

## dictionaries and tolerant retrieval

Most slides are from Prof. Schütze, Center for Information and Language Processing, University of Munich

September 21, 2017

# Overview

- 1 Recap
- 2 Dictionaries
- 3 Wildcard queries
- 4 Edit distance
- 5 Spelling correction
- 6 Soundex

# Outline

- 1 Recap
- 2 Dictionaries
- 3 Wildcard queries
- 4 Edit distance
- 5 Spelling correction
- 6 Soundex

## Type/token distinction

- **Token** – an instance of a word or term occurring in a document
- **Type** – an equivalence class of tokens
- *In June, the dog likes to chase the cat in the barn.*
- 12 word tokens, 9 word types
- In(1) June(2) the(3) dog(4) likes(5) to(6) chase(7)  
[the] cat(8) [in] [the] barn(9).

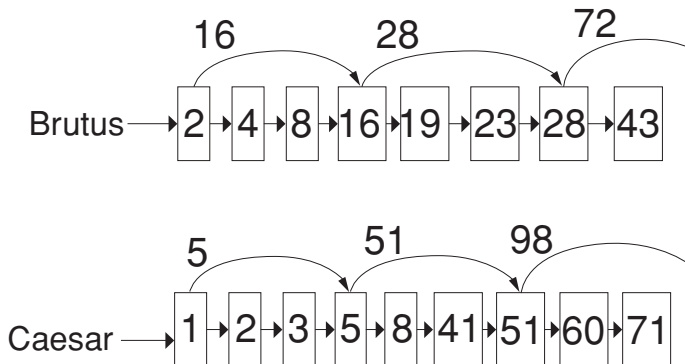
## Problems in tokenization

- What are the delimiters? Space? Apostrophe? Hyphen?
- For each of these: sometimes they delimit, sometimes they don't.
- No whitespace in many languages! (e.g., Chinese)
- No whitespace in Dutch, German, Swedish compounds (*Lebensversicherungsgesellschaftsangestellter*)

# Problems with equivalence classing

- A term is an equivalence class of tokens.
- How do we define equivalence classes?
- Numbers (3/20/91 vs. 20/3/91)
- Case folding
- Stemming, Porter stemmer
- Morphological analysis: inflectional vs. derivational
- Equivalence classing problems in other languages
  - More complex morphology than in English
  - Finnish: a single verb may have 12,000 different forms
  - Accents, umlauts

# Skip pointers



# Positional indexes

- **nonpositional** index: each posting is just a docID
- **positional** index: each posting is a docID and a **list of positions**
- Example query: *"to<sub>1</sub> be<sub>2</sub> or<sub>3</sub> not<sub>4</sub> to<sub>5</sub> be<sub>6</sub>"*

TO, 993427:

⟨ 1: ⟨7, 18, 33, 72, 86, 231⟩;  
2: ⟨1, 17, 74, 222, 255⟩;  
4: ⟨8, 16, 190, 429, 433⟩;  
5: ⟨363, 367⟩;  
7: ⟨13, 23, 191⟩; ...⟩

BE, 178239:

⟨ 1: ⟨17, 25⟩;  
4: ⟨17, 191, 291, 430, 434⟩;  
5: ⟨14, 19, 101⟩; ...⟩

Document 4 is a match!



## Positional indexes

- With a positional index, we can answer [phrase queries](#).
- With a positional index, we can answer [proximity queries](#).

# Take-away

- **Tolerant retrieval:** What to do if there is no exact match between query term and document term
- Wildcard queries
- Spelling correction

# Outline

- 1 Recap
- 2 Dictionaries**
- 3 Wildcard queries
- 4 Edit distance
- 5 Spelling correction
- 6 Soundex

# Inverted index

For each term  $t$ , we store a list of all documents that contain  $t$ .

BRUTUS → 1 2 4 11 31 45 173 174

CAESAR → 1 2 4 5 6 16 57 132 ...

CALPURNIA → 2 31 54 101

⋮

  
**dictionary**

  
**postings**

# Inverted index

For each term  $t$ , we store a list of all documents that contain  $t$ .

BRUTUS → 1 | 2 | 4 | 11 | 31 | 45 | 173 | 174

CAESAR → 1 | 2 | 4 | 5 | 6 | 16 | 57 | 132 | ...

CALPURNIA → 2 | 31 | 54 | 101

⋮

  
**dictionary**

  
**postings**

# Dictionaries

- The dictionary is the data structure for storing the term vocabulary.
- **Term vocabulary**: the **data**
- **Dictionary**: the **data structure** for storing the term vocabulary

## Dictionary as array of fixed-width entries

- For each term, we need to store a couple of items:
  - document frequency
  - pointer to postings list
  - ...
- Assume for the time being that we can store this information in a fixed-length entry.
- Assume that we store these entries in an array.

## Dictionary as array of fixed-width entries

| term   | document frequency | pointer to postings list |
|--------|--------------------|--------------------------|
| a      | 656,265            | →                        |
| aachen | 65                 | →                        |
| ...    | ...                | ...                      |
| zulu   | 221                | →                        |

space needed: 20 bytes 4 bytes 4 bytes

How do we look up a query term  $q_i$  in this array at query time?  
 That is: which data structure do we use to locate the entry (row) in the array where  $q_i$  is stored?



# Data structures for looking up term

- Two main classes of data structures: hashes and trees
- Some IR systems use hashes, some use trees.
- Criteria for when to use hashes vs. trees:
  - Is there a fixed number of terms or will it keep growing?
  - What are the relative frequencies with which various keys will be accessed?
  - How many terms are we likely to have?

# Hashes

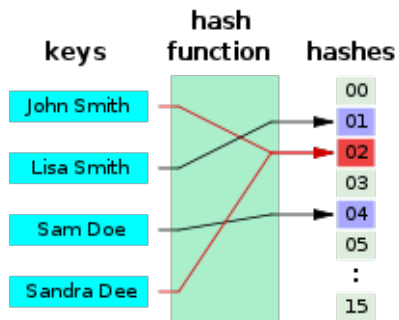


fig from wikipedia

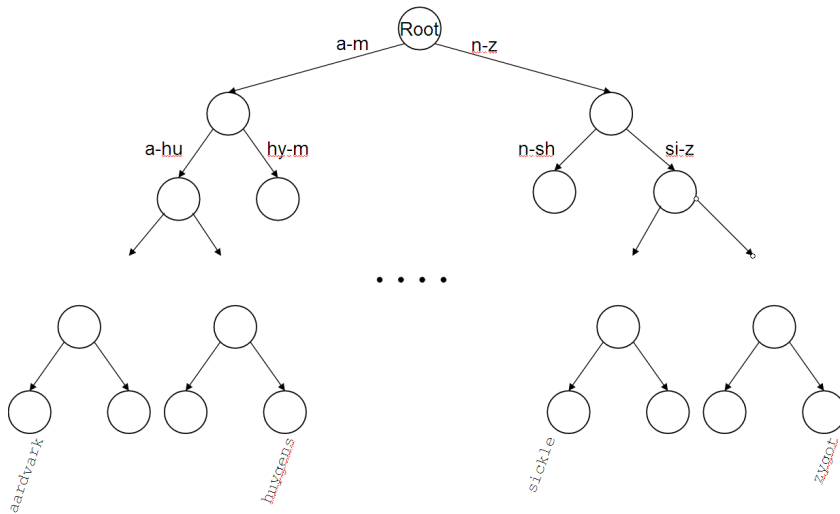
# Hashes

- Each vocabulary term is hashed into an integer, its row number in the array
- At query time: hash query term, locate entry in fixed-width array
- Pros: Lookup in a hash is faster than lookup in a tree.
  - Lookup time is constant.
- Cons
  - no way to find minor variants (*resume* vs. *résumé*)
  - no prefix search (all terms starting with *automat*)
  - need to rehash everything periodically if vocabulary keeps growing

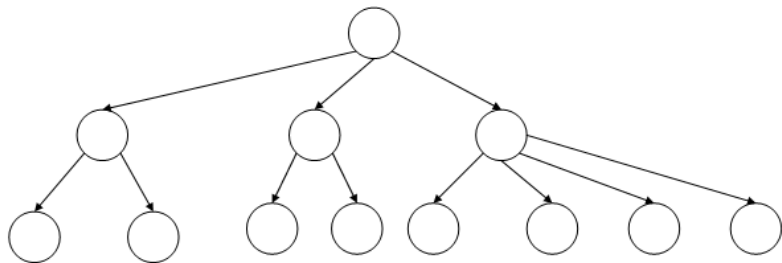
# Trees

- Trees solve the prefix problem (find all terms starting with *automat*).
- Simplest tree: binary tree
- Search is slightly slower than in hashes:  $O(\log M)$ , where  $M$  is the size of the vocabulary.
- $O(\log M)$  only holds for **balanced** trees.
- Rebalancing binary trees is expensive.
- **B-trees** mitigate the rebalancing problem.
- B-tree definition: every internal node has a number of children in the interval  $[a, b]$  where  $a, b$  are appropriate positive integers, e.g.,  $[2, 4]$ .

# Binary tree



# B-tree



- a generalization of binary search tree
- allow nodes to have more than 2 children

# Outline

- 1 Recap
- 2 Dictionaries
- 3 Wildcard queries**
- 4 Edit distance
- 5 Spelling correction
- 6 Soundex

## Wildcard queries

- $mon^*$ : find all docs containing any term beginning with *mon*
- Easy with B-tree dictionary: retrieve all terms  $t$  in the range:  $mon \leq t < moo$
- $*mon$ : find all docs containing any term ending with *mon*
  - Maintain an additional tree for terms *backwards*
  - Then retrieve all terms  $t$  in the range:  $nom \leq t < non$
- Result: A set of terms that are matches for wildcard query
- Then retrieve documents that contain any of these terms

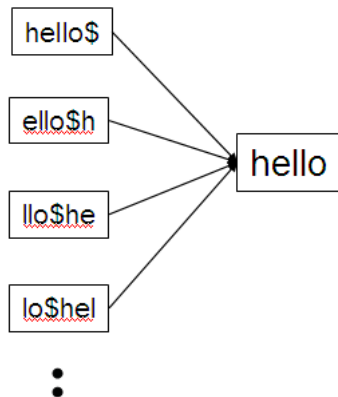


## How to handle \* in the middle of a term

- Example:  $m*n$
- We could look up  $m*$  and  $*n$  in the B-tree and intersect the two term sets.
- Expensive
- Alternative: [permuterm](#) index
- Basic idea: Rotate every wildcard query, so that the \* occurs at the end.
- Store each of these rotations in the dictionary, say, in a B-tree

# Permuterm index

- For term HELLO: add the following to the B-tree where \$ is a special symbol
  - *hello\$*,
  - *ello\$h*,
  - *llo\$he*,
  - *lo\$hel*,
  - *o\$hell*,
  - *\$hello*



# Permuterm index

- For HELLO, we've stored: *hello\$, ello\$h, llo\$he, lo\$hel, o\$hell, \$hello*
- Queries
  - For X, look up X\$
  - For X\*, look up \$X\*
  - For \*X, look up X\$\*
  - For \*X\*, look up X\*
  - For X\*Y, look up Y\$X\*
  - Example: For hel\*o, look up o\$hel\*
- Permuterm index would better be called a permuterm [tree](#).
- But permuterm index is the more common name.

## Processing a lookup in the permuterm index

- Rotate query wildcard to the right
- Use B-tree lookup as before
- Problem: Permuterm more than **quadruples** the size of the dictionary compared to a regular B-tree. (empirical number)

## *k*-gram indexes

- More space-efficient than permuterm index
- Enumerate all character *k*-grams (sequence of *k* characters) occurring in a term
- 2-grams are called **bigrams**.
  - april → ap pr ri il l\$
  - April is the cruelest month
  - → \$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\$
- \$ is a special word boundary symbol, as before.
- Maintain an inverted index from bigrams to the terms that contain the bigram

## why k-gram is more space efficient

- permuterm of hello  $\rightarrow$  hello\$, ello\$h, llo\$he, lo\$hel, o\$hell
- 2-grams of hello  $\rightarrow$  he, el, ll, lo, o\$
- 2-grams of low  $\rightarrow$  lo, ow, w\$

## why k-gram is more space efficient

- permuterm of hello  $\rightarrow$  hello\$, ello\$h, llo\$he, lo\$hel, o\$hell
- 2-grams of hello  $\rightarrow$  he, el, ll, lo, o\$
- 2-grams of low  $\rightarrow$  lo, ow, w\$

# Postings list in a 3-gram inverted index





## $k$ -gram (bigram, trigram, ...) indexes

- Note that we now have two different types of inverted indexes
- The term-document inverted index for finding documents based on a query consisting of terms
- The  $k$ -gram index for finding terms based on a query consisting of  $k$ -grams

## Processing wildcard terms in a bigram index

- Query `mon*` can now be run as:  
`$m AND mo AND on`
- Gets us all terms with the prefix *mon* ...
- ...but also many “false positives” like MOON.
- We must postfilter these terms against query.
- Surviving terms are then looked up in the term-document inverted index.
- *k*-gram index vs. permuterm index
  - *k*-gram index is more space efficient.
  - Permuterm index doesn't require postfiltering.

# Outline

- 1 Recap
- 2 Dictionaries
- 3 Wildcard queries
- 4 Edit distance**
- 5 Spelling correction
- 6 Soundex

# Edit distance

- The edit distance between string  $s_1$  and string  $s_2$  is the minimum number of basic operations that convert  $s_1$  to  $s_2$ .
- Levenshtein distance: The admissible basic operations are
  - insert, cost =1
  - delete, cost =1
  - replace, cost=1
  - copy, cost=0

|                        | s1  | s2   | operation      | cost |
|------------------------|-----|------|----------------|------|
|                        | dog | do   | insert         | 1    |
| • Levenshtein distance | cat | cart | insert         | 1    |
|                        | cat | cut  | replace        | 1    |
|                        | cat | act  | delete, insert | 2    |

## there are other distance definitions

- Damerau-Levenshtein distance *cat-act*: 1
- includes transposition operation.

## problem definition

- For two strings
  - X of length n
  - Y of length m
- $D(i,j)$ 
  - the edit distance between  $X[1..i]$  and  $Y[1..j]$
  - score of the best alignment from  $X[1..i]$  to  $Y[1..j]$
  - i.e., the first  $i$  characters of  $X$  and the first  $j$  characters of  $Y$
  - The edit distance between  $X$  and  $Y$  is thus  $D(n,m)$
- Properties for  $D(i,j)$ 
  - $D(i,0)=i$ ; delete  $i$  letters
  - $D(0,j)=j$ ; insert  $j$  letters

## recurrence relation

$$D(i, j) = \begin{cases} D(i-1, j-1) + d(X_i, Y_j); \text{ replace or copy} \\ D(i-1, j) + 1; \text{ insert} \\ D(i, j-1) + 1; \text{ delete} \end{cases} \quad (1)$$

$$d(x, y) = \begin{cases} 0; \text{ if } x=y \\ 1; \text{ otherwise} \end{cases} \quad (2)$$

# Edit distance using dynamic programming

- Dynamic programming: A tabular computation of  $D(n,m)$
- Solving problems by combining solutions to subproblems.
- Bottom-up
  - We compute  $D(i,j)$  for small  $i,j$
  - And compute larger  $D(i,j)$  based on previously computed smaller values
  - i.e., compute  $D(i,j)$  for all  $i$  ( $0 < i < n$ ) and  $j$  ( $0 < j < m$ )



# Levenshtein distance: Computation

|   |   |   |   |   |   |
|---|---|---|---|---|---|
|   |   | 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 |

## Levenshtein distance: Algorithm

LEVENSHTEINDISTANCE( $s_1, s_2$ )

```
1  for  $i \leftarrow 0$  to  $|s_1|$ 
2  do  $m[i, 0] = i$ 
3  for  $j \leftarrow 0$  to  $|s_2|$ 
4  do  $m[0, j] = j$ 
5  for  $i \leftarrow 1$  to  $|s_1|$ 
6  do for  $j \leftarrow 1$  to  $|s_2|$ 
7      do if  $s_1[i] = s_2[j]$ 
8          then  $m[i, j] = \min\{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]\}$ 
9          else  $m[i, j] = \min\{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1\}$ 
10 return  $m[|s_1|, |s_2|]$ 
```

Operations: insert (cost 1), delete (cost 1), replace (cost 1), copy (cost 0)

## Levenshtein distance: Algorithm

LEVENSHTEINDISTANCE( $s_1, s_2$ )

```
1  for  $i \leftarrow 0$  to  $|s_1|$ 
2  do  $m[i, 0] = i$ 
3  for  $j \leftarrow 0$  to  $|s_2|$ 
4  do  $m[0, j] = j$ 
5  for  $i \leftarrow 1$  to  $|s_1|$ 
6  do for  $j \leftarrow 1$  to  $|s_2|$ 
7      do if  $s_1[j] = s_2[j]$ 
8          then  $m[i, j] = \min\{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]\}$ 
9          else  $m[i, j] = \min\{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1\}$ 
10 return  $m[|s_1|, |s_2|]$ 
```

Operations: insert (cost 1), delete (cost 1), replace (cost 1), copy (cost 0)

## Levenshtein distance: Algorithm

LEVENSHTEINDISTANCE( $s_1, s_2$ )

```

1  for  $i \leftarrow 0$  to  $|s_1|$ 
2  do  $m[i, 0] = i$ 
3  for  $j \leftarrow 0$  to  $|s_2|$ 
4  do  $m[0, j] = j$ 
5  for  $i \leftarrow 1$  to  $|s_1|$ 
6  do for  $j \leftarrow 1$  to  $|s_2|$ 
7      do if  $s_1[i] = s_2[j]$ 
8          then  $m[i, j] = \min\{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]\}$ 
9          else  $m[i, j] = \min\{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1\}$ 
10 return  $m[|s_1|, |s_2|]$ 

```

Operations: insert (cost 1), delete (cost 1), replace (cost 1), copy (cost 0)

## Levenshtein distance: Algorithm

LEVENSHTEINDISTANCE( $s_1, s_2$ )

```
1  for  $i \leftarrow 0$  to  $|s_1|$ 
2  do  $m[i, 0] = i$ 
3  for  $j \leftarrow 0$  to  $|s_2|$ 
4  do  $m[0, j] = j$ 
5  for  $i \leftarrow 1$  to  $|s_1|$ 
6  do for  $j \leftarrow 1$  to  $|s_2|$ 
7     do if  $s_1[i] = s_2[j]$ 
8         then  $m[i, j] = \min\{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]\}$ 
9         else  $m[i, j] = \min\{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1\}$ 
10 return  $m[|s_1|, |s_2|]$ 
```

Operations: insert (cost 1), delete (cost 1), **replace (cost 1)**, copy (cost 0)

## Levenshtein distance: Algorithm

LEVENSHTEINDISTANCE( $s_1, s_2$ )

```
1  for  $i \leftarrow 0$  to  $|s_1|$ 
2  do  $m[i, 0] = i$ 
3  for  $j \leftarrow 0$  to  $|s_2|$ 
4  do  $m[0, j] = j$ 
5  for  $i \leftarrow 1$  to  $|s_1|$ 
6  do for  $j \leftarrow 1$  to  $|s_2|$ 
7      do if  $s_1[i] = s_2[j]$ 
8          then  $m[i, j] = \min\{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]\}$ 
9          else  $m[i, j] = \min\{m[i-1, j]+1, m[i, j-1]+1, m[i-1, j-1]+1\}$ 
10 return  $m[|s_1|, |s_2|]$ 
```

Operations: insert (cost 1), delete (cost 1), replace (cost 1), **copy** (cost 0)

## Levenshtein distance: Example

|   |                      | f                        | a                        | s                        | t                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| c | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 5</u><br><u>4 4</u> |
| a | <u>2</u><br><u>2</u> | <u>2 2</u><br><u>3 2</u> | <u>1 3</u><br><u>3 1</u> | <u>3 4</u><br><u>2 2</u> | <u>4 5</u><br><u>3 3</u> |
| t | <u>3</u><br><u>3</u> | <u>3 3</u><br><u>4 3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>2 4</u><br><u>3 2</u> |
| s | <u>4</u><br><u>4</u> | <u>4 4</u><br><u>5 4</u> | <u>4 3</u><br><u>5 3</u> | <u>2 3</u><br><u>4 2</u> | <u>3 3</u><br><u>3 3</u> |

## Each cell of Levenshtein matrix

|  |   |
|--|---|
| cost of getting here from my upper left neighbor (copy or replace) | cost of getting here from my upper neighbor (delete)                            |
| cost of getting here from my left neighbor (insert)                | the minimum of the three possible “movements”; the cheapest way of getting here |



## Levenshtein distance: Example

|   |  | f        | a          | s          | t          |            |
|---|--|----------|------------|------------|------------|------------|
|   |  | <b>0</b> | <b>1 1</b> | <b>2 2</b> | <b>3 3</b> | <b>4 4</b> |
| c |  | <b>1</b> | <b>1 2</b> | <b>2 3</b> | <b>3 4</b> | <b>4 5</b> |
|   |  | <b>1</b> | <b>2 1</b> | <b>2 2</b> | <b>3 3</b> | <b>4 4</b> |
| a |  | <b>2</b> | <b>2 2</b> | <b>1 3</b> | <b>3 4</b> | <b>4 5</b> |
|   |  | <b>2</b> | <b>3 2</b> | <b>3 1</b> | <b>2 2</b> | <b>3 3</b> |
| t |  | <b>3</b> | <b>3 3</b> | <b>3 2</b> | <b>2 3</b> | <b>2 4</b> |
|   |  | <b>3</b> | <b>4 3</b> | <b>4 2</b> | <b>3 2</b> | <b>3 2</b> |
| s |  | <b>4</b> | <b>4 4</b> | <b>4 3</b> | <b>2 3</b> | <b>3 3</b> |
|   |  | <b>4</b> | <b>5 4</b> | <b>5 3</b> | <b>4 2</b> | <b>3 3</b> |

## Dynamic programming (Cormen et al.)

- Optimal substructure: The optimal solution to the problem contains within it **subsolutions**, i.e., optimal solutions to subproblems.
- Overlapping subsolutions: The subsolutions overlap. These subsolutions are computed over and over again when computing the global optimal solution in a brute-force algorithm.
- Subproblem in the case of edit distance: what is the edit distance of two prefixes
- Overlapping subsolutions: We need most distances of prefixes 3 times – this corresponds to moving right, diagonally, down.

## Weighted edit distance

- As above, but weight of an operation depends on the characters involved.
- Meant to capture keyboard errors, e.g.,  $m$  more likely to be mistyped as  $n$  than as  $q$ .
- Therefore, replacing  $m$  by  $n$  is a smaller edit distance than by  $q$ .
- We now require a weight matrix as input.
- Modify dynamic programming to handle weights

sub[X, Y] = Substitution of X (incorrect) for Y (correct)

| X | Y (correct) |    |    |    |     |   |    |    |     |   |   |    |    |     |    |    |   |    |    |    |    |   |    |   |    |   |
|---|-------------|----|----|----|-----|---|----|----|-----|---|---|----|----|-----|----|----|---|----|----|----|----|---|----|---|----|---|
|   | a           | b  | c  | d  | e   | f | g  | h  | i   | j | k | l  | m  | n   | o  | p  | q | r  | s  | t  | u  | v | w  | x | y  | z |
| a | 0           | 0  | 7  | 1  | 342 | 0 | 0  | 2  | 118 | 0 | 1 | 0  | 0  | 3   | 76 | 0  | 0 | 1  | 35 | 9  | 9  | 0 | 1  | 0 | 5  | 0 |
| b | 0           | 0  | 9  | 9  | 2   | 2 | 3  | 1  | 0   | 0 | 0 | 5  | 11 | 5   | 0  | 10 | 0 | 0  | 2  | 1  | 0  | 0 | 8  | 0 | 0  | 0 |
| c | 6           | 5  | 0  | 16 | 0   | 9 | 5  | 0  | 0   | 0 | 1 | 0  | 7  | 9   | 1  | 10 | 2 | 5  | 39 | 40 | 1  | 3 | 7  | 1 | 1  | 0 |
| d | 1           | 10 | 13 | 0  | 12  | 0 | 5  | 5  | 0   | 0 | 2 | 3  | 7  | 3   | 0  | 1  | 0 | 43 | 30 | 22 | 0  | 0 | 4  | 0 | 2  | 0 |
| e | 388         | 0  | 3  | 11 | 0   | 2 | 2  | 0  | 89  | 0 | 0 | 3  | 0  | 5   | 93 | 0  | 0 | 14 | 12 | 6  | 15 | 0 | 1  | 0 | 18 | 0 |
| f | 0           | 15 | 0  | 3  | 1   | 0 | 5  | 2  | 0   | 0 | 0 | 3  | 4  | 1   | 0  | 0  | 0 | 6  | 4  | 12 | 0  | 0 | 2  | 0 | 0  | 0 |
| g | 4           | 1  | 11 | 11 | 9   | 2 | 0  | 0  | 0   | 1 | 1 | 3  | 0  | 0   | 2  | 1  | 3 | 5  | 13 | 21 | 0  | 0 | 1  | 0 | 3  | 0 |
| h | 1           | 8  | 0  | 3  | 0   | 0 | 0  | 0  | 0   | 0 | 2 | 0  | 12 | 14  | 2  | 3  | 0 | 3  | 1  | 11 | 0  | 0 | 2  | 0 | 0  | 0 |
| i | 103         | 0  | 0  | 0  | 146 | 0 | 1  | 0  | 0   | 0 | 0 | 6  | 0  | 0   | 49 | 0  | 0 | 0  | 2  | 1  | 47 | 0 | 2  | 1 | 15 | 0 |
| j | 0           | 1  | 1  | 9  | 0   | 0 | 1  | 0  | 0   | 0 | 0 | 2  | 1  | 0   | 0  | 0  | 0 | 0  | 5  | 0  | 0  | 0 | 0  | 0 | 0  | 0 |
| k | 1           | 2  | 8  | 4  | 1   | 1 | 2  | 5  | 0   | 0 | 0 | 0  | 5  | 0   | 2  | 0  | 0 | 0  | 6  | 0  | 0  | 0 | 4  | 0 | 0  | 3 |
| l | 2           | 10 | 1  | 4  | 0   | 4 | 5  | 6  | 13  | 0 | 1 | 0  | 0  | 14  | 2  | 5  | 0 | 11 | 10 | 2  | 0  | 0 | 0  | 0 | 0  | 0 |
| m | 1           | 3  | 7  | 8  | 0   | 2 | 0  | 6  | 0   | 0 | 4 | 4  | 0  | 180 | 0  | 6  | 0 | 0  | 9  | 15 | 13 | 3 | 2  | 2 | 3  | 0 |
| n | 2           | 7  | 6  | 5  | 3   | 0 | 1  | 19 | 1   | 0 | 4 | 35 | 78 | 0   | 0  | 7  | 0 | 28 | 5  | 7  | 0  | 0 | 1  | 2 | 0  | 2 |
| o | 91          | 1  | 1  | 3  | 116 | 0 | 0  | 0  | 25  | 0 | 2 | 0  | 0  | 0   | 0  | 14 | 0 | 2  | 4  | 14 | 39 | 0 | 0  | 0 | 18 | 0 |
| p | 0           | 11 | 1  | 2  | 0   | 6 | 5  | 0  | 2   | 9 | 0 | 2  | 7  | 6   | 15 | 0  | 0 | 1  | 3  | 6  | 0  | 4 | 1  | 0 | 0  | 0 |
| q | 0           | 0  | 1  | 0  | 0   | 0 | 27 | 0  | 0   | 0 | 0 | 0  | 0  | 0   | 0  | 0  | 0 | 0  | 0  | 0  | 0  | 0 | 0  | 0 | 0  | 0 |
| r | 0           | 14 | 0  | 30 | 12  | 2 | 2  | 8  | 2   | 0 | 5 | 8  | 4  | 20  | 1  | 14 | 0 | 0  | 12 | 22 | 4  | 0 | 0  | 1 | 0  | 0 |
| s | 11          | 8  | 27 | 33 | 35  | 4 | 0  | 1  | 0   | 1 | 0 | 27 | 0  | 6   | 1  | 7  | 0 | 14 | 0  | 15 | 0  | 0 | 5  | 3 | 20 | 1 |
| t | 3           | 4  | 9  | 42 | 7   | 5 | 19 | 5  | 0   | 1 | 0 | 14 | 9  | 5   | 5  | 6  | 0 | 11 | 37 | 0  | 0  | 2 | 19 | 0 | 7  | 6 |
| u | 20          | 0  | 0  | 0  | 44  | 0 | 0  | 0  | 64  | 0 | 0 | 0  | 0  | 2   | 43 | 0  | 0 | 4  | 0  | 0  | 0  | 2 | 0  | 0 | 8  | 0 |
| v | 0           | 0  | 7  | 0  | 0   | 3 | 0  | 0  | 0   | 0 | 0 | 1  | 0  | 0   | 1  | 0  | 0 | 0  | 8  | 3  | 0  | 0 | 0  | 0 | 0  | 0 |
| w | 2           | 2  | 1  | 0  | 1   | 0 | 0  | 2  | 0   | 0 | 1 | 0  | 0  | 0   | 0  | 7  | 0 | 6  | 3  | 3  | 1  | 0 | 0  | 0 | 0  | 0 |
| x | 0           | 0  | 0  | 2  | 0   | 0 | 0  | 0  | 0   | 0 | 0 | 0  | 0  | 0   | 0  | 0  | 0 | 0  | 9  | 0  | 0  | 0 | 0  | 0 | 0  | 0 |
| y | 0           | 0  | 2  | 0  | 15  | 0 | 1  | 7  | 15  | 0 | 0 | 0  | 2  | 0   | 6  | 1  | 0 | 7  | 36 | 8  | 5  | 0 | 0  | 1 | 0  | 0 |
| z | 0           | 0  | 0  | 7  | 0   | 0 | 0  | 0  | 0   | 0 | 0 | 7  | 5  | 0   | 0  | 0  | 0 | 2  | 21 | 3  | 0  | 0 | 0  | 0 | 3  | 0 |

## Using edit distance for spelling correction

- Given query, first enumerate all character sequences within a preset (possibly weighted) edit distance
- Intersect this set with our list of “correct” words
- Then suggest terms in the intersection to the user.
- → exercise in a few slides

# Exercise

- 1 Compute Levenshtein distance matrix for OSLO – SNOW
- 2 What are the Levenshtein editing operations that transform *cat* into *catcat*?

|   |                      | s          | n          | o          | w          |
|---|----------------------|------------|------------|------------|------------|
|   | <u>0</u>             | <u>1 1</u> | <u>2 2</u> | <u>3 3</u> | <u>4 4</u> |
| o | <u>1</u><br><u>1</u> |            |            |            |            |
| s | <u>2</u><br><u>2</u> |            |            |            |            |
| l | <u>3</u><br><u>3</u> |            |            |            |            |
| o | <u>4</u><br><u>4</u> |            |            |            |            |

|   |                   | s                      | n                   | o                   | w                   |
|---|-------------------|------------------------|---------------------|---------------------|---------------------|
|   | <u>  </u><br>0    | <u>  1  </u><br>1 1    | <u>  2  </u><br>2 2 | <u>  3  </u><br>3 3 | <u>  4  </u><br>4 4 |
| o | <u>  1  </u><br>1 | <u>  1  2  </u><br>2 ? |                     |                     |                     |
| s | <u>  2  </u><br>2 |                        |                     |                     |                     |
| l | <u>  3  </u><br>3 |                        |                     |                     |                     |
| o | <u>  4  </u><br>4 |                        |                     |                     |                     |



|   |                      | s                        | n          | o          | w          |
|---|----------------------|--------------------------|------------|------------|------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u> | <u>3 3</u> | <u>4 4</u> |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> |            |            |            |
| s | <u>2</u><br><u>2</u> |                          |            |            |            |
| l | <u>3</u><br><u>3</u> |                          |            |            |            |
| o | <u>4</u><br><u>4</u> |                          |            |            |            |

|   |                      | s                        | n                        | o          | w          |
|---|----------------------|--------------------------|--------------------------|------------|------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u> | <u>4 4</u> |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 ?</u> |            |            |
| s | <u>2</u><br><u>2</u> |                          |                          |            |            |
| l | <u>3</u><br><u>3</u> |                          |                          |            |            |
| o | <u>4</u><br><u>4</u> |                          |                          |            |            |

|   |                      | s                        | n                        | o          | w          |
|---|----------------------|--------------------------|--------------------------|------------|------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u> | <u>4 4</u> |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> |            |            |
| s | <u>2</u><br><u>2</u> |                          |                          |            |            |
| l | <u>3</u><br><u>3</u> |                          |                          |            |            |
| o | <u>4</u><br><u>4</u> |                          |                          |            |            |

|   |                      | s                        | n                        | o                        | w          |
|---|----------------------|--------------------------|--------------------------|--------------------------|------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u> |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 ?</u> |            |
| s | <u>2</u><br><u>2</u> |                          |                          |                          |            |
| l | <u>3</u><br><u>3</u> |                          |                          |                          |            |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |            |

|   |                      | s                        | n                        | o                        | w          |
|---|----------------------|--------------------------|--------------------------|--------------------------|------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u> |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> |            |
| s | <u>2</u><br><u>2</u> |                          |                          |                          |            |
| l | <u>3</u><br><u>3</u> |                          |                          |                          |            |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |            |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 ?</u> |
| s | <u>2</u><br><u>2</u> |                          |                          |                          |                          |
| l | <u>3</u><br><u>3</u> |                          |                          |                          |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> |                          |                          |                          |                          |
| l | <u>3</u><br><u>3</u> |                          |                          |                          |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 ?</u> |                          |                          |                          |
| l | <u>3</u><br><u>3</u> |                          |                          |                          |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |



|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> |                          |                          |                          |
| l | <u>3</u><br><u>3</u> |                          |                          |                          |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 ?</u> |                          |                          |
| l | <u>3</u><br><u>3</u> |                          |                          |                          |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> |                          |                          |
| l | <u>3</u><br><u>3</u> |                          |                          |                          |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 ?</u> |                          |
| l | <u>3</u><br><u>3</u> |                          |                          |                          |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> |                          |
| l | <u>3</u><br><u>3</u> |                          |                          |                          |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 ?</u> |
| l | <u>3</u><br><u>3</u> |                          |                          |                          |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> |                          |                          |                          |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 ?</u> |                          |                          |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |



|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> |                          |                          |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |  |                      | s                        | n                        | o                        | w                        |
|---|--|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   |  | <b>0</b>             | <b>1 1</b>               | <b>2 2</b>               | <b>3 3</b>               | <b>4 4</b>               |
| o |  | <b>1</b><br><b>1</b> | <b>1 2</b><br><b>2 1</b> | <b>2 3</b><br><b>2 2</b> | <b>2 4</b><br><b>3 2</b> | <b>4 5</b><br><b>3 3</b> |
| s |  | <b>2</b><br><b>2</b> | <b>1 2</b><br><b>3 1</b> | <b>2 3</b><br><b>2 2</b> | <b>3 3</b><br><b>3 3</b> | <b>3 4</b><br><b>4 3</b> |
| l |  | <b>3</b><br><b>3</b> | <b>3 2</b><br><b>4 2</b> | <b>2 3</b><br><b>3 ?</b> |                          |                          |
| o |  | <b>4</b><br><b>4</b> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> |                          |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 ?</u> |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> |                          |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 ?</u> |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 4</u> |
| o | <u>4</u><br><u>4</u> |                          |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 4</u> |
| o | <u>4</u><br><u>4</u> | <u>4 3</u><br><u>5 ?</u> |                          |                          |                          |



|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 4</u> |
| o | <u>4</u><br><u>4</u> | <u>4 3</u><br><u>5 3</u> |                          |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 4</u> |
| o | <u>4</u><br><u>4</u> | <u>4 3</u><br><u>5 3</u> | <u>3 3</u><br><u>4 ?</u> |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 4</u> |
| o | <u>4</u><br><u>4</u> | <u>4 3</u><br><u>5 3</u> | <u>3 3</u><br><u>4 3</u> |                          |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 4</u> |
| o | <u>4</u><br><u>4</u> | <u>4 3</u><br><u>5 3</u> | <u>3 3</u><br><u>4 3</u> | <u>2 4</u><br><u>4 ?</u> |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 4</u> |
| o | <u>4</u><br><u>4</u> | <u>4 3</u><br><u>5 3</u> | <u>3 3</u><br><u>4 3</u> | <u>2 4</u><br><u>4 2</u> |                          |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 4</u> |
| o | <u>4</u><br><u>4</u> | <u>4 3</u><br><u>5 3</u> | <u>3 3</u><br><u>4 3</u> | <u>2 4</u><br><u>4 2</u> | <u>4 5</u><br><u>3 ?</u> |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 4</u> |
| o | <u>4</u><br><u>4</u> | <u>4 3</u><br><u>5 3</u> | <u>3 3</u><br><u>4 3</u> | <u>2 4</u><br><u>4 2</u> | <u>4 5</u><br><u>3 3</u> |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 4</u> |
| o | <u>4</u><br><u>4</u> | <u>4 3</u><br><u>5 3</u> | <u>3 3</u><br><u>4 3</u> | <u>2 4</u><br><u>4 2</u> | <u>4 5</u><br><u>3 3</u> |



|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 4</u> |
| o | <u>4</u><br><u>4</u> | <u>4 3</u><br><u>5 3</u> | <u>3 3</u><br><u>4 3</u> | <u>2 4</u><br><u>4 2</u> | <u>4 5</u><br><u>3 3</u> |

How do I read out the editing operations that transform OSLO into SNOW?

|   |            | s                        | n                        | o                        | w                        |
|---|------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>   | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1 1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2 2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3 3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 4</u> |
| o | <u>4 4</u> | <u>4 3</u><br><u>5 3</u> | <u>3 3</u><br><u>4 3</u> | <u>2 4</u><br><u>4 2</u> | <u>4 5</u><br><u>3 3</u> |

| cost | operation | input | output |
|------|-----------|-------|--------|
| 1    | insert    | *     | w      |

|   |  |        |            |            |            |            |
|---|--|--------|------------|------------|------------|------------|
|   |  |        | s          | n          | o          | w          |
|   |  | 0      | 1 1        | 2 2        | 3 3        | 4 4        |
| o |  | 1<br>1 | 1 2<br>2 1 | 2 3<br>2 2 | 2 4<br>3 2 | 4 5<br>3 3 |
| s |  | 2<br>2 | 1 2<br>3 1 | 2 3<br>2 2 | 3 3<br>3 3 | 3 4<br>4 3 |
| l |  | 3<br>3 | 3 2<br>4 2 | 2 3<br>3 2 | 3 4<br>3 3 | 4 4<br>4 4 |
| o |  | 4<br>4 | 4 3<br>5 3 | 3 3<br>4 3 | 2 4<br>4 2 | 4 5<br>3 3 |

| cost | operation | input | output |
|------|-----------|-------|--------|
| 0    | (copy)    | o     | o      |
| 1    | insert    | *     | w      |

|   |            | s                        | n                        | o                        | w                        |
|---|------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>   | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1 1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2 2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3 3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 4</u> |
| o | <u>4 4</u> | <u>4 3</u><br><u>5 3</u> | <u>3 3</u><br><u>4 3</u> | <u>2 4</u><br><u>4 2</u> | <u>4 5</u><br><u>3 3</u> |

| cost | operation | input | output |
|------|-----------|-------|--------|
| 1    | replace   | l     | n      |
| 0    | (copy)    | o     | o      |
| 1    | insert    | *     | w      |

|   |        | s          | n          | o          | w          |
|---|--------|------------|------------|------------|------------|
|   | 0      | 1 1        | 2 2        | 3 3        | 4 4        |
| o | 1<br>1 | 1 2<br>2 1 | 2 3<br>2 2 | 2 4<br>3 2 | 4 5<br>3 3 |
| s | 2<br>2 | 1 2<br>3 1 | 2 3<br>2 2 | 3 3<br>3 3 | 3 4<br>4 3 |
| l | 3<br>3 | 3 2<br>4 2 | 2 3<br>3 2 | 3 4<br>3 3 | 4 4<br>4 4 |
| o | 4<br>4 | 4 3<br>5 3 | 3 3<br>4 3 | 2 4<br>4 2 | 4 5<br>3 3 |

| cost | operation | input | output |
|------|-----------|-------|--------|
| 0    | (copy)    | s     | s      |
| 1    | replace   | l     | n      |
| 0    | (copy)    | o     | o      |
| 1    | insert    | *     | w      |

|   |                      | s                        | n                        | o                        | w                        |
|---|----------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|   | <u>0</u>             | <u>1 1</u>               | <u>2 2</u>               | <u>3 3</u>               | <u>4 4</u>               |
| o | <u>1</u><br><u>1</u> | <u>1 2</u><br><u>2 1</u> | <u>2 3</u><br><u>2 2</u> | <u>2 4</u><br><u>3 2</u> | <u>4 5</u><br><u>3 3</u> |
| s | <u>2</u><br><u>2</u> | <u>1 2</u><br><u>3 1</u> | <u>2 3</u><br><u>2 2</u> | <u>3 3</u><br><u>3 3</u> | <u>3 4</u><br><u>4 3</u> |
| l | <u>3</u><br><u>3</u> | <u>3 2</u><br><u>4 2</u> | <u>2 3</u><br><u>3 2</u> | <u>3 4</u><br><u>3 3</u> | <u>4 4</u><br><u>4 4</u> |
| o | <u>4</u><br><u>4</u> | <u>4 3</u><br><u>5 3</u> | <u>3 3</u><br><u>4 3</u> | <u>2 4</u><br><u>4 2</u> | <u>4 5</u><br><u>3 3</u> |

| cost | operation | input | output |
|------|-----------|-------|--------|
| 1    | delete    | o     | *      |
| 0    | (copy)    | s     | s      |
| 1    | replace   | l     | n      |
| 0    | (copy)    | o     | o      |
| 1    | insert    | *     | w      |

|   |  | c        |          | a        |          | t        |          | c        |          | a        |          | t        |          |          |
|---|--|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
|   |  | <b>0</b> | <b>1</b> | <b>1</b> | <b>2</b> | <b>2</b> | <b>3</b> | <b>3</b> | <b>4</b> | <b>4</b> | <b>5</b> | <b>5</b> | <b>6</b> | <b>6</b> |
| c |  | <b>1</b> | <b>0</b> | 2        | 2        | 3        | 3        | 4        | <b>3</b> | 5        | 5        | 6        | 6        | 7        |
|   |  | <b>1</b> | 2        | <b>0</b> | <b>1</b> | <b>1</b> | <b>2</b> | <b>2</b> | <b>3</b> | <b>3</b> | <b>4</b> | <b>4</b> | <b>5</b> | <b>5</b> |
| a |  | <b>2</b> | 2        | <b>1</b> | <b>0</b> | 2        | 2        | 3        | 3        | 4        | <b>3</b> | 5        | 5        | 6        |
|   |  | <b>2</b> | 3        | <b>1</b> | 2        | <b>0</b> | <b>1</b> | <b>1</b> | <b>2</b> | <b>2</b> | <b>3</b> | <b>3</b> | <b>4</b> | <b>4</b> |
| t |  | <b>3</b> | 3        | <b>2</b> | 2        | <b>1</b> | <b>0</b> | 2        | 2        | 3        | 3        | 4        | <b>3</b> | 5        |
|   |  | <b>3</b> | 4        | <b>2</b> | 3        | <b>1</b> | 2        | <b>0</b> | <b>1</b> | <b>1</b> | <b>2</b> | <b>2</b> | <b>3</b> | <b>3</b> |

|   |     |            |            |            |            |            |            |
|---|-----|------------|------------|------------|------------|------------|------------|
|   |     | c          | a          | t          | c          | a          | t          |
|   | 0   | 1 1        | 2 2        | 3 3        | 4 4        | 5 5        | 6 6        |
| c | 1 1 | 0 2<br>2 0 | 2 3<br>1 1 | 3 4<br>2 2 | 3 5<br>3 3 | 5 6<br>4 4 | 6 7<br>5 5 |
| a | 2 2 | 2 1<br>3 1 | 0 2<br>2 0 | 2 3<br>1 1 | 3 4<br>2 2 | 3 5<br>3 3 | 5 6<br>4 4 |
| t | 3 3 | 3 2<br>4 2 | 2 1<br>3 1 | 0 2<br>2 0 | 2 3<br>1 1 | 3 4<br>2 2 | 3 5<br>3 3 |

| cost | operation | input | output |
|------|-----------|-------|--------|
| 1    | insert    | *     | c      |
| 1    | insert    | *     | a      |
| 1    | insert    | *     | t      |
| 0    | (copy)    | c     | c      |
| 0    | (copy)    | a     | a      |
| 0    | (copy)    | t     | t      |



|   |        |            |            |            |            |            |            |
|---|--------|------------|------------|------------|------------|------------|------------|
|   |        | c          | a          | t          | c          | a          | t          |
|   | 0      | 1 1        | 2 2        | 3 3        | 4 4        | 5 5        | 6 6        |
| c | 1<br>1 | 0 2<br>2 0 | 2 3<br>1 1 | 3 4<br>2 2 | 3 5<br>3 3 | 5 6<br>4 4 | 6 7<br>5 5 |
| a | 2<br>2 | 2 1<br>3 1 | 0 2<br>2 0 | 2 3<br>1 1 | 3 4<br>2 2 | 3 5<br>3 3 | 5 6<br>4 4 |
| t | 3<br>3 | 3 2<br>4 2 | 2 1<br>3 1 | 0 2<br>2 0 | 2 3<br>1 1 | 3 4<br>2 2 | 3 5<br>3 3 |

| cost | operation | input | output |
|------|-----------|-------|--------|
| 0    | (copy)    | c     | c      |
| 1    | insert    | *     | a      |
| 1    | insert    | *     | t      |
| 1    | insert    | *     | c      |
| 0    | (copy)    | a     | a      |
| 0    | (copy)    | t     | t      |

|   |        |  |  |  |  |  |  |
|---|--------|--|--|--|--|--|--|
|   |        | c  | a  | t  | c  | a  | t  |
|   | 0      | 1 1  | 2 2  | 3 3  | 4 4  | 5 5  | 6 6  |
| c | 1<br>1 | <b>0</b>   <b>2</b><br><b>2</b>   <b>0</b> | 2 3<br>1 1                                 | 3 4<br>2 2                                 | <b>3</b>   <b>5</b><br><b>3</b>   <b>3</b> | 5 6<br>4 4                                 | 6 7<br>5 5                                 |
| a | 2<br>2 | 2 1<br>3 1                                 | <b>0</b>   <b>2</b><br><b>2</b>   <b>0</b> | <b>2</b>   <b>3</b><br><b>1</b>   <b>1</b> | <b>3</b>   <b>4</b><br><b>2</b>   <b>2</b> | <b>3</b>   <b>5</b><br><b>3</b>   <b>3</b> | 5 6<br>4 4                                 |
| t | 3<br>3 | 3 2<br>4 2                                 | 2 1<br>3 1                                 | <b>0</b>   2<br>2   <b>0</b>               | 2 3<br>1 1                                 | 3 4<br>2 2                                 | <b>3</b>   <b>5</b><br><b>3</b>   <b>3</b> |

| cost | operation | input | output |
|------|-----------|-------|--------|
| 0    | (copy)    | c     | c      |
| 0    | (copy)    | a     | a      |
| 1    | insert    | *     | t      |
| 1    | insert    | *     | c      |
| 1    | insert    | *     | a      |
| 0    | (copy)    | t     | t      |

|   |  |        |            |            |            |            |            |            |
|---|--|--------|------------|------------|------------|------------|------------|------------|
|   |  |        | c          | a          | t          | c          | a          | t          |
|   |  | 0      | 1 1        | 2 2        | 3 3        | 4 4        | 5 5        | 6 6        |
| c |  | 1<br>1 | 0 2<br>2 0 | 2 3<br>1 1 | 3 4<br>2 2 | 3 5<br>3 3 | 5 6<br>4 4 | 6 7<br>5 5 |
| a |  | 2<br>2 | 2 1<br>3 1 | 0 2<br>2 0 | 2 3<br>1 1 | 3 4<br>2 2 | 3 5<br>3 3 | 5 6<br>4 4 |
| t |  | 3<br>3 | 3 2<br>4 2 | 2 1<br>3 1 | 0 2<br>2 0 | 2 3<br>1 1 | 3 4<br>2 2 | 3 5<br>3 3 |

| cost | operation | input | output |
|------|-----------|-------|--------|
| 0    | (copy)    | c     | c      |
| 0    | (copy)    | a     | a      |
| 0    | (copy)    | t     | t      |
| 1    | insert    | *     | c      |
| 1    | insert    | *     | a      |
| 1    | insert    | *     | t      |

# Outline

- 1 Recap
- 2 Dictionaries
- 3 Wildcard queries
- 4 Edit distance
- 5 Spelling correction**
- 6 Soundex

# Spelling correction

- Two principal uses
  - Correcting documents being indexed
  - Correcting user queries
- Two different methods for spelling correction
  - **Isolated word** spelling correction
    - Check each word on its own for misspelling
    - Will not catch typos resulting in correctly spelled words, e.g.,  
*an asteroid that fell **form** the sky*
  - **Context-sensitive** spelling correction
    - Look at surrounding words
    - Can correct *form/from* error above

## Correcting documents

- We're not interested in interactive spelling correction of documents (e.g., MS Word) in this class.
- In IR, we use document correction primarily for OCR'ed documents. (OCR = optical character recognition)
- The general philosophy in IR is: don't change the documents.

## Correcting queries

- First: isolated word spelling correction
- Premise 1: There is a list of “correct words” from which the correct spellings come.
- Premise 2: We have a way of computing the **distance** between a misspelled word and a correct word.
- Simple spelling correction algorithm: return the “correct” word that has the smallest distance to the misspelled word.
- Example: *informaton* → *information*
- For the list of correct words, we can use the vocabulary of all words that occur in our collection.
- **Why is this problematic?**

## Alternatives to using the term vocabulary

- A standard dictionary (Webster's, OED etc.)
- An industry-specific dictionary (for specialized IR systems)
- The term vocabulary of the collection, appropriately weighted



# Distance between misspelled word and “correct” word

- Several alternatives
- Edit distance and Levenshtein distance
- Weighted edit distance
- $k$ -gram overlap

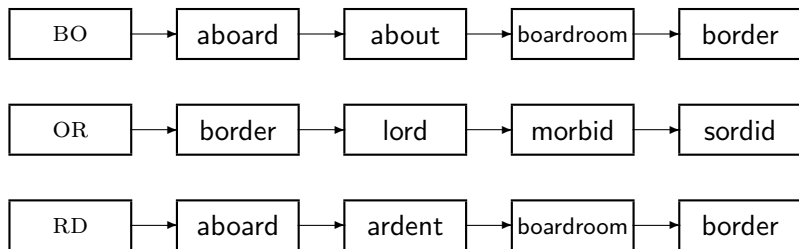
# Spelling correction

- Now that we can compute edit distance: how to use it for isolated word spelling correction – this is the last slide in this section.
- $k$ -gram indexes for isolated word spelling correction.
- Context-sensitive spelling correction
- General issues

## *k*-gram indexes for spelling correction

- Enumerate all *k*-grams in the query term
- Example: bigram index, misspelled word *bordroom*
- Bigrams: *bo, or, rd, dr, ro, oo, om*
- Use the *k*-gram index to retrieve “correct” words that match query term *k*-grams
- Threshold by number of matching *k*-grams
- E.g., only vocabulary terms that differ by at most 3 *k*-grams

## *k*-gram indexes for spelling correction: *bord*



$$BO \cap OR \cap RD = \{border\}$$

terms matched twice: aboard, boardroom

## Context-sensitive spelling correction

- Our example was: *an asteroid that fell **form** the sky*
- How can we correct *form* here?
- One idea: **hit-based** spelling correction
  - Retrieve “correct” terms close to each query term
  - for *flew form munich*: *flea* for *flew*, *from* for *form*, *munch* for *munich*
  - Now try all possible resulting phrases as queries with one word “fixed” at a time
  - Try query “*flea form munich*”
  - Try query “*flew from munich*”
  - Try query “*flew form munch*”
  - The correct query “*flew from munich*” has the most hits.
- Suppose we have 7 alternatives for *flew*, 20 for *form* and 3 for *munich*, how many “corrected” phrases will we enumerate?

# Context-sensitive spelling correction

- The “hit-based” algorithm we just outlined is not very efficient.
- More efficient alternative: look at “collection” of queries, not documents

# General issues in spelling correction

- User interface
  - automatic vs. suggested correction
  - *Did you mean* only works for one suggestion.
  - What about multiple possible corrections?
  - Tradeoff: simple vs. powerful UI
- Cost
  - Spelling correction is potentially expensive.
  - Avoid running on every query?
  - Maybe just on queries that match few documents.
  - Guess: Spelling correction of major search engines is efficient enough to be run on every query.

## Exercise: Understand Peter Norvig's spelling corrector

```
import re, collections
def words(text): return re.findall('[a-z]+', text.lower())
def train(features):
    model = collections.defaultdict(lambda: 1)
    for f in features:
        model[f] += 1
    return model
NWORDS = train(words(file('big.txt').read()))
alphabet = 'abcdefghijklmnopqrstuvwxyz'
def edits1(word):
    splits      = [(word[:i], word[i:]) for i in range(len(word) + 1)]
    deletes     = [a + b[1:] for a, b in splits if b]
    transposes  = [a + b[1] + b[0] + b[2:] for a, b in splits if len(b) > 1]
    replaces    = [a + c + b[1:] for a, b in splits for c in alphabet if b]
    inserts     = [a + c + b      for a, b in splits for c in alphabet]
    return set(deletes + transposes + replaces + inserts)
def known_edits2(word):
    return set(e2 for e1 in edits1(word) for e2 in edits1(e1) if e2 in NWORDS)
def known(words): return set(w for w in words if w in NWORDS)
def correct(word):
    candidates = known([word]) or known(edits1(word)) or known_edits2(word) or [word]
    return max(candidates, key=NWORDS.get)
```

<http://norvig.com/spell-correct.html>



# Outline

- 1 Recap
- 2 Dictionaries
- 3 Wildcard queries
- 4 Edit distance
- 5 Spelling correction
- 6 Soundex**

# Soundex

- Soundex is the basis for finding **phonetic** (as opposed to orthographic) alternatives.
- Example: *chebyshev* / *tchebyscheff*
- Algorithm:
  - Turn every token to be indexed into a 4-character reduced form
  - Do the same with query terms
  - Build and search an index on the reduced forms

# Soundex algorithm

- 1 Retain the first letter of the term.
- 2 Change all occurrences of the following letters to '0' (zero): A, E, I, O, U, H, W, Y
- 3 Change letters to digits as follows:
  - B, F, P, V to 1
  - C, G, J, K, Q, S, X, Z to 2
  - D, T to 3
  - L to 4
  - M, N to 5
  - R to 6
- 4 Repeatedly remove one out of each pair of consecutive identical digits
- 5 Remove all zeros from the resulting string; pad the resulting string with trailing zeros and return the first four positions, which will consist of a letter followed by three digits

## Example: Soundex of *HERMAN*

- Retain H
- *ERMAN* → *ORMON*
- *ORMON* → *06505*
- *06505* → *06505*
- *06505* → *655*
- Return *H655*
- Note: *HERMANN* will generate the same code
- *065055* → *06505* → *655*
- Retain the first letter of the term.
- A, E, I, O, U, H, W, Y → 0;
- Change letters to digits as follows:
  - B, F, P, V to 1
  - C, G, J, K, Q, S, X, Z to 2
  - D, T to 3
  - L to 4
  - M, N to 5
  - R to 6
- reduce consecutive identical digits
- remove zeros

## How useful is Soundex?

- Not very – for information retrieval
- Ok for “high recall” tasks in other applications (e.g., Interpol)
- Zobel and Dart (1996) suggest better alternatives for phonetic matching in IR.

## Exercise

- Compute Soundex code of your last name

# Recap

- fast access to the terms: hashing, B-tree
- Tolerant retrieval: What to do if there is no exact match between query term and document term
  - Wildcard queries.
  - Spelling correction. edit distance. dynamic programming algorithm.
  - k-gram index
  - soundex