# Теория информационного поиска

## Лекция 6. Исправление опечаток

Дмитрий Грановский

СПбГУ

28.04.2025

# Примеры

**~10–15%** запросов содержат ошибки/опечатки.

- [однаклассники]
- [однокласники]
- [однолассники]
- [jlyjrkfcybrb]
- [одноклас]
- [одно классники]
- [аднакласни]

# Типы ошибок

1. Орфография
   - [скочать ...], [скчать ...], [скаачать ...], [скачтаь ...]
2. Пробелы
   - [арендагазели], [анти статик]
3. Раскладка/транслит (не всегда ошибки)
   - [мл],
   - [спбу ру],
   - [macbuk kak pereklychit yazik]
4. Смешанные
   - [vr,erghj] -> [мкбукпро] -> [макбук про]

# Типы ошибок

5. Непонятные
   - [пагода в пекине]
   - [rfcnh.kz] — домен или кастрюля?
   - [CNJK 2V] — стол или модель чего-то?
   - [клон] vs [африканский клон]

# Ключевые вопросы

1. Есть ли в запросе опечатки и где именно?
2. Как должен выглядеть исправленный запрос?
3. Что мы делаем с опечатками?

# Стратегии

В зависимости от нашей уверенности в исправлении:

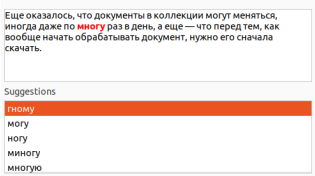
1. Автоисправление
2. Подсказка
   - «возможно, вы искали…»
   - мы ограничены в количестве
3. Смешивание
   - требует обработки двух запросов!
   - по результатам поиска можем заменить на автоисправление

# Первый подход

1. Ищем слово в словаре
2. Если не нашлось, то это опечатка
   - словарь нужно все время актуализировать
3. Ищем ближайшее* слово для замены
   - или несколько

# Первый подход: проблемы

- нормально работает в текстовых редакторах, но:
- не можем давать выбор из множества вариантов
- словарь всегда неполон
  - как вообще составить словарь?
- опечатка может быть существующим словом (*real-word spelling error*)
  - [всемирный потом]
  - мешает «умная» клавиатура в мобильных устройствах



# Поиск ближайшего слова

Ключевое понятие: **расстояние Левенштейна**
(тж. расстояние редактирования, *edit distance*)

# Поиск ближайшего слова

Ключевое понятие: **расстояние Левенштейна**
(тж. расстояние редактирования, *edit distance*)

Идея: посчитаем, сколько операций нужно сделать для превращения одного слова в другое

# Расстояние Левенштейна

| | Я | Н | **В** | **А** | Р | **Ь** |
|---|---|---|---|---|---|---|
| К | Р | О | **В** | **А** | Т | **Ь** |

# Расстояние Левенштейна

| | Я | Н | **В** | **А** | Р | **Ь** |
|---|---|---|---|---|---|---|
| К | Р | О | **В** | **А** | Т | **Ь** |

$d_L(\text{январь}, \text{кровать}) = 4$

# Расстояние Левенштейна

| | Я | Н | **В** | **А** | Р | **Ь** |
|---|---|---|---|---|---|---|
| К | Р | О | **В** | **А** | Т | **Ь** |

$$d_L(\text{январь}, \text{кровать}) = 4$$

| Х | О | Р | **В** | **А** | **Т** | И | Я |
|---|---|---|---|---|---|---|---|
| К | Р | О | **В** | **А** | **Т** | Ь | |

# Расстояние Левенштейна

| | Я | Н | **В** | **А** | Р | **Ь** |
|---|---|---|---|---|---|---|
| К | Р | О | **В** | **А** | Т | **Ь** |

$d_L(\text{январь}, \text{кровать}) = 4$

| Х | О | Р | **В** | **А** | **Т** | И | Я |
|---|---|---|---|---|---|---|---|
| К | Р | О | **В** | **А** | **Т** | Ь | |

$d_L(\text{хорватия}, \text{кровать}) = 5$

# Расстояние Дамерау-Левенштейна

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Х | О | Р | **В** | **А** | **Т** | И | Я |
| К | Р | О | **В** | **А** | **Т** | Ь | |

# Расстояние Дамерау-Левенштейна

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Х | О | Р | **В** | **А** | **Т** | И | Я |
| К | Р | О | **В** | **А** | **Т** | Ь | |

$$d_{DL}(\text{хорватия}, \text{кровать}) = 4$$

1. вставка
2. удаление
3. замена
4. **перестановка**

# Ещё о расстоянии редактирования

- веса разных операций имеет смысл делать разными
  - например, учитывать расстояние на клавиатуре
  - или статистику из логов запросов
- можно добавлять свои операции
  - например, «смена раскладки»
- как выбрать между несколькими словами с одинаковым $d_L$?
  - `[тстер] -> {тестер, тостер}`
  - `[белеты] -> {`**`билеты`**`, береты, белены, ...}`
    - может помочь частота
- можно ли учесть контекст?
  - `[белеты до омска]`

# Переход к вероятности

$$fix^* = \underset{fix \in D}{argmax}(P(fix \mid orig)),$$

где

- $orig$ — запрос, который исправляем
- $fix^*$ — искомое исправление
- $D$ — множество всех возможных запросов
- возможно, $orig = fix$

# Модели

$$P(fix \mid orig) = \frac{P(orig|fix) \times P(fix)}{P(orig)}$$

- (это формула Байеса)
- $P(orig \mid fix)$ — из модели ошибок
  - какова вероятность, что пользователь набирал $fix$, а получилось $orig$?
- $P(orig)$, $P(fix)$ — из модели языка
  - какова вероятность, что это «настоящее» слово?
- в целом это модель зашумленного канала (*noisy channel model*), тж. используется в классическом машинном переводе

# Модели

1. Модель ошибок
   - разумно связать с близостью слов
   - например: $P(orig \mid fix) = \alpha^{-d_L(orig, fix)}$
   - можно упростить до вероятностей на N-граммах
   - можно усложнять пользовательской статистикой и т.д.
2. Модель языка
   - скорее генеративная модель, чем словарь
   - $P(w_1 \dots w_n) = P(w_1 w_2) P(w_2 w_3) \dots P(w_n)$
     - скорее всего, нужно сглаживание
   - нужно регулярно обновлять

# Что дальше

- проблема: для $argmax$ нужно перебрать все потенциально возможные замены и посчитать вероятность каждой (нереально)
- значит, нужна генерация кандидатов в исправления, скорее всего — пословно
- значит, нужен некий индекс или другая структура, возвращающая множество слов, *более или менее похожих* на данное
- это называется нечеткий поиск (*approximate string matching*, *fuzzy search*)

# Soundex/Metaphone

- алгоритмы для сопоставления слову цифро-буквенного кода
  - Soundex — изначально для имен/фамилий
- идея: похожие на слух слова получат одинаковые коды
  - можно считать хэшированием (*phonetic hash*)
- можно создать индекс, где ключом будет фонетический код
- работает только для отдельных типов опечаток

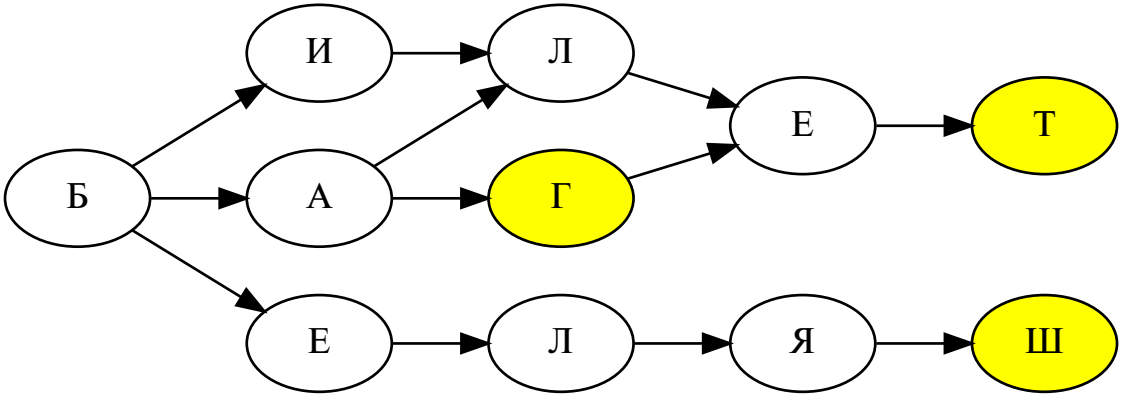
# Soundex: алгоритм

| буква | переходит в |
|---|---|
| a, e, i, o, u, y, h, w | 0 |
| b, f, p, v | 1 |
| c, g, j, k, q, s, x, z | 2 |
| d, t | 3 |
| l | 4 |
| m, n | 5 |
| r | 6 |

# Soundex: примеры

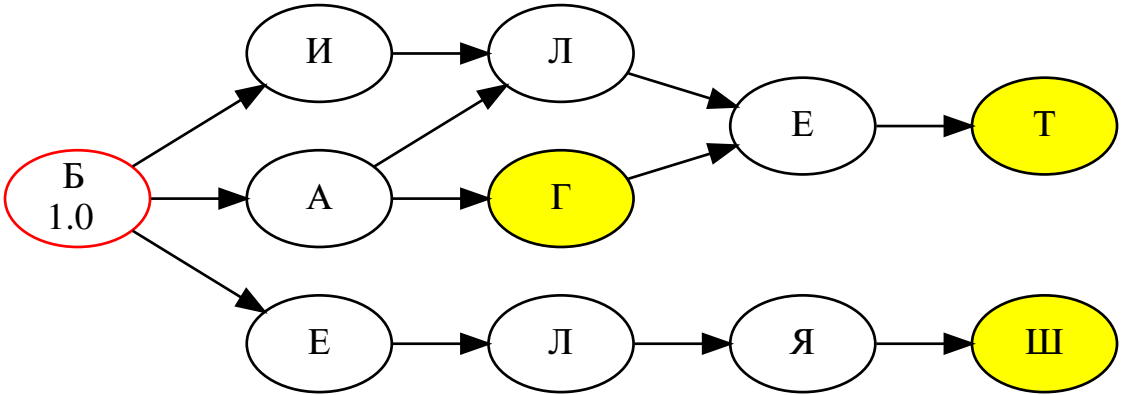
| слово | код |
|---|---|
| Jane | J500 |
| Johann | J500 |
| Jake | J200 |
| Yoan | Y500 |
| Robert | R163 |
| Rupert | R163 |

# Trie (префиксное дерево, бор)



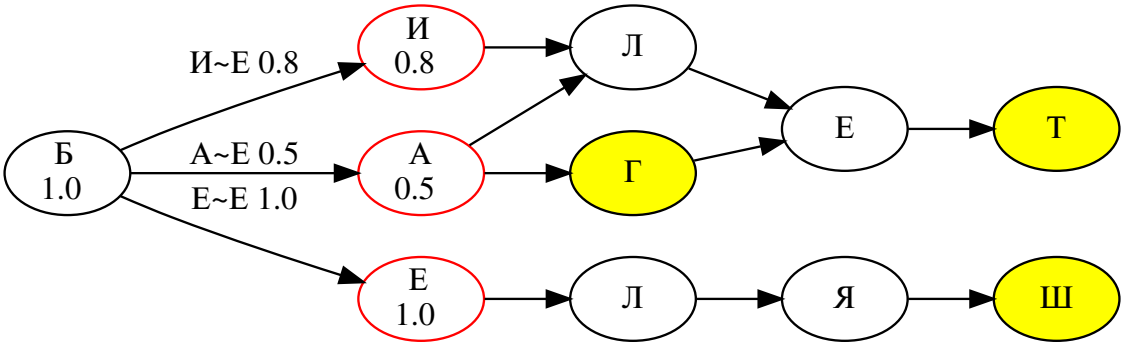

баг, багет, балет, билет, беляш

# Trie: пример



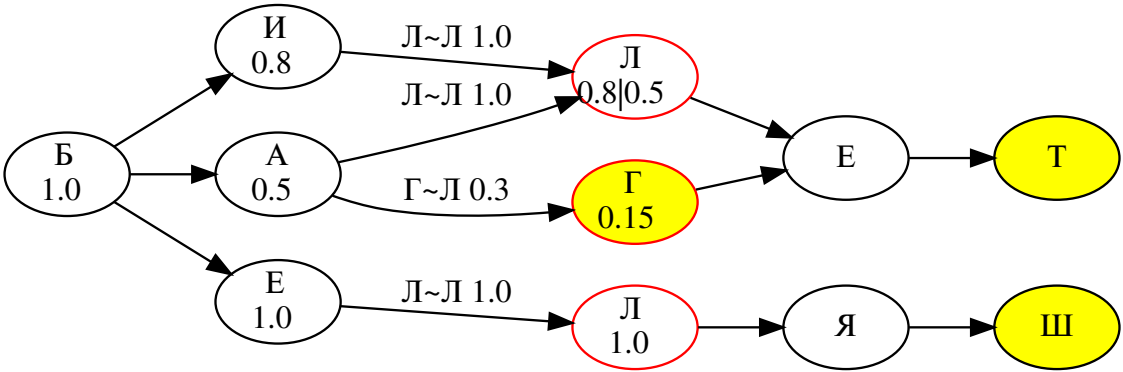

БЕЛЕТ

# Trie: пример




БЕЛЕТ

# Trie: пример




БЕЛЕТ