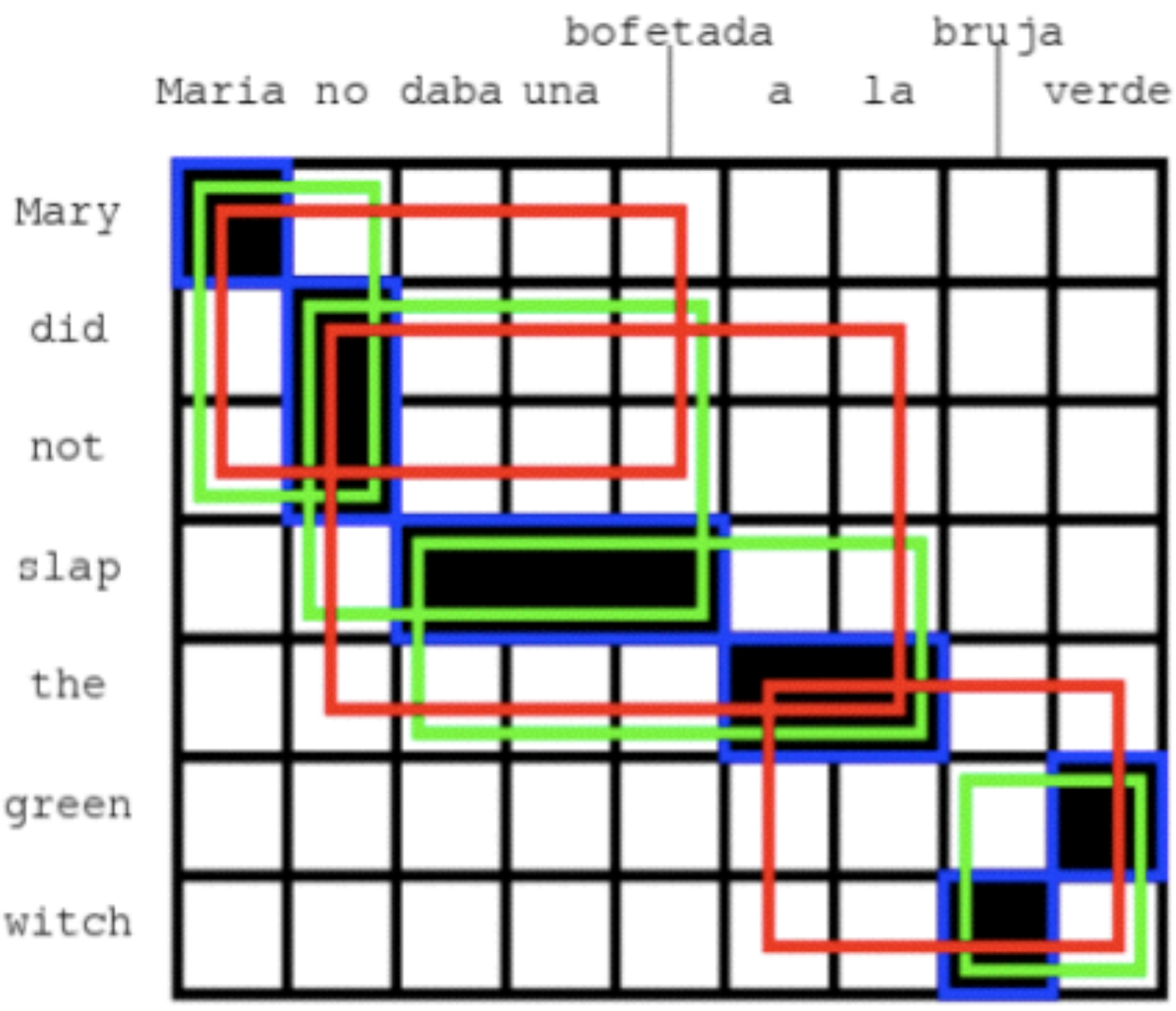
- Существует несколько эвристических методов выравнивания фраз по матрице пересечений





Извлечение фраз

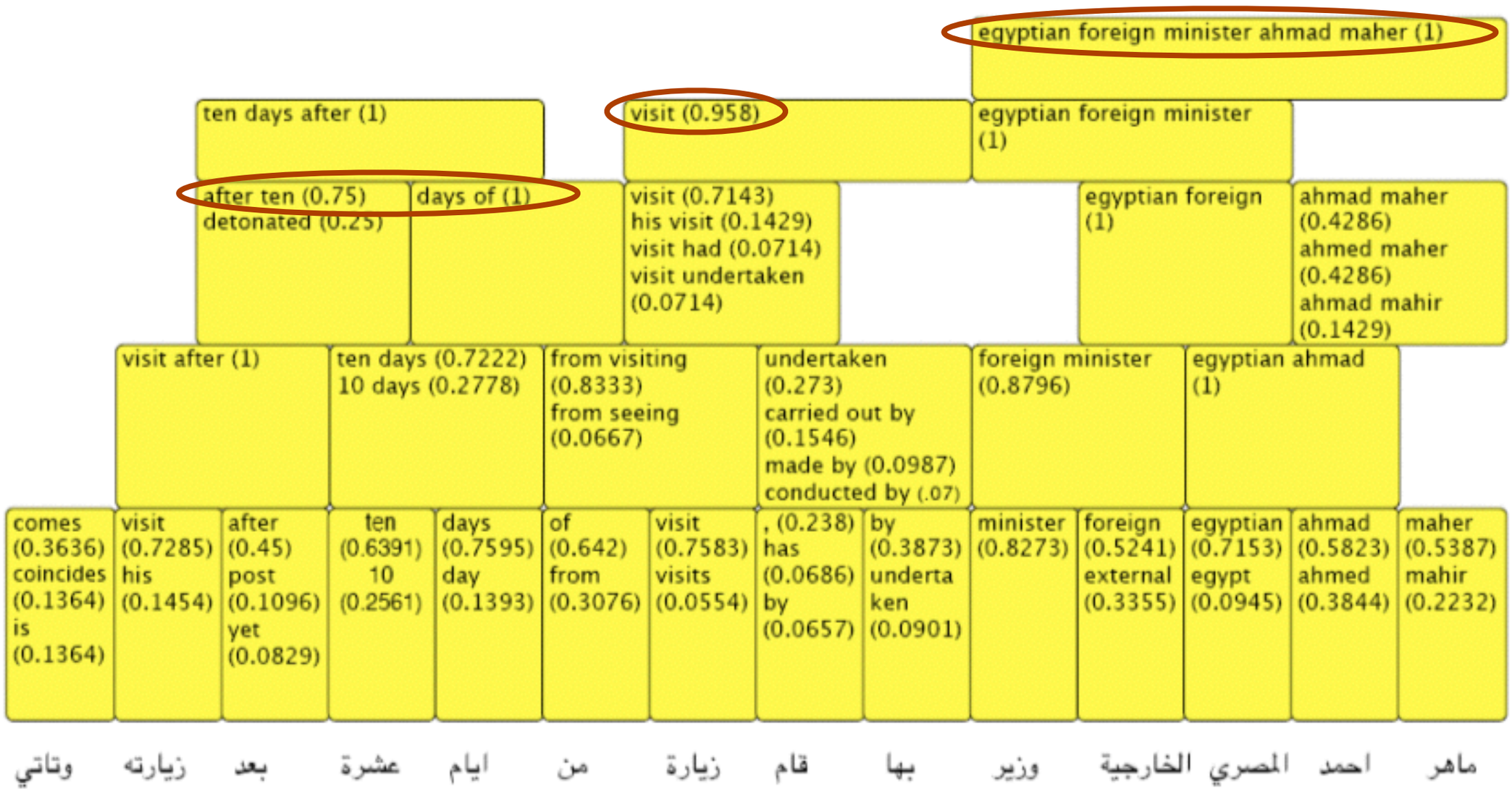
Google (Slav Petrov, SYRCoDIS'11):



Выбираем все консистентные выравнивания

Декодирование

- Аналог Витерби: выбрать предложение e максимизирующее P(e) x P(f | e)



Оценка моделей

- Оценка людьми
  - плавность
  - достоверность
    - адекватность (по фиксированной шкале)
    - информативность (ответ на вопрос по переводу)
- Автоматическая оценка
  - сравнение с одним или несколькими экспертными переводами
  - Меры качества
    - BLUE
    - NIST
    - TER
    - METEOR

Оценка моделей: BLEU

- Определить число N-грамм из машинного перевода в экспертных переводах
- Вычислить модифицированную меру точности

Оценка моделей: BLEU

Cand 1: Mary no slap the witch green
Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.
Ref 2: Mary did not smack the green witch.
Ref 3: Mary did not hit a green sorceress.

Cand 1 точность по 1-граммам: 5/6

Оценка моделей: BLEU

Cand 1: Mary no slap the witch green.
Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.
Ref 2: Mary did not smack the green witch.
Ref 3: Mary did not hit a green sorceress.

Cand 1 точность по 2-граммам: 1/5

Оценка моделей: BLEU

Cand 1: Mary no slap the witch green.
Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.
Ref 2: Mary did not smack the green witch.
Ref 3: Mary did not hit a green sorceress.

Для каждой N-граммы счетчик не должен превышать максимального количества этой n-граммы в любом предложении

Cand 2 точность 1-грамм: 7/10

Оценка моделей: BLEU

Cand 1: Mary no slap the witch green.
Cand 2: Mary did not give a smack to a green witch.

Ref 1: Mary did not slap the green witch.
Ref 2: Mary did not smack the green witch.
Ref 3: Mary did not hit a green sorceress.

Cand 2 точность 2-грамм: 4/9

Модифицированная точность

- Среднее геометрическое всех N-граммам (обычно N < 5)

$$p_n = \frac{\sum_{C \in corpus} \sum_{\text{n-gram} \in C} \text{count}_{\text{clip}}(\text{n-gram})}{\sum_{C \in corpus} \sum_{\text{n-gram} \in C} \text{count}\ (\text{n-gram})}$$

$$p = \sqrt[N]{\prod_{n=1}^{N} p_n}$$

Cand 1: $p = \sqrt[2]{\frac{5}{6}\frac{1}{5}} = 0.408$

Cand 2: $p = \sqrt[2]{\frac{7}{10}\frac{4}{9}} = 0.558$

Штраф за краткость

- Сложно посчитать полноту (recall) из-за нескольких экспертных мнений
- Вместо этого используется штраф за краткость
- Пусть *r* - длина экспертного предложения с наибольшим количеством совпадающих N-грамм. Пусть *c* - длина машинного перевода

$$BP = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}$$

Подсчет BLUE

- В итоге: BLUE = BP x p

Cand 1: Mary no slap the witch green.

Best Ref: Mary did not slap the green witch.

$$c = 6, \quad r = 7, \quad BP = e^{(1-7/6)} = 0.846$$

$$BLEU = 0.846 \times 0.408 = 0.345$$

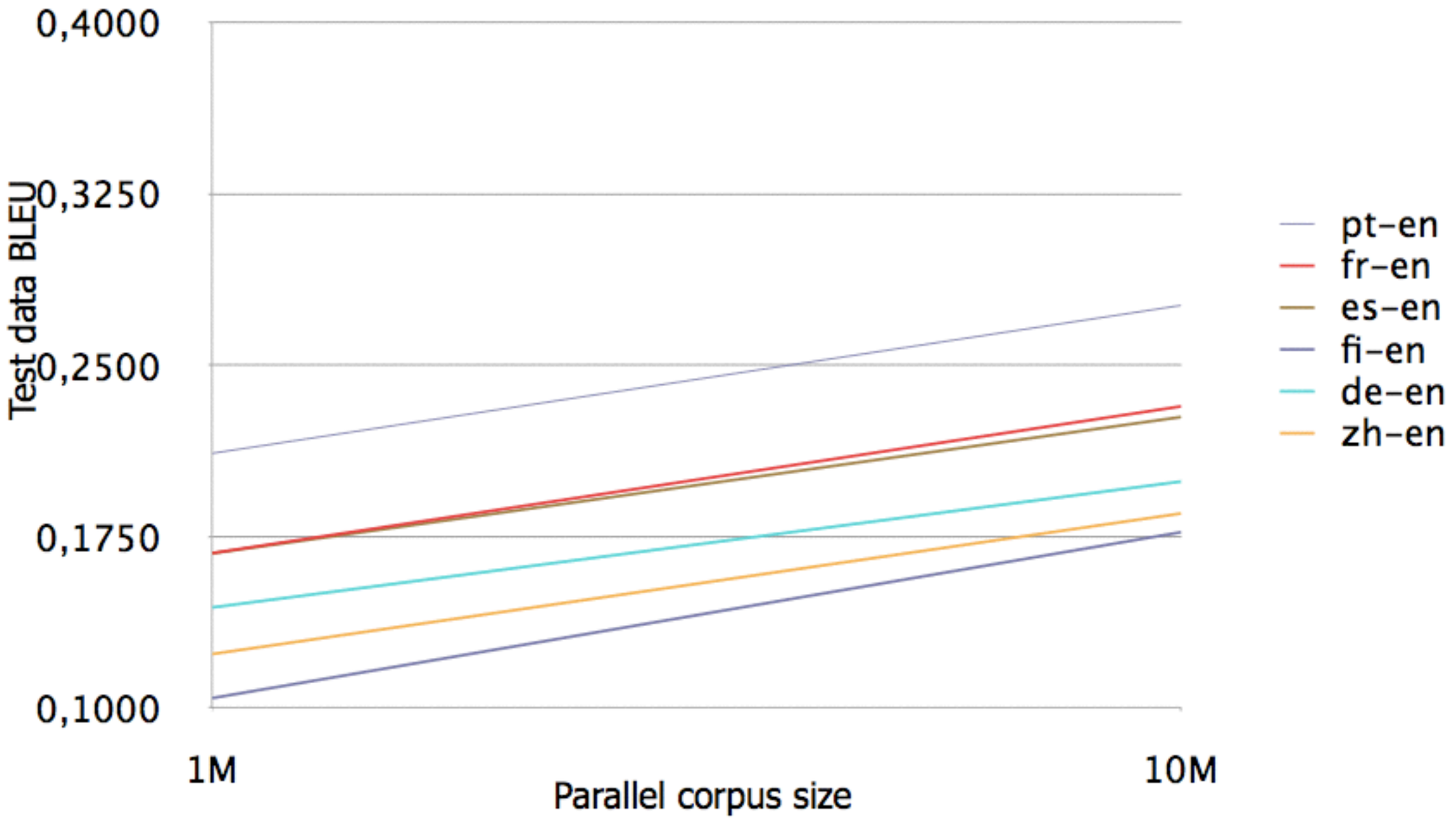
Cand 2: Mary did not give a smack to a green witch.

Best Ref: Mary did not smack the green witch.

$$c = 10, \quad r = 7, \quad BP = 1$$

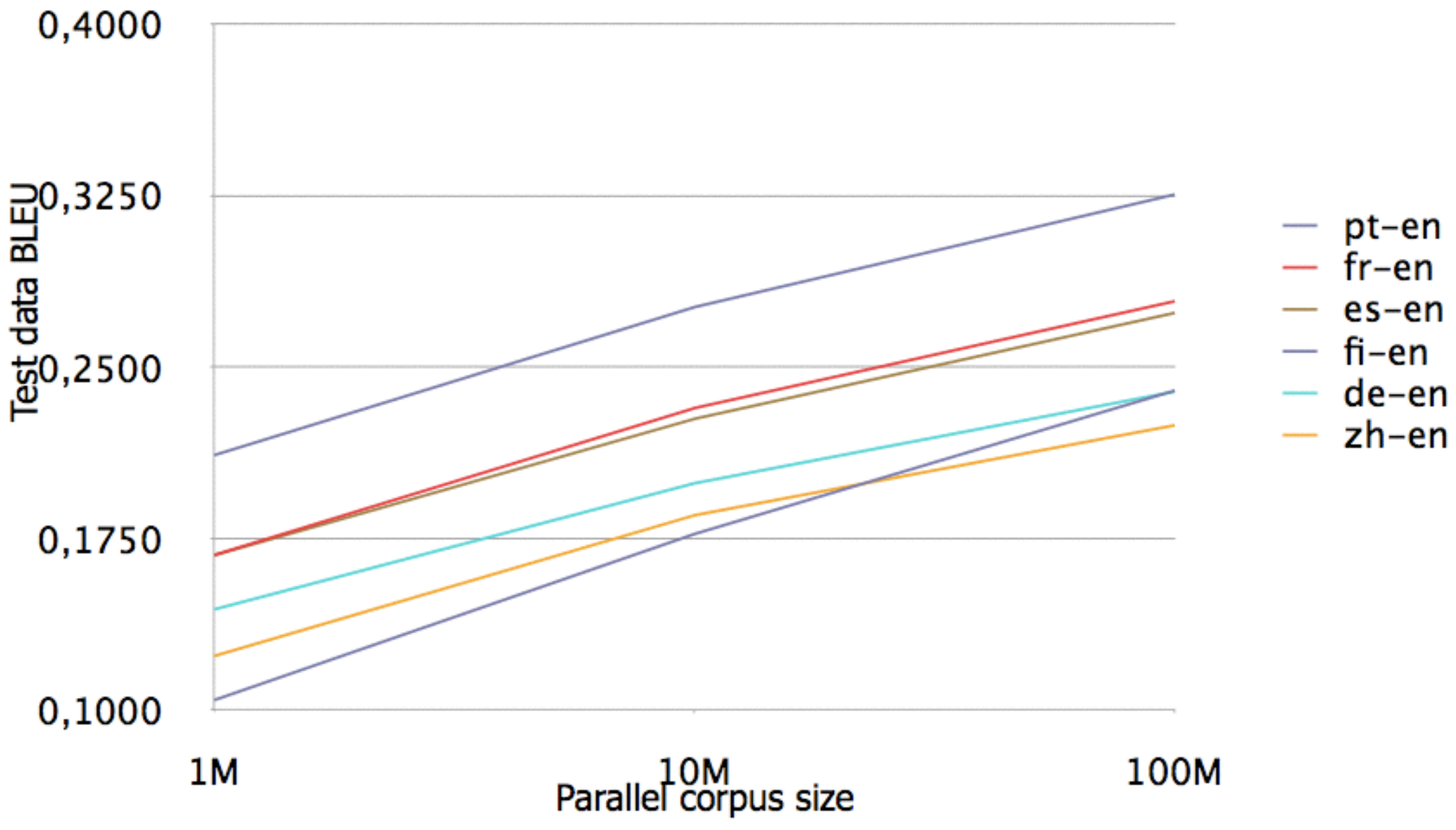
$$BLEU = 1 \times 0.558 = 0.558$$

Лучшие данные - много данных



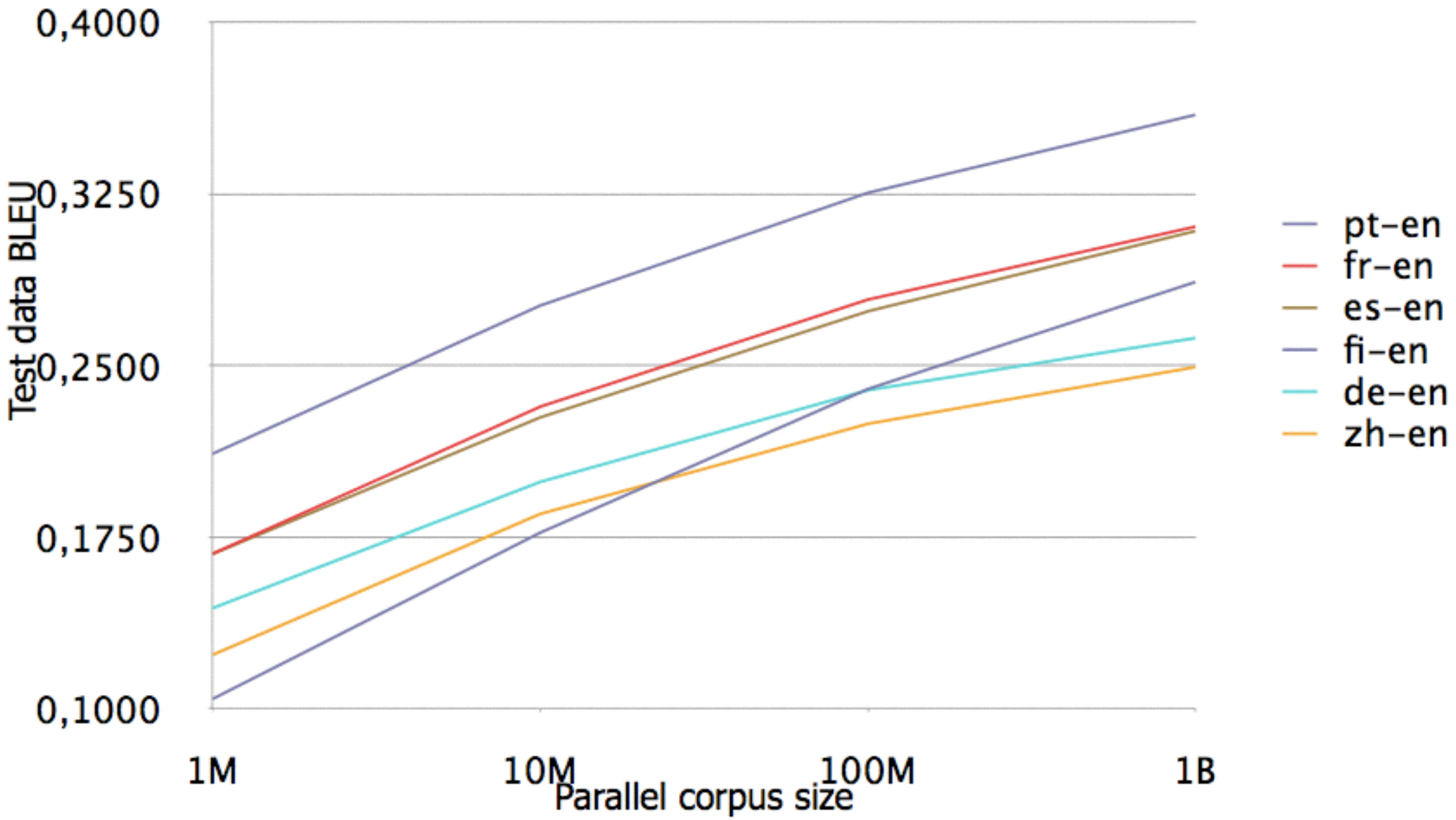
by Google

Лучшие данные - много данных



by Google

Лучшие данные - много данных



by Google

Заключение

- Трудность перевода заключается в существенных **различиях** между языками
- Классические подходы: **прямой перевод, преобразование, интерлингва**
- Для статистического машинного перевода применяется **модель зашумленного канала**, комбинирующая **модель перевода** и **языковую модель**
- Для **выравнивания** слов в двуязычных корпусах применяются формальные модели, например, **IBM Model 1**
- Для оценки систем используются различные метрики: **BLEU**, TER, METEOR.

Следующая лекция

- Тематическое моделирование

Основы обработки текстов

Лекция 11

Тематическое моделирование

Тематическое моделирование
(Topic Modelling)

- Тематическая модель коллекции текстовых документов определяет к каким темам относится каждый документ и какие слова (термины) образуют каждую тему
- Тема - набор терминов, неслучайно часто встречающихся вместе в относительно узком подмножестве документов

Задача тематического моделирования

- Вход
  - D - коллекция текстовых документов
- Задача
  - Для каждого документа определить к каким темам и в какой степени он принадлежит
  - Для каждого слова определить к каким темам и в какой степени это слово принадлежит
- Задача мягкой кластеризации
- Тематическую модель можно использовать как языковую модель

Применение

- Кластеризация документов
- Определение близости и рекомендательные системы
  - Определить насколько похожи интересы пользователей Твиттера на основе их постов
- Уменьшение размерности
  - Возможность решать задачу классификации в пространстве меньшей размерности
- Семантический поиск
- Анализ и агрегирование новостных потоков
- Поиск научной информации и фронта исследований

Основные предположения

- Порядок документов в коллекции не важен
- Порядок слов в документе не важен

- **Предварительная обработка**
  - Лемматизация или стемминг
  - Выделение терминов и словосочетаний
  - Удаление стоп-слов и слишком редких слов

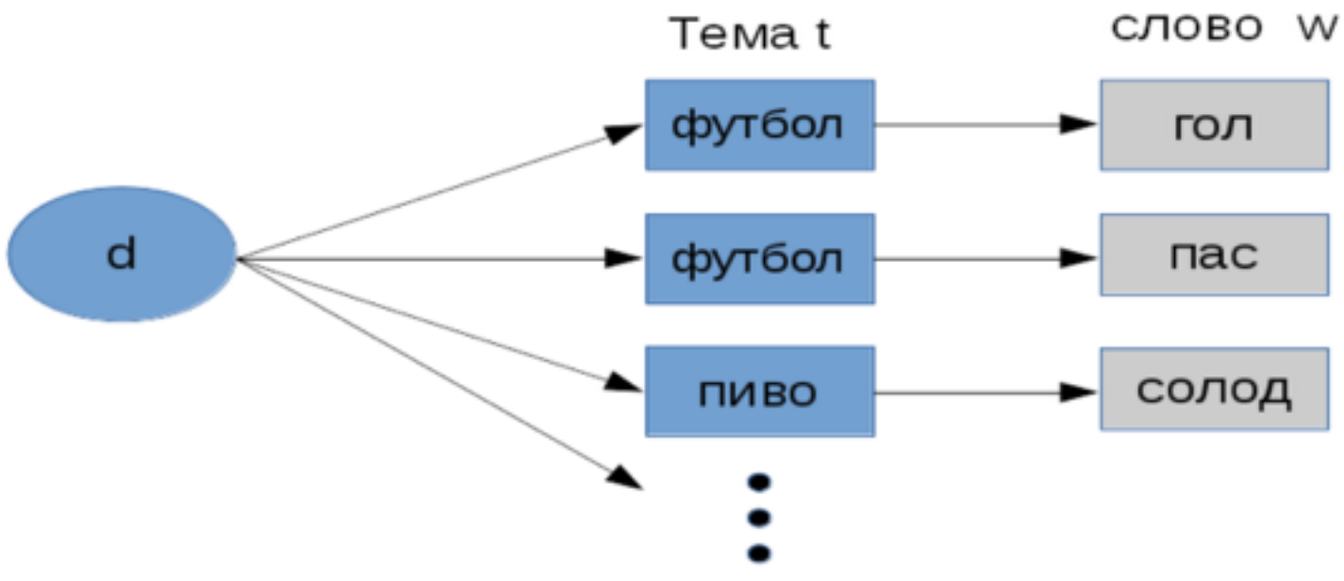
Вероятностная формализация

- Для каждой темы *t* и документа *d* зададим вероятность темы в документе *p(t|d)*
- То же самое сделаем для слов и тем: *p(w|t)* - вероятность встретить слово *w* в теме *t*
- Предположим что слова в документе зависят только от темы $p(w|d,t) = p(w|t)$
- Вероятностная модель порождения документа

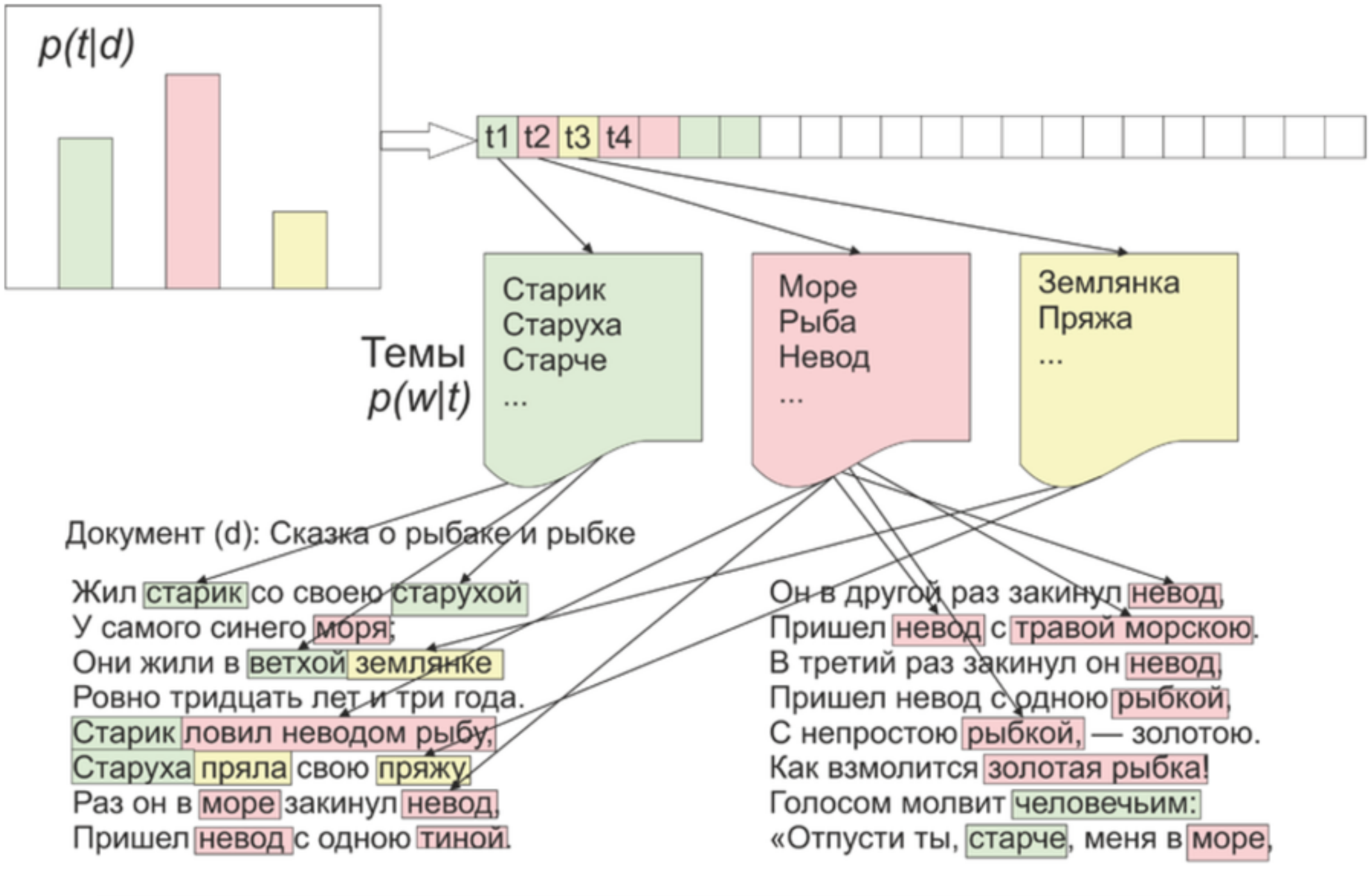
$$p(w|d) = \sum_{t \in T} p(w|d,t)p(t|d) = \sum_{t \in T} p(w|t)p(t|d)$$

Генерация документов

- Пусть мы хотим сгенерировать документ в 100 слов. Документ написан про футбол (на 0.7), про пиво (на 0.2) и про космические ракеты (на 0.1)
  1. Выбираем тему *t* для первого слова (каждая тема *t* выбирается с вероятностью *p(t|d)*)
  2. Из этой темы выбираем слово *w* (слово *w* выбирается с вероятностью *p(w|t)*)
  3. Повторяем шаги 1 и 2 для остальных 99 слов
- Как видим, слова генерируются независимо друг от друга



Пример


p(t|d)
t1 t2 t3 t4
Темы
p(w|t)
Старик
Старуха
Старче
...
Море
Рыба
Невод
...
Землянка
Пряжа
...
Документ (d): Сказка о рыбаке и рыбке
Жил старик со своею старухой
У самого синего моря;
Они жили в ветхой землянке
Ровно тридцать лет и три года.
Старик ловил неводом рыбу,
Старуха пряла свою пряжу.
Раз он в море закинул невод,
Пришел невод с одною тиной.
Он в другой раз закинул невод,
Пришел невод с травой морскою.
В третий раз закинул он невод,
Пришел невод с одною рыбкой,
С непростою рыбкой, — золотою.
Как взмолится золотая рыбка!
Голосом молвит человечьим:
«Отпусти ты, старче, меня в море,

Принцип максимума правдоподобия

- **Правдоподобие** - плотность распределения выборки

$$p(D) = \prod_{i=1}^{n} p(d_i, w_i) = \prod_{d \in D} \prod_{w \in d} p(d, w)^{n_{dw}}$$

  - n_{dw} - число вхождений термина w в документ d
- Обозначим
  - ϕ_{wt} - распределение терминов по темам
  - θ_{td} - распределение тем по документам
- Задача: найти максимум (логарифма) правдоподобия

$$L(\Phi, \Theta) = \sum_{d \in D} \sum_{w \in d} n_{dw} ln \sum_{t \in T} \phi_{wt} \theta_{td} \to max$$

с ограничениями

$$\forall t \quad \sum_w p(w|t) = 1, \quad \forall d \quad \sum_t p(t|d) = 1$$
$$\forall t, w \quad p(w|t) \geq 0, \quad \forall d, t \quad p(t|d) \geq 0$$

Принцип максимума правдоподобия

- **Правдоподобие** - плотность распределения выборки

$$p(D) = \prod_{i=1}^{n} p(d_i, w_i) = \prod_{d \in D} \prod_{w \in d} p(d, w)^{n_{dw}}$$

  - n_{dw} - число вхождений термина w в документ d
- Обозначим
  - ϕ_{wt} - распределение терминов по темам
  - θ_{td} - распределение тем по документам
- Задача: найти максимум (логарифма) правдоподобия

$$L(\Phi, \Theta) = \sum_{d \in D} \sum_{w \in d} n_{dw} ln \sum_{t \in T} \phi_{wt} \theta_{td} \to max$$

Куда делся множитель p(d)?

$$(t|d) = 1$$

$$p(t|d) \geq 0$$

Некоторые постановки задачи

- Можно не делать предположение об априорном распределении слов по темам и тем по документам (**PLSA**: вероятностный латентный семантический анализ)
- Можно предполагать, что распределения слов по темам и тем по документам получены из распределения Дирихле (**LDA**: скрытое размещение Дирихле)
- Можно учитывать редкие и общие слова (**Robust PLSA**)

PLSA

- PLSA не делает никаких предположений относительно распределений
- Параметры будем оценивать с помощью EM-алгоритма
  - Оцениваем число слов в документе *d*, порожденных темой *t*
  - Уточняем распределения документов по темам
  - Уточняем распределение тем по словам
- По правилу Байеса

$$p(t|d,w) = \frac{p(w|t)p(t|d)}{p(w|d)} = \frac{\phi_{wt}\theta_{td}}{\sum_{s\in T}\phi_{ws}\theta_{sd}}$$

Уточнение распределения тем по документам

- **E-шаг: Оценка числа слов из темы**
  - Оцениваем число слов документа *d*, порожденных из темы *t*

$$n_{td} = \sum_w n_{wd} \frac{\phi_{wt}\theta_{td}}{\sum_t \phi_{wt}\theta_{td}}$$

- **M-шаг: Оценка вероятности темы в документе**

$$\theta_{td} = p(t|d) = \frac{n_{td}}{n_d}$$

Уточнение распределения слов по темам

- **E-шаг: Оценка числа слов из темы**
  - Оцениваем число слов в теме *t*

$$n_{wt} = \sum_d n_{wd} \frac{\phi_{wt}\theta_{td}}{\sum_t \phi_{wt}\theta_{td}}$$

- **M-шаг: Оценка вероятности темы в документе**

$$\phi_{wt} = p(w|t) = \frac{n_{wt}}{n_t}$$

Недостатки PLSA

- PLSA переобучается, т.к. число параметров ϕ_{wt} и θ_{td} слишком велико $|D| \cdot |T| + |W| \cdot |T|$
- PLSA не позволяет управлять разреженностью
  – если в начале $\phi_{wt} = 0$, то в финале $\phi_{wt} = 0$
  – если в начале $\theta_{td} = 0$, то в финале $\theta_{td} = 0$
- PLSA неверно оценивает вероятность новых слов: если

$$n_w = 0 \quad \text{то} \quad \hat{p}(w|d) = 0, \ \forall t \in T$$

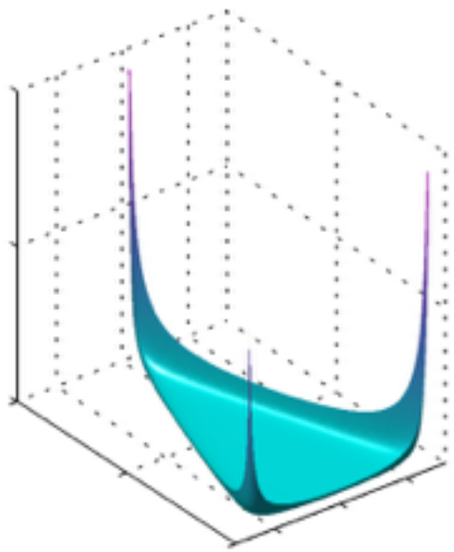
Модель LDA

- Пусть распределение тем по документам и слов по темам имеет априорное распределение Дирихле (симметричное) с плотностью вероятности

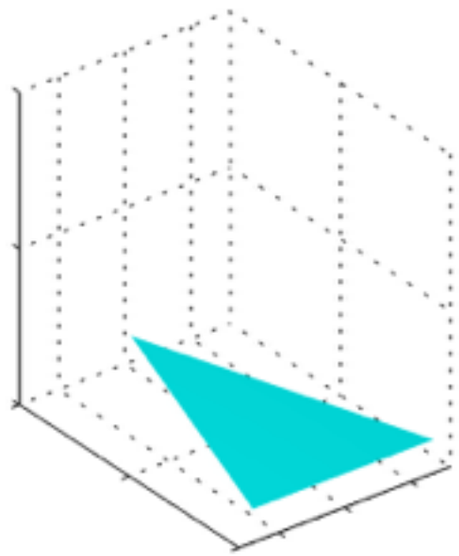
$$f(x_1, x_2, \ldots x_n) = C x_1^{\alpha-1} \times x_2^{\alpha-1} \times \cdots \times x_n^{\alpha-1}$$

- Чем больше параметр α, тем более ***сглаженные*** распределения будем получать

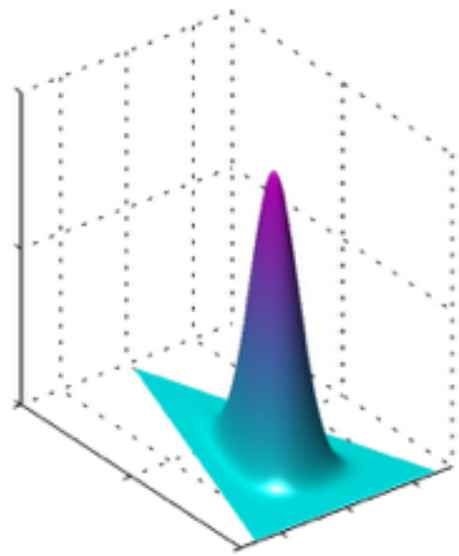
Распределение Дирихле



$\alpha_1 = \alpha_2 = \alpha_3 = 0.1$



$\alpha_1 = \alpha_2 = \alpha_3 = 1$



$\alpha_1 = \alpha_2 = \alpha_3 = 10$

- Почему именно распределение Дирихле?
  - Математическое удобство
  - Порождает как сглаженные, так и разреженные векторы
  - Неплохо описывает кластерные структуры на симплексе

Отличие LDA от PLSA

- В PLSA - несмещенные оценки максимума правдоподобия:

$$\phi_{wt} = \frac{n_{wt}}{n_t} \qquad \theta_{td} = \frac{n_{td}}{n_d}$$

- В LDA - сглаженные байесовские оценки:

$$\phi_{wt} = \frac{n_{wt} + \beta_w}{n_t + \beta_0} \qquad \theta_{td} = \frac{n_{td} + \alpha_t}{n_d + \alpha_0}$$

Недостатки LDA

- Слабые лингвистические обоснования “особой роли” распределения Дирихле
- В оригинальном методе сложный вывод параметров (требует интегрирования по пространству параметров модели)
- Сглаживание вместо разреживания
- На практике на достаточно больших данных нет значимых различий между PLSA и LDA

Robust PLSA

$$p(w|d) = \frac{Z_{dw} + \gamma\pi_{dw} + \epsilon\pi_w}{1 + \gamma + \epsilon}$$

Z_{dw} Тематическая компонента. Совпадает с моделью PLSA. Если она плохо объясняет избыточную частоту слова в документа, то слово относят к фону или шуму

$\pi_{dw} \equiv p_{noise}(w|d)$ Шумовая компонента. Слова специфичные для конкретного документа d, либо редкие термины, относящиеся к темам, слабо представленным в данной коллекции

$\pi_w \equiv p_{bgr}(w)$ Фоновая компонента. Общеупотребительные слова, в частности, стоп-слова, не отброшенные на стадии предварительной обработки

$\gamma,\ \epsilon$ Параметры, ограничивающие долю слов в каждой компоненте

Реализации тематических моделей

- **Gensim** - реализация для Python
- **BigARTM** - Распределенная реализация PLSA с аддитивный регуляризацией на C
- Topic Modelling в **Spark** - распределенная реализация Robust PLSA с аддитивной регуляризацией на фреймворке Spark (https://github.com/akopich/dplsa)

Пример LDA на gensim

- Википедия, в качестве коллекции

```
>>> lda.print_topics(20)
topic #0: 0.009*river + 0.008*lake + 0.006*island + 0.005*mountain + 0.004*area + 0.004*park + 0.004*antarctic + 0
topic #1: 0.026*relay + 0.026*athletics + 0.025*metres + 0.023*freestyle + 0.022*hurdles + 0.020*ret + 0.017*divis
topic #2: 0.002*were + 0.002*he + 0.002*court + 0.002*his + 0.002*had + 0.002*law + 0.002*government + 0.002*polic
topic #3: 0.040*courcelles + 0.035*centimeters + 0.023*mattythewhite + 0.021*wine + 0.019*stamps + 0.018*oko + 0.0
topic #4: 0.039*al + 0.029*sysop + 0.019*iran + 0.015*pakistan + 0.014*ali + 0.013*arab + 0.010*islamic + 0.010*ar
topic #5: 0.020*copyrighted + 0.020*northamerica + 0.014*uncopyrighted + 0.007*rihanna + 0.005*cloudz + 0.005*know
topic #6: 0.061*israel + 0.056*israeli + 0.030*sockpuppet + 0.025*jerusalem + 0.025*tel + 0.023*aviv + 0.022*pales
topic #7: 0.015*melbourne + 0.014*rovers + 0.013*vfl + 0.012*australian + 0.012*wanderers + 0.011*afl + 0.008*dina
topic #8: 0.011*film + 0.007*her + 0.007*she + 0.004*he + 0.004*series + 0.004*his + 0.004*episode + 0.003*films +
topic #9: 0.019*wrestling + 0.013*château + 0.013*ligue + 0.012*discus + 0.012*estonian + 0.009*uci + 0.008*hockey
topic #10: 0.078*edits + 0.059*notability + 0.035*archived + 0.025*clearer + 0.022*speedy + 0.021*deleted + 0.016*
topic #11: 0.013*admins + 0.009*acid + 0.009*molniya + 0.009*chemical + 0.007*ch + 0.007*chemistry + 0.007*compoun
topic #12: 0.018*india + 0.013*indian + 0.010*tamil + 0.009*singh + 0.008*film + 0.008*temple + 0.006*kumar + 0.00
topic #13: 0.047*bwebs + 0.024*malta + 0.020*hobart + 0.019*basa + 0.019*columella + 0.019*huon + 0.018*tasmania +
topic #14: 0.014*jewish + 0.011*rabbi + 0.008*bgwhite + 0.008*lebanese + 0.007*lebanon + 0.006*homs + 0.005*beirut
topic #15: 0.025*german + 0.020*der + 0.017*von + 0.015*und + 0.014*berlin + 0.012*germany + 0.012*die + 0.010*des
topic #16: 0.003*can + 0.003*system + 0.003*power + 0.003*are + 0.003*energy + 0.002*data + 0.002*be + 0.002*used
topic #17: 0.049*indonesia + 0.042*indonesian + 0.031*malaysia + 0.024*singapore + 0.022*greek + 0.021*jakarta + 0
topic #18: 0.031*stakes + 0.029*webs + 0.018*futsal + 0.014*whitish + 0.013*hyun + 0.012*thoroughbred + 0.012*dnf
topic #19: 0.119*oblast + 0.034*uploaded + 0.034*uploads + 0.033*nordland + 0.025*selsoviet + 0.023*raion + 0.022*
```

- 6 часов 20 минут на MacBook Pro, Intel Core i7 2.3GHz, 16GB DDR3 RAM, OS X

Для дальнейшего изучения

- *Thomas Hofmann*. Probilistic latent semantic analysis // Proceedings of the Twenty-Second Annual International SIGIR Conference on Research and Development in Information Retrieval. 1999.
- *David M. Blei, Andrew Ng, Michael Jordan*. Latent Dirichlet allocation // Journal of Machine Learning Research (3) 2003 pp. 993-1022.
- Воронцов К. В., Потапенко А. А. Модификации EM-алгоритма для вероятностного тематического моделирования // Машинное обучение и анализ данных. — 2013
- *Коршунов Антон, Гомзин Андрей* Тематическое моделирование текстов на естественном языке // *Труды Института системного программирования РАН* : журнал. — 2012.
- Воронцов К. В. Лекции по вероятностным тематическим моделям
- Байесовские методы машинного обучения (курс лекций, Д.П. Ветров, Д.А. Кропотов)
- machinelearning.ru

Заключение

- Тематические модели являются одним из способов моделирования языка
- Тематические модели являются генеративными моделями: каждый документ определяет темы, а каждая тема определяет слова
- Тематическое моделирование можно рассматривать как задачу мягкой кластеризации документов