

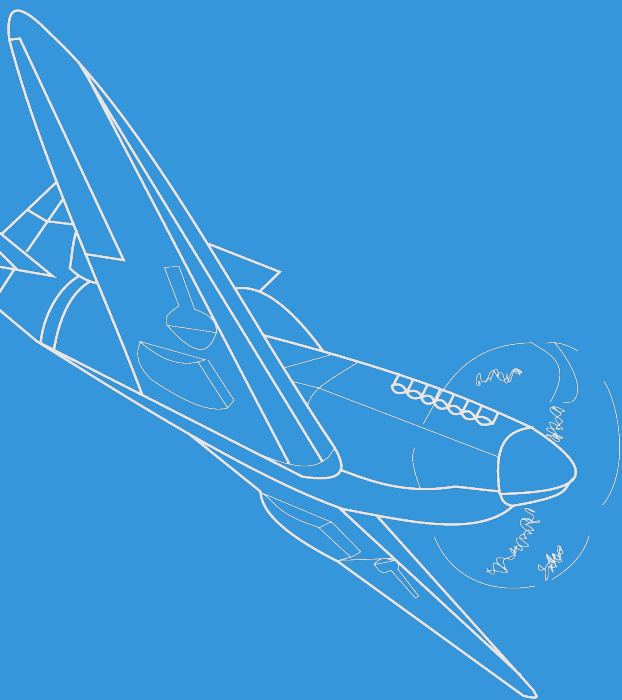
# Словари и нечёткий поиск

(основано на слайдах Андрея Калинина, Hinrich Schütze,  
Christina Lioma)

---

## Содержание занятия

1. Словарь
2. Запросы с мета-символами
3. Проверка правописания
4. Soundex
5. Исправление запросов



# Словарь

# Обратный индекс



Brutus → 1 → 2 → 4 → 11 → 31 → 45 → 173

Calpurnia → 2 → 31 → 54 → 101

Caesar → 1 → 2 → 4 → 5 → 6 → 16 → ...

Словарь

Координаты

# Словарь как массив



- Для каждого термина нужно сохранить:
  - количество документов (частотность)
  - указатель на координаты
  - ...
- На время допустим, что можно представить эту информацию в виде структуры фиксированной длины.
- Тогда можно использовать массив для хранения словаря.

# Словарь как массив



Термин	Частотность	Координатный блок
a	656256	→
aachen	65	→
...	...	...
zulu	221	→
<b>объём:</b> 20 байт	4 байта	4 байта

Как искать термин запроса  $q_i$  в этом массиве? То есть: какую структуру данных можно использовать, чтобы найти строку, в которой находится  $q_i$ ?

# Структуры данных поиска терминов



- Два основных класса: **хеши** и **деревья**.
- Некоторые ИСП используют хеши, некоторые — деревья.
- Основные вопросы выбора:
  - Количество терминов фиксировано, или растёт?
  - Какие относительные частоты доступа к разным ключам?
  - Сколько разных ключей имеется?



- Каждый термин хешируется в целое число.
- Боремся с коллизиями.
- Во время запроса: хешируем термин запроса, разрешаем коллизии, находим нужную строку в массиве.
- **Плюсы:**
  - Поиск в хеш-таблице быстрее, чем поиск в дереве.
  - Время поиска — константа.



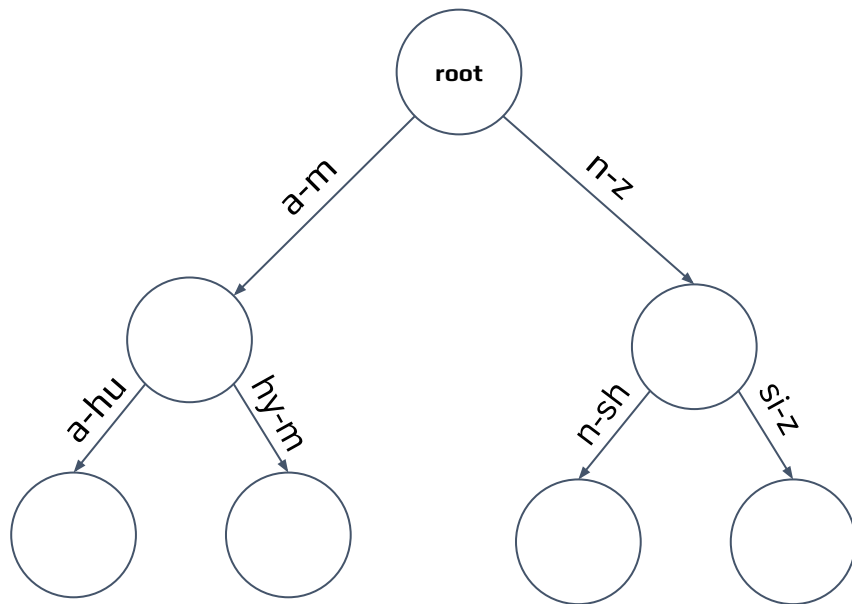


- **Минусы**
  - Нельзя найти небольшие различия (resume и résumé)
  - Нельзя искать по префиксу (все термины, начинающиеся с automat)
  - Для растущего словаря придётся время от времени всё рехешировать.
- Теоретически, можно сделать «морфологическую» хеш-функцию.

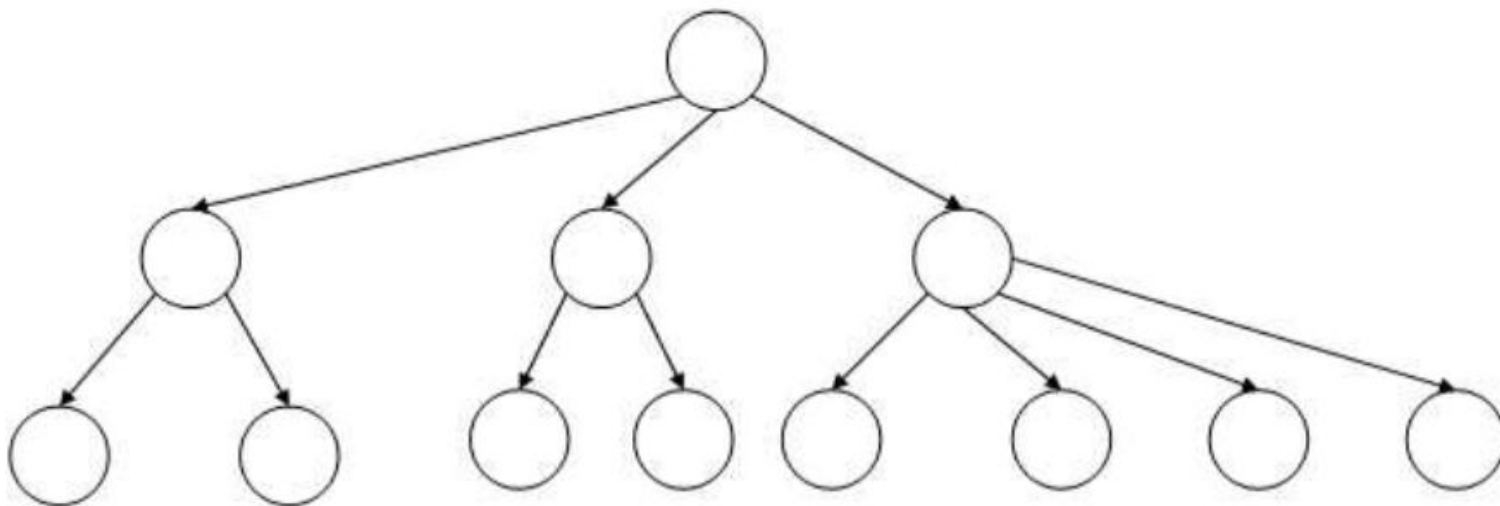


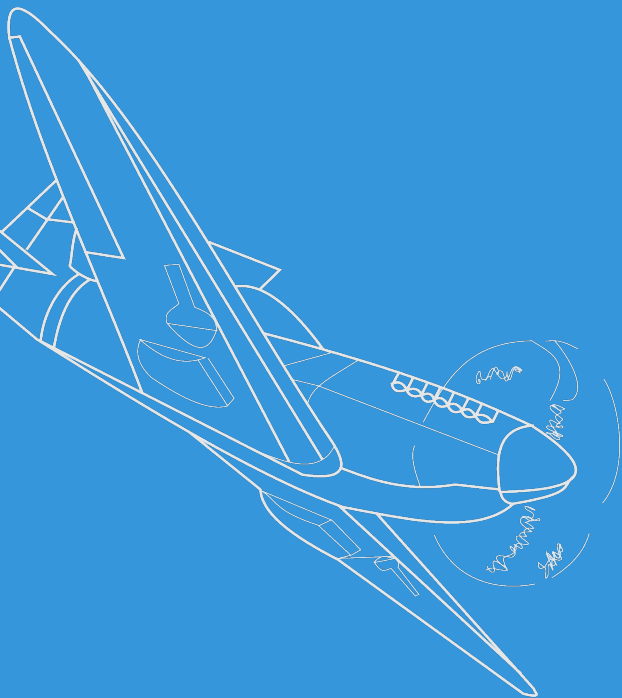
- Деревья позволяют искать термины с общим префиксом.
- Простейшее дерево — бинарное.
- Поиск медленнее хешей,  $O(\log M)$ , где  $M$  — размер словаря.
- $O(\log M)$  соблюдается для **сбалансированных** деревьев.
- Так же можно использовать **В-деревья**.

# Бинарное дерево



# В-дерево





# Запросы с мета- символами

# Запросы с мета-символами



- $mon^*$ : найти все документы, содержащие термин, начинающийся с  $mon$
- Просто для B-дерева: найти все термины  $t$ , находящиеся в диапазоне  $mon \leq t < moo$
- $*mon$ : найти все термины, заканчивающиеся на  $mon$ 
  - Создаём дополнительное дерево, для терминов, записанных задом наперёд.
  - Теперь по этому дереву получаем термины  $t$  в диапазоне  $nom \leq t < non$
- Результат: множество терминов, подходящих под маску.
- Теперь нужно найти документы, содержащие любой из этих терминов.

# Как обработать \* внутри термина



- Например: m\*nchen
- Можно поискать m\* и \*nchen в В-деревьях и пересечь два полученных множества.
- Довольно расточительно.
- Альтернатива: индекс **перестановок**
- Основная идея: «переворачивать» каждый запрос с маской таким образом, чтобы \* оказалась в конце.
- Хранить каждый поворот каждого термина в словаре, в том же В-дереве.

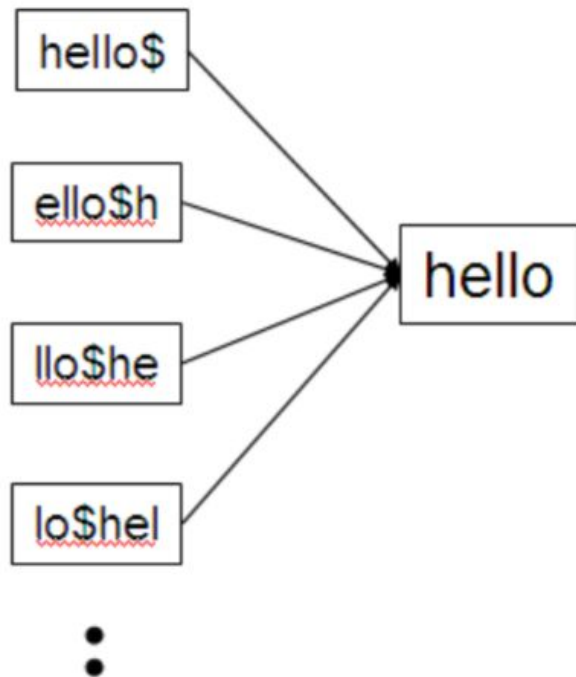
# Индекс перестановок



- Для термина `hello`: добавим *hello\$*, *ello\$h*, *llo\$he*, *lo\$hel*, и *o\$hell* в B-дерево, где \$ — специальный символ.



# Отображение перестановок в термины



# Индекс перестановок



- Итак, для `hello` храним: `hello$`, `ello$h`, `llo$he`, `lo$hel` и `o$hell`
- Тогда запросы
  - `X`, ищем `X$`
  - `X*`, ищем `X*$`
  - `*X`, ищем `X$*`
  - `*X*`, ищем `X*`
  - `X*Y`, ищем `Y$X*`
  - Например: для `hel*o` ищем `o$hel*`
  - Как обработать запрос `X*Y*Z?`

# Поиск в индексе перестановок



- Прокрутить запрос так, чтобы \* была справа.
- Искать как обычно.
- Однако: такой индекс как минимум **учетверяет** размер словаря (для английского языка, для русского — увеличит в 7-8 раз).



- Занимает меньше места, чем индекс перестановок.
- Индексируем все символьные  $k$ -граммы (последовательности из  $k$  символов) термина.
- 2-граммы часто называют **биграммами**.
- Например: из April is the cruelest month получим следующие биграммы: \$a ap pr ri il l\$ \$i is s\$ \$t th he e\$ \$c cr ru ue el le es st t\$ \$m mo on nt h\$
- \$ — специальный символ, обозначающий границу слова.
- Добавляем в новый индекс не термины, а биграммы.

# 3-граммный обратный индекс



# k-граммные индексы



- Теперь у нас два разных вида обратных индексов.
- Есть индекс терминов-документов.
- И есть индекс  $k$ -грамм, чтобы находить термины по запросам, состоящие из  $k$ -грамм.

# Выполнение запроса с метасимволами для биграмм



- Запрос `mon*` можно обработать так:  
`$m and mo and on`
- Так получим все термины с префиксом `mon...`
- ... но и много «ложных срабатываний», таких как `moon`.
- Их нужно отфильтровать, напрямую сравнивая термины с запросом.
- Оставшиеся термины нужно искать в индексе терминов-документов.
- *k*-граммный индекс и индекс перестановок
  - *k*-граммный индекс занимает меньше места.
  - Индекс перестановок не требует пост-фильтрации.

# Упражнение



Почему у больших веб-поисков нет поддержки запросов с масками?

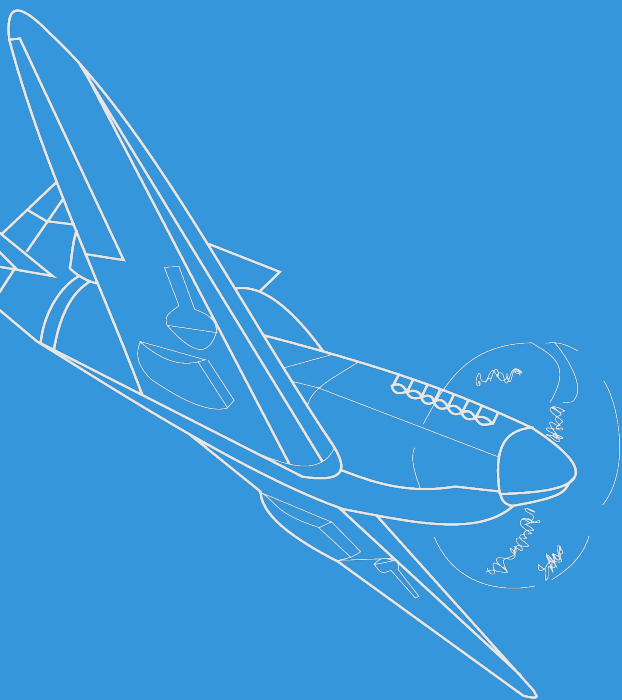


# Упражнение



Почему у больших веб-поисков нет поддержки запросов с масками?

- Много слов.
- Увеличивается количество обрабатываемых терминов.
- Люди будут вводить меньше символов в словах.



# Проверка правописания



- Два возможных применения:
  - Исправление документов.
  - Исправление запросов.
- Два разных метода:
  - Исправление **отдельных слов**
    - Проверяет каждое слово.
    - Не сможет исправить опечатки в словарных терминах, например [an asteroid that fell **form** the sky]
  - **Контекстно-зависимое** исправление
    - Обращает внимание на контекст, окружающие слова.
    - Может заменить ошибку в предыдущем примере (form/from)

# Исправление документов



- Интерактивная коррекция документов не нужна.
- Используется в основном для распознанных документов (системы OCR).
- Обычно документы никак не изменяются.



- Самое простое: исправление отдельных слов
- Допущение 1: имеется список «правильных слов».
- Допущение 2: есть способ вычисления **расстояния** между словом с опечаткой и правильным словом.
- Тогда простейший алгоритм возвращает «правильное» слово с наименьшим расстоянием к слову с опечаткой.
- Например: informaton → information
- В качестве списка правильных слов можно использовать словарь ИПС.
- Почему это плохо?

# Альтернативные источники «правильных» слов



- Стандартные словари (Зализняк)
- Технические словари (для специализированных ИПС)
- Отфильтрованные словари корпуса ИПС

# Расстояния между словами



- Две альтернативы:
  - Расстояние **Левенштейна**
    - Взвешенное расстояние Левенштейна
  - Пересечение k-грамм

# Расстояние Левенштейна



- Расстояние между строками  $s_1$  и  $s_2$  — количество элементарных операций редактирования, нужных для преобразования  $s_1$  в  $s_2$ .
- Расстояние Левенштейна: операции вставки, удаления и замены.
  - dog-do: 1
  - cat-cart: 1
  - cat-cut: 1
  - cat-act: 2
- Расстояние Левенштейна-Дameraу: добавлена операция перестановки двух рядом стоящих символов.
  - cat-act: 1



# Вычисление расстояния Левенштейна



		f	a	s	t
	0	1	2	3	4
c	1	1	2	3	4
a	2	2	1	2	3
t	3	3	2	2	2
s	4	4	3	2	3

# Расстояние Левенштейна: вычисление



---

**Algorithm** Edit distance

---

**Input:**  $\alpha = \alpha_1 \dots \alpha_n$  and  $\beta = \beta_1 \dots \beta_m$

```
1: for  $i \leftarrow 0$  to  $n$  do
2:    $D_{i,0} \leftarrow i$ ;
3: end for
4: for  $j \leftarrow 0$  to  $m$  do
5:    $D_{0,j} \leftarrow j$ ;
6: end for
7: for  $i \leftarrow 1$  to  $n$  do
8:   for  $j \leftarrow 1$  to  $m$  do
9:      $t \leftarrow (\alpha_i = \beta_j) ? 0 : 1$ ;
10:     $D_{i,j} \leftarrow \min\{D_{i-1,j-1} + t, D_{i,j-1} + 1, D_{i-1,j} + 1\}$ ;
11:   end for
12: end for
13. return  $D_{n,m}$ 
```

---

# Взвешенное расстояние



- Аналогично предыдущему, но веса операций зависят от символов.
- Нужно для учёта клавиатурных опечаток, например  $m$  более вероятно ошибочно напечатать как  $n$ , чем как  $q$ .
- Поэтому, замена  $m$  на  $n$  — меньшее расстояние, чем замена на  $q$ .
- Теперь нужна матрица весов.
- Так же нужно добавить в алгоритм учёт этих весов.

# Исправление опечаток с учётом весов



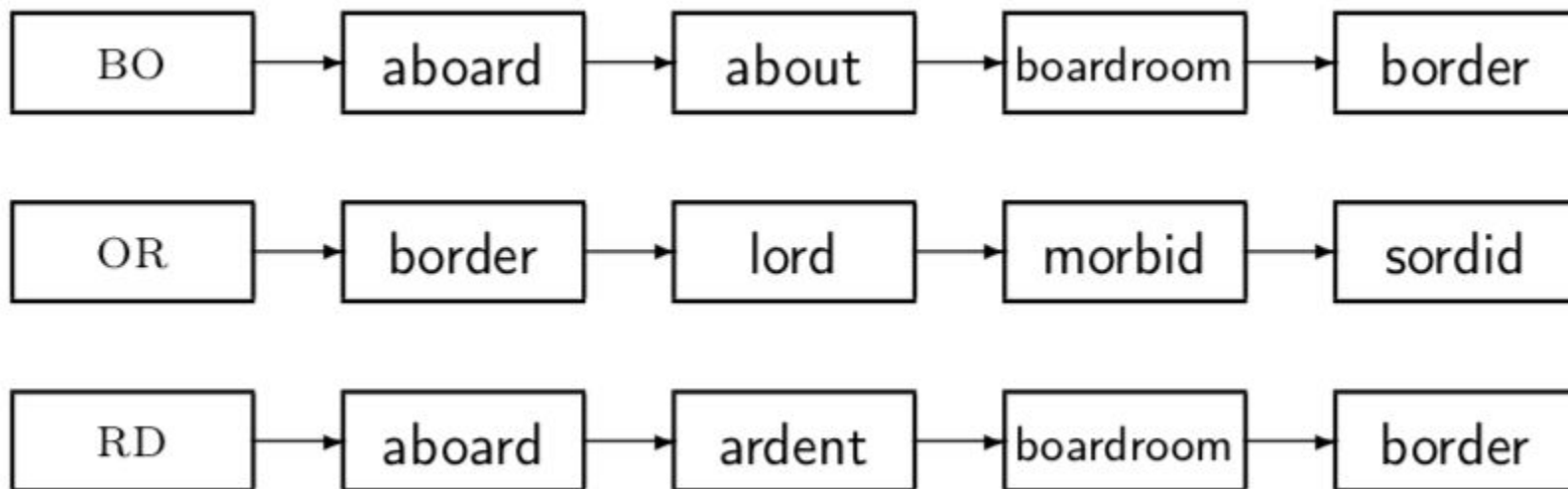
- Для данного запроса перебрать все строки на заданном расстоянии.
- Пересечь это множество со списком «правильных» слов.
- Предложить термины из пересечения пользователю.
- Или автоматически исправить запрос.
- Что лучше?

# k-граммный индекс для исправления опечаток



- Перебрать все k-граммы из термина запроса
- Например: биграммный индекс, слово с опечаткой `bordroom`
- Биграммы: `bo`, `or`, `rd`, `dr`, `ro`, `oo`, `om`
- Используем индекс k-грамм для получения «правильных» слов
- Устанавливаем предел по количеству совпавших k-грамм
- Например, нужны только такие термины, которые отличаются не больше чем по трём k-граммам.

# Исправление опечатки для bordroom



# Пример с триграммами



- Проблема: фиксированное количество отличающихся k-грамм по разному работает для слов разной длины.
- Например, правильное слово november
  - Триграммы: nov, ove, vem, **emb**, **mbe**, **ber**
- И запрос december
  - Триграммы: dec, ese, sem, **emb**, **mbe**, **ber**
- Таким образом, **3 триграммы** пересекаются (из 6 у каждого термина)
- Нужна нормализованная метрика.

# Коэффициент Жаккара



- Метрика пересечения двух множеств.
- Два множества, A и B
- Коэффициент Жаккара: 
$$\frac{|A \cap B|}{|A \cup B|}$$
- A и B не обязаны иметь одинаковый размер.
- Результат — число между 0 и 1.
- `december/november` — какой коэффициент Жаккара?
- В проверке правописания можно использовать в качестве лимита, исправлять только для значений коэффициента  $> 0.8$ .



# Контекстно-зависимая проверка (1)



- Пример: an asteroid that fell **form** the sky
- Как можно исправить слово form?
- Одна из идей: статистика вхождений.
  - Получить «правильные» термины, близкие к каждому термину.
  - Для запроса `flew form munich`: flea для flew, from для form, munch для munich

## Контекстно-зависимая проверка (2)

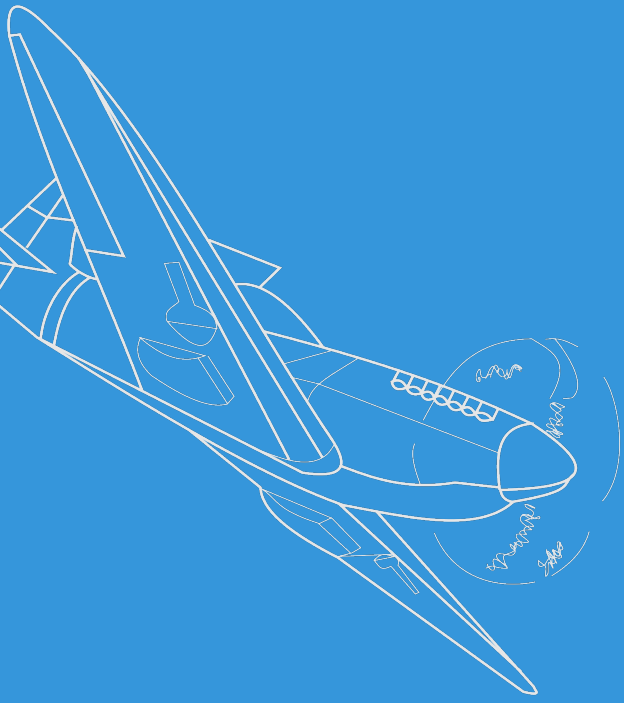


- Поискать все возможные варианты с одним исправленным словом:
  - Сначала `"flea form munich"`
  - Затем `"flew from munich"`
  - Потом `"flew form munch"`
- Правильный запрос `"flew from munich"` вернёт больше всего результатов.
- Допустим, у нас есть 7 вариантов для `flew`, 20 для `form` и 3 for `munich`, сколько запросов на проверку получится?

## Контекстно-зависимая проверка (3)



- Алгоритм на предыдущем слайде не очень эффективен.
- Лучшая альтернатива — исследовать запросы, а не документы.



# Soundex



- **Soundex** — алгоритм нахождения фонетических альтернатив.
- Например: chebyshev / tchebyscheff
- Алгоритм:
  - Преварить каждый токен в 4-х символьную сокращённую форму.
  - То же самое сделать для терминов запроса.
  - Построить и использовать отдельный индекс сокращённых форм.



1. Оставим первый символ термина.
2. Следующие символы заменяются на '0' (ноль): A, E, I, O, U, H, W, Y
3. Заменить символы на цифры:
  - a. B,F,P,V на 1
  - b. C,G,J,K,Q,S,X,Z на 2
  - c. D,T на 3
  - d. L на 4
  - e. M,N на 5
  - f. R на 6
4. Повторно удалять по цифре из последовательных повторов.
5. Удалить все нули. Добавить в конец нули так, чтобы в оставшемся коде было бы как минимум 4 знака. Вернуть код из 4-х знаков: буква на первой позиции и первые три цифры.

# Soundex для HERMAN



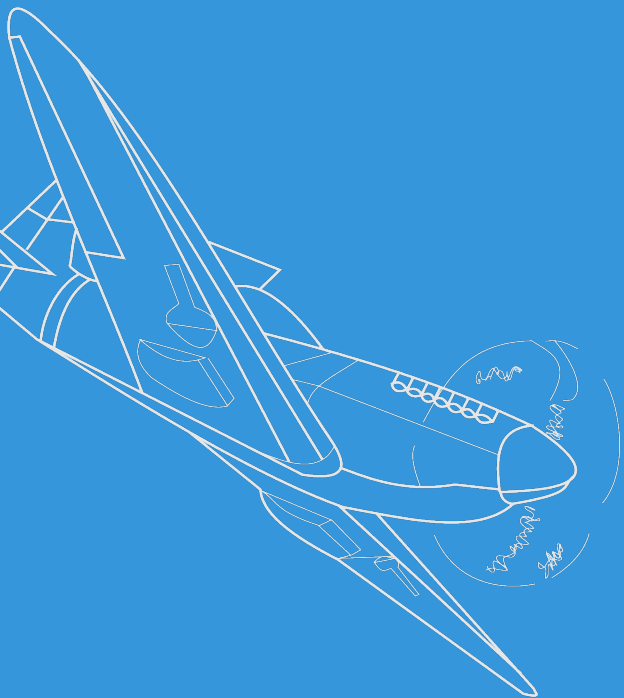
- Оставим H
- ERMAN → ORMON
- ORMON → 06505
- 06505 → 06505
- 06505 → 655
- Результат: H655
- Для HERMANN будет сгенерирован тот же код.

# Насколько полезен Soundex?



- Для информационного поиска не очень.
- Подходит для задач с высоким уровнем полноты, например, Интерпол использует Soundex для своей картотеки.
- Существуют лучшие альтернативы.
- Есть адаптации для русского языка, не очень хорошие.





# Исправление запросов

# Бритни Спирс:

## <https://archive.google.com/jobs/britney.html>



488941 britney spears	29 britent spears	9 brinttany spears	5 brney spears	3 britiy spears	2 brirreny spears
40134 brittany spears	29 britttany spears	9 britanay spears	5 broitney spears	3 britmeny spears	2 brirtany spears
36315 brittney spears	29 britttany spears	9 britinany spears	5 brotny spears	3 britneey spears	2 brirttany spears
24342 britany spears	29 btiney spears	9 britn spears	5 bruteny spears	3 britnehy spears	2 brirttney spears
7331 britny spears	26 birttney spears	9 britnew spears	5 btiyney spears	3 britnely spears	2 britain spears
6633 briteny spears	26 breitney spears	9 britneyn spears	5 btrittney spears	3 britnesy spears	2 britane spears
2696 brittney spears	26 brinity spears	9 britney spears	5 gritney spears	3 britnetty spears	2 britaneny spears
1807 briney spears	26 britenay spears	9 britny spears	5 spritney spears	3 britnex spears	2 britania spears
1635 brittny spears	26 britneyt spears	9 brtittney spears	4 bittny spears	3 britneyxxx spears	2 britann spears
1479 brintey spears	26 brittan spears	9 brtny spears	4 bnritney spears	3 britnity spears	2 britanna spears
1479 britanny spears	26 brittne spears	9 rbytny spears	4 brandy spears	3 britntey spears	2 britannie spears
1338 britiny spears	26 btittany spears	9 rbitny spears	4 brbritney spears	3 britney spears	2 britannt spears
1211 britnet spears	24 beitney spears	8 birtiny spears	4 breatiny spears	3 britterny spears	2 britannu spears
1096 britiney spears	24 birtney spears	8 bithney spears	4 breatney spears	3 brittneey spears	2 britanyl spears
991 britaney spears	24 brightney spears	8 brattany spears	4 bretiney spears	3 britttnney spears	2 britanyt spears
991 britnay spears	24 brintiny spears	8 breitny spears	4 brfitney spears	3 britttnyey spears	2 briteeny spears
811 britney spears	24 britanty spears	8 breteny spears	4 briattany spears	3 brityen spears	2 britteny spears
811 brtiney spears	24 britenny spears	8 brightny spears	4 brieteny spears	3 briytney spears	2 britenet spears
664 birtney spears	24 britini spears	8 brintay spears	4 briety spears	3 brltney spears	2 britteniy spears
664 brintney spears	24 brittny spears	8 brinttey spears	4 briitny spears	3 broteny spears	2 britenys spears
664 briteney spears	24 brittni spears	8 briotney spears	4 briittany spears	3 brtaneey spears	2 britianey spears
601 bitney spears	24 brittnie spears	8 britanys spears	4 brinie spears	3 brtiiany spears	2 britin spears
601 brinty spears	21 birtney spears	8 britley spears	4 brinteny spears	3 brtinay spears	2 britinary spears
544 brittaney spears	21 birtay spears	8 britneyb spears	4 brintne spears	3 brtinney spears	2 britmy spears
544 brittnay spears	21 biteny spears	8 britnrey spears	4 britaby spears	3 brititany spears	2 britnaney spears
364 britey spears	21 bratney spears	8 britity spears	4 britaey spears	3 brititeny spears	2 britnat spears
364 brittiny spears	21 britani spears	8 brittner spears	4 britaeney spears	3 brtnet spears	2 britnbey spears
329 brtney spears	21 britanie spears	8 brottany spears	4 britinie spears	3 brytiny spears	2 britndy spears
269 bretney spears	21 briteany spears	7 baritney spears	4 britinney spears	3 btney spears	2 britneh spears
269 britneys spears	21 brittany spears	7 birtney spears	4 britmney spears	3 drittney spears	2 britneney spears
244 britne spears	21 brittinay spears	7 biteney spears	4 britnear spears	3 pretney spears	2 britney6 spears
244 brytney spears	21 brtany spears	7 bitiny spears	4 britnel spears	3 rbritney spears	2 britneye spears
220 breatney spears	21 britnyu spears	7 breatney spears	4 britneuy spears	2 barittany spears	2 britneyh spears
220 britiany spears	19 birney spears	7 brianty spears	4 britnewy spears	2 bbbritney spears	2 britneym spears
199 britnnay spears	19 brirtney spears	7 brintey spears	4 britnmey spears	2 bbitney spears	2 britneyyy spears
163 britny spears	19 britnaey spears	7 britianny spears	4 brittaby spears	2 bbritny spears	2 britnhey spears
147 breatny spears	19 britnee spears	7 britly spears	4 brittery spears	2 bbbrittany spears	2 britnjey spears
147 brittiney spears	19 britony spears	7 britnej spears	4 britthey spears	2 beitany spears	2 britnne spears

# Система исправления опечаток в запросах



- Дипломная работа Татьяны Романовой.
- Защищена в 2010-м году на оценку «отлично».

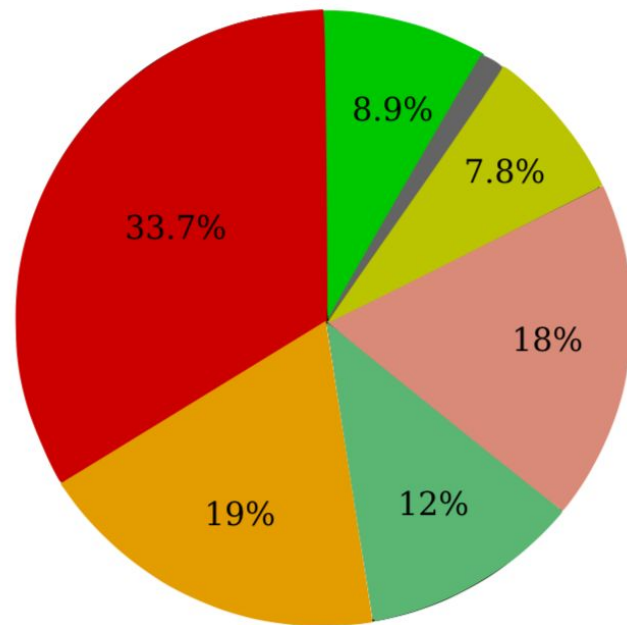
# Ошибки в запросах



- Около 10 % запросов содержат ошибки.

Распределение ошибок по типам:

- Орфографические (видио, кораловый)
- Опечатки (аэропорь, в отрданом)
- Фамилии и бренды (том соер, нафтезин)
- Транслитерация (ntrcns вместо тексты)
- Пробелы (нтвсмотреть, нижний нов город)
- Контекстные (свиной гриб, вокруг меха)
- Иностранные слова (lie to mi, clinitec)





- Списки введенных запросов (логи):
  1. Большой лог запросов — 11 млн. запросов.
  2. Тематические логи запросов — 10–40 тыс. запросов.
  3. Частотный словарь русского языка — 30 тыс. слов.
- Словари:
  1. Словарь юниграмм — 1 млн. слов.
  2. Словарь биграмм — 7 млн. словосочетаний.



Имея запрос  $q_t$ , найдем запрос  $q_c$  так, чтобы:

$$P(q_c|q_t) \rightarrow \max_{q_c \in \Sigma^*}$$

По формуле Байеса:

$$P(q_c|q_t) = \frac{P(q_t|q_c)P(q_c)}{P(q_t)}$$

Итеративное исправление запроса:

ра<sup>с</sup>то<sup>я</sup>и<sup>е</sup> лев<sup>и</sup>ш<sup>т</sup>ейна → ра<sup>с</sup>то<sup>я</sup>ни<sup>е</sup> лев<sup>и</sup>н<sup>ш</sup>тейна → расстояние левенштейна

# Шаги одной итерации



1. Разбить запрос на части;
2. Для каждой части составить список вариантов замен;
3. Оценить вес каждой замены;
4. Составить граф слов;
5. Найти оптимальный путь в графе — алгоритм Витерби.

# Общие проблемы исправления опечаток



- Пользовательский интерфейс:
  - Заменять автоматически или предлагать?
  - Возможно, Вы имели в виду подходит только для одного предложения.
  - Что делать с большим количеством вариантов?
  - Компромисс между простым или гибким интерфейсом.
- Затраты:
  - Потенциально очень затратно.
  - Обработать выборочно?
  - Например, только для запросов, вернувших мало результатов.
  - Исправление опечаток для крупных ИПС достаточно быстро работает, чтобы обслуживать каждый запрос.



## Рекомендуемая литература

Введение в информационный поиск  
I Маннинг Кристофер Д., Шютце  
Хайнрих



Для саморазвития (опционально)  
Чтобы не набирать двумя  
пальчиками

Спасибо за  
внимание!

**Антон Кухтичев**



[a.kukhtichev@mail.ru](mailto:a.kukhtichev@mail.ru)



[@toshunster](https://www.instagram.com/toshunster)