Similarity and distance

CMT209
Informatics

Cardiff School of **Computer Science & Informatics**

http://www.cs.cf.ac.uk

CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD

Similarity

What is similarity?

"We know it if we see it."

Why does similarity matter?

- web search
- comparing documents
- grouping information
- recommendations
- dealing with typos
- ...

CARDIFF UNIVERSITY PRIFYSGOL CAERDYDD

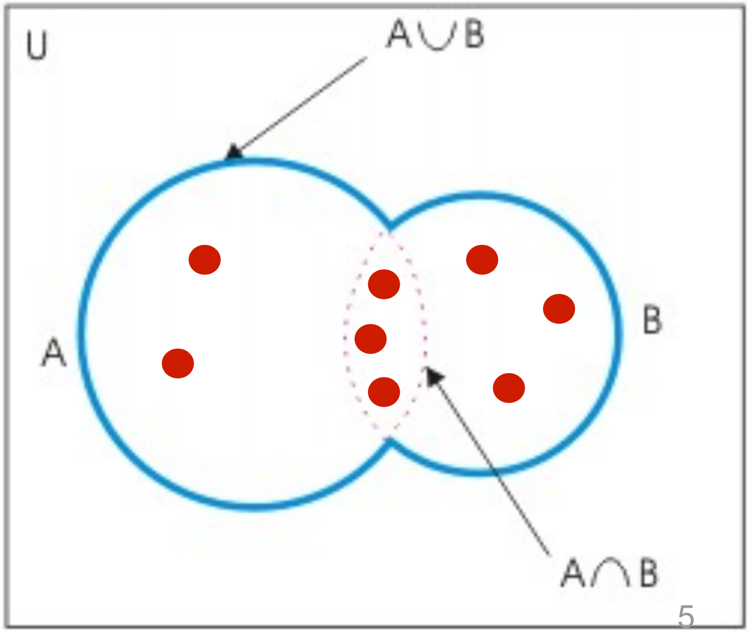
Comparing sets: Jaccard similarity

- Jaccard similarity of sets A and B is defined as the ratio of the size of the intersection of A and B to the size of their union, i.e.

$$sim(A, B) = |A \cap B| / |A \cup B|$$

- e.g. $sim(A, B) = 3 / (2 + 3 + 3)$
  $= 3 / 8$
  $= 0.375$

- note that $sim(A, B) \in [0, 1]$
  - $sim(A, B) = 0$ if $A \cap B = \varnothing$
  - $sim(A, B) = 1$ if $A = B$



CARDIFF UNIVERSITY PRIFYSGOL CAERDYDD

Document similarity

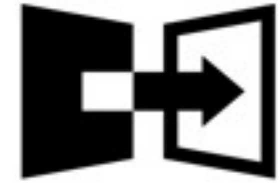
- an important class of problems that Jaccard similarity addresses well is that of finding textually similar documents in a large corpus such as the Web
- documents are represented as "bags" of words and we compare documents by measuring the overlap of their bag representations
- applications:
  - plagiarism
  - mirror pages
  - news aggregation

Plagiarism

- finding plagiarised documents tests our ability to find textual similarity
- the plagiariser may ...
  - copy some parts of a document
  - alter a few words
  - alter the order in which sentences appear

- yet the resulting document may still contain >50% of the original material
- comparing documents character by character will not detect sophisticated plagiarism, but Jaccard similarity can

Mirror pages



- it is common for an important or popular Web site to be duplicated at a number of hosts to improve its availability
- these mirror sites are quite similar, but are rarely identical
- e.g. they might contain information associated with their particular host
- it is important to be able to detect mirror pages, because search engines should avoid showing nearly identical pages within the first page of results

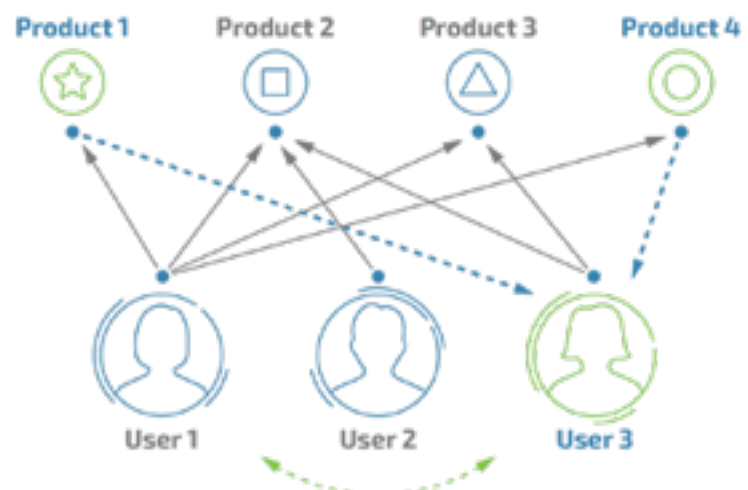
News aggregation

- the same news gets reported by different publishers
- each articles is somewhat different
- news aggregators, such as Google News, try to find all versions in order to group them together
- this requires finding Web pages that are textually similar, although not identical

CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD

Collaborative filtering

- the method of making automatic predictions (filtering) about the interests of a user by collecting taste information from many users (collaborating)
- the underlying assumption of collaborative filtering is that those who agreed in the past tend to agree again in the future, e.g.
  - online purchases
  - movie ratings
  - ...

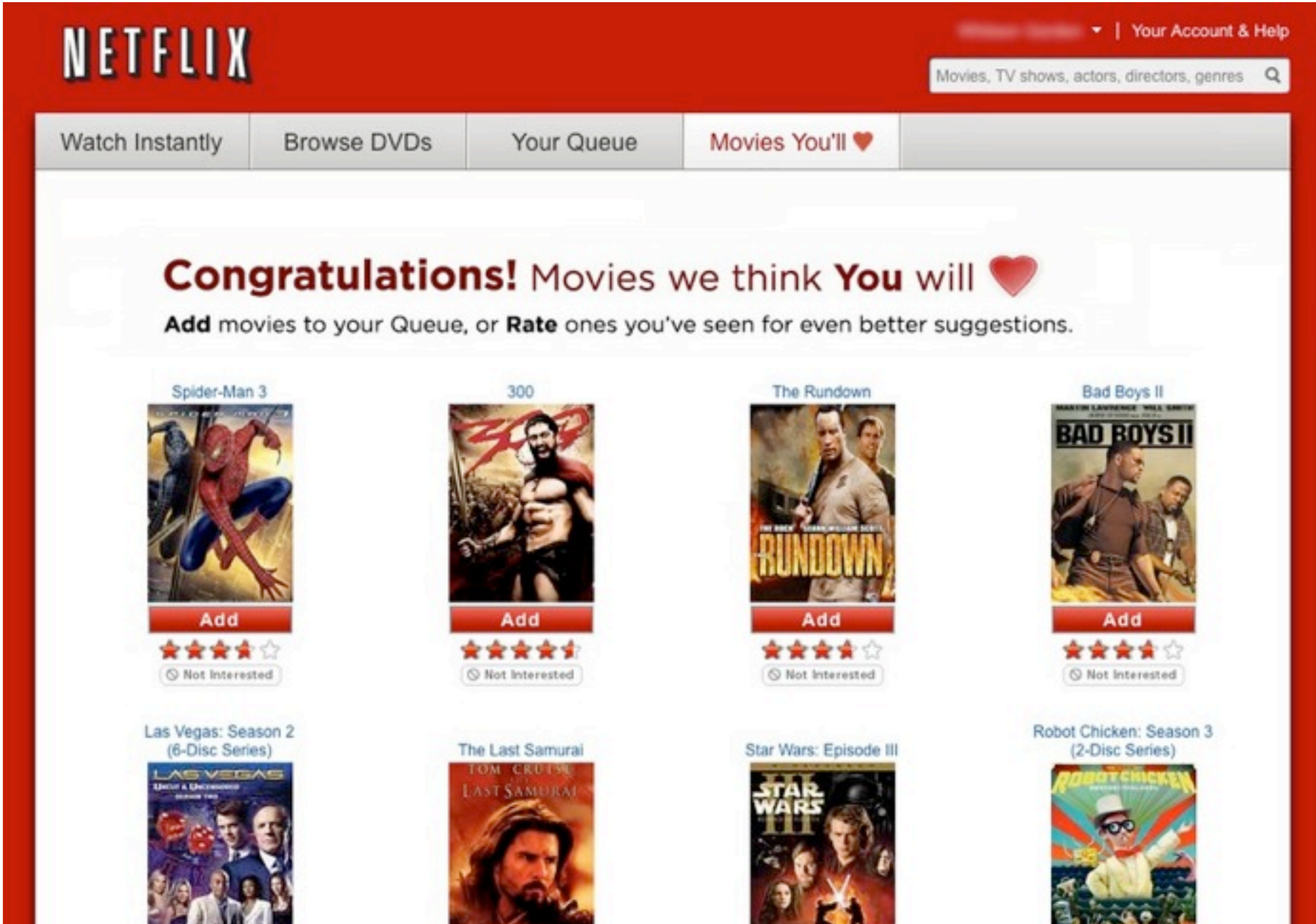


Online purchases

- Amazon has millions of customers and sells millions of products
- its database records which products have been bought by which customers
- we can say that two customers are similar if their sets of purchased products have a high Jaccard similarity
- likewise, two products that have sets of purchasers with high Jaccard similarity could be considered similar

Movie ratings

- NetFlix records:
  customer C watched movie M and gave rating R
- two movies are similar if many customers have seen both and have given similar ratings to them
- two customers are similar if they watched similar sets of movies and rated them similarly


NETFLIX
Your Account & Help
Movies, TV shows, actors, directors, genres
Watch Instantly
Browse DVDs
Your Queue
Movies You'll ♥
Congratulations! Movies we think You will ♥
Add movies to your Queue, or Rate ones you've seen for even better suggestions.
Spider-Man 3
300
The Rundown
Bad Boys II
Add
Not Interested
Las Vegas: Season 2 (6-Disc Series)
The Last Samurai
Star Wars: Episode III
Robot Chicken: Season 3 (2-Disc Series)

Distance

Distance vs. similarity

- similarity is measure of how close to each other two instances are
  - the closer the instances are to each other, the larger is the similarity value
- distance is also a measure of how close to each other two instances are
  - the closer the instances are to each other, the smaller is the distance value

Distance vs. similarity

- typically, given a similarity measure, one can "revert" it to serve as the distance measure and vice versa
- conversions may differ, e.g. if d is a distance measure, then one can use:

$$sim(x,y) = \frac{1}{d(x,y)} \quad \text{or} \quad sim(x,y) = \frac{1}{d(x,y)+0.5}$$

- if sim is the similarity measure that ranges between 0 and 1, then the corresponding distance measure can be defined as:

$$d(x,y) = 1 - sim(x,y)$$

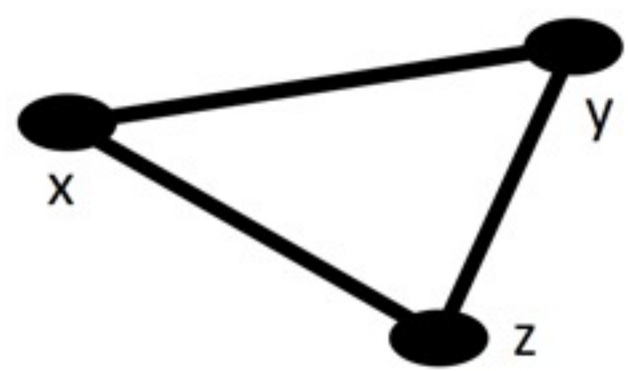
CARDIFF UNIVERSITY PRIFYSGOL CAERDYDD

Distance axioms

- formally, distance is a measure that satisfies the following conditions:

1. $d(x, y) \geq 0$	non-negativity
2. $d(x, y) = 0$ iff $x = y$	coincidence
3. $d(x, y) = d(y, x)$	symmetry
4. $d(x, z) \leq d(x, y) + d(y, z)$	triangle inequality

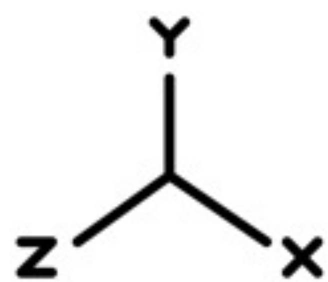
- these conditions express intuitive notions about the concept of distance



CARDIFF UNIVERSITY PRIFYSGOL CAERDYDD

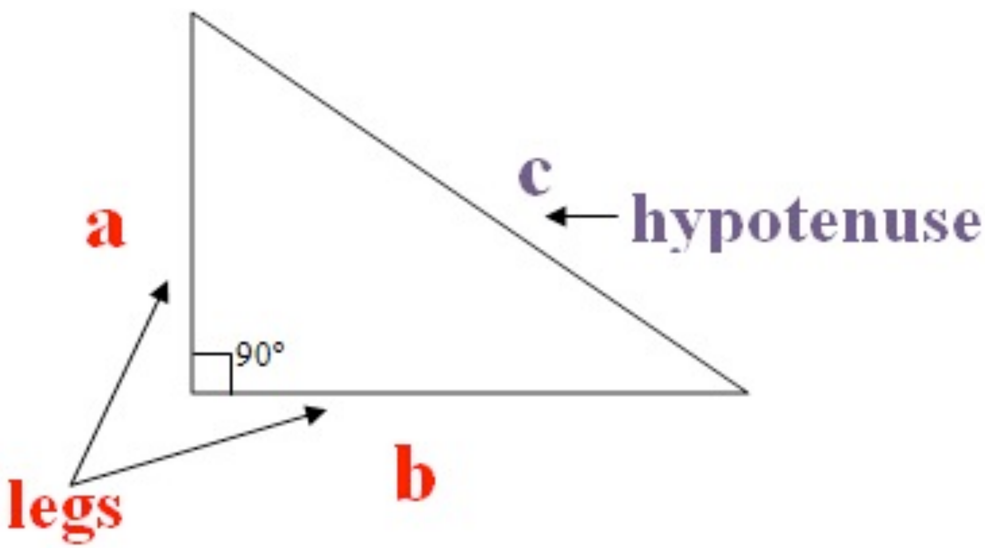
Euclidian distances

- the most familiar distance measure is the one we normally think of as "distance"
- an n-dimensional Euclidean space is one where points are vectors of n real numbers



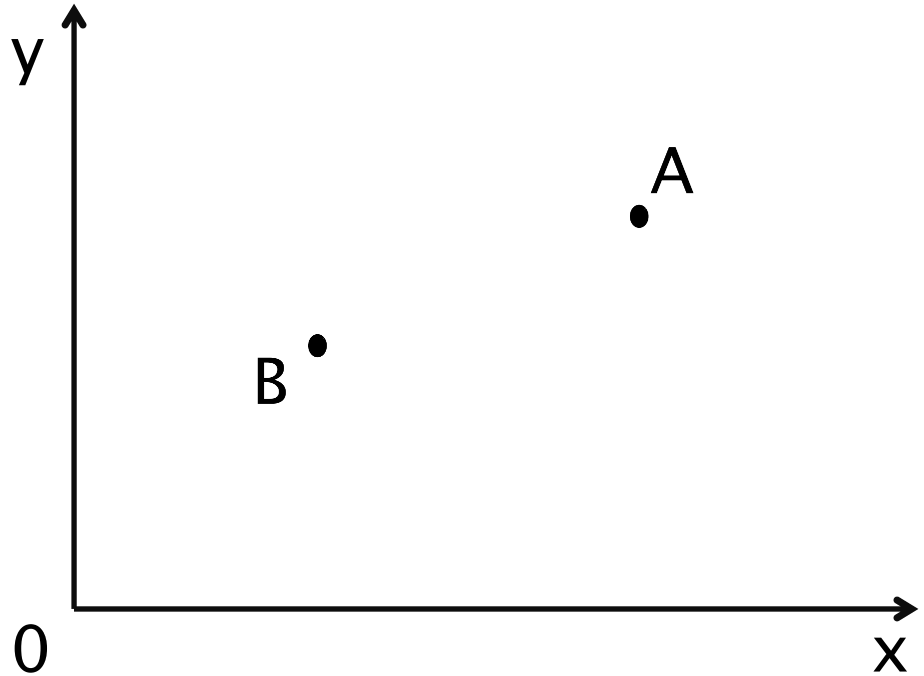
- e.g. a point in a 3D space is represented as (x, y, z)

Pythagorean theorem

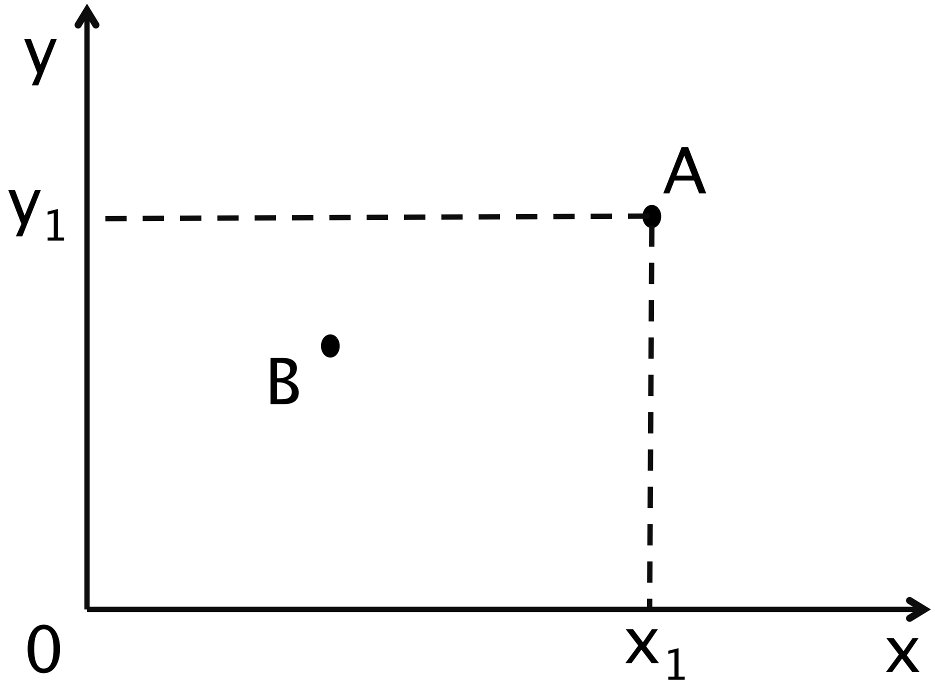


$$a^2 + b^2 = c^2$$

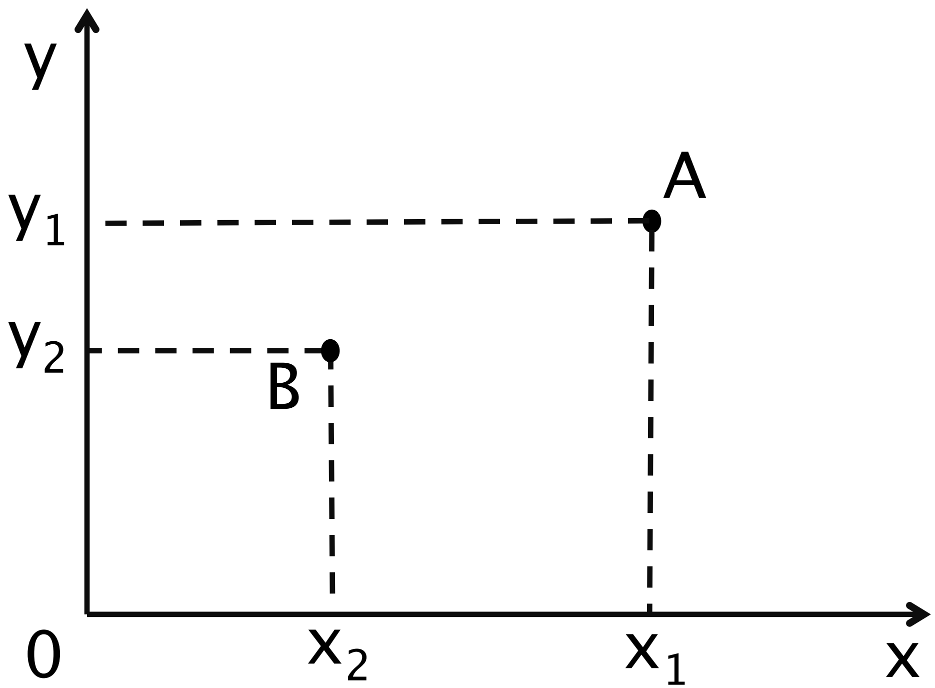
Computing Euclidean distance



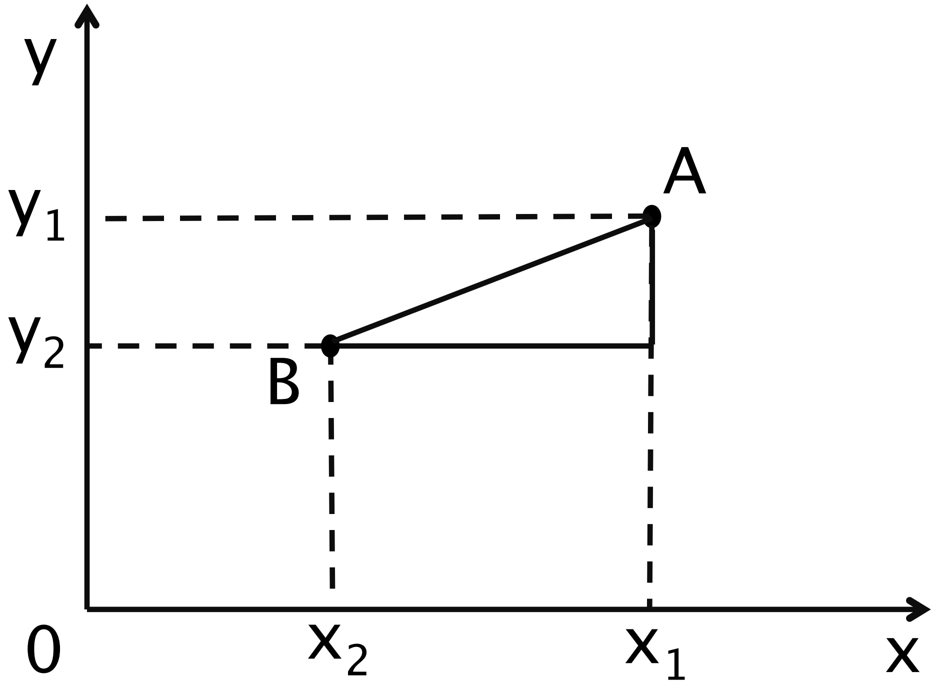
Computing Euclidean distance



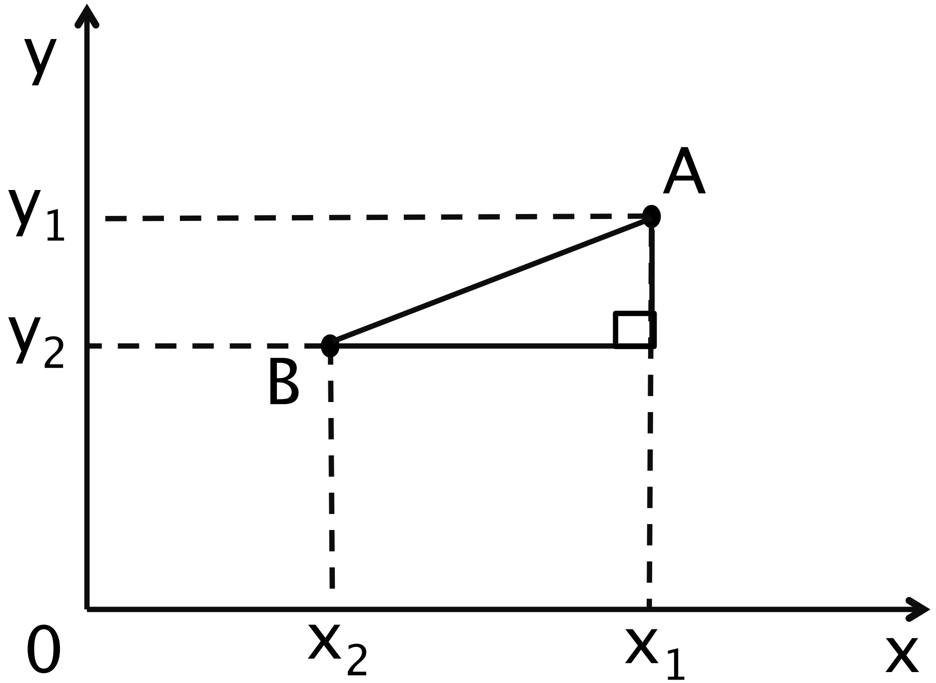
Computing Euclidean distance



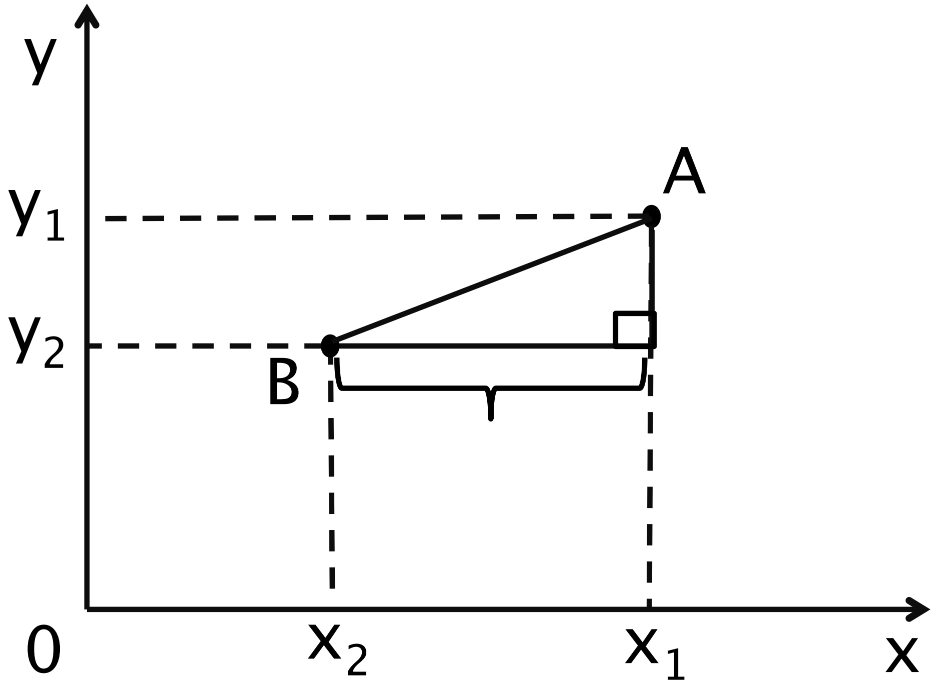
Computing Euclidean distance



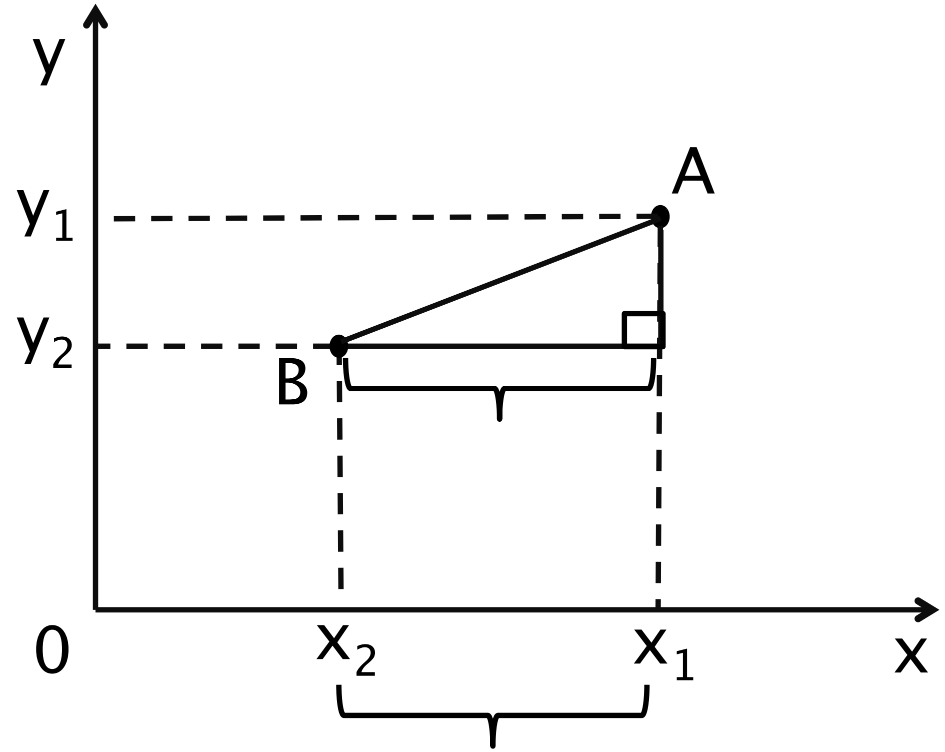
Computing Euclidean distance



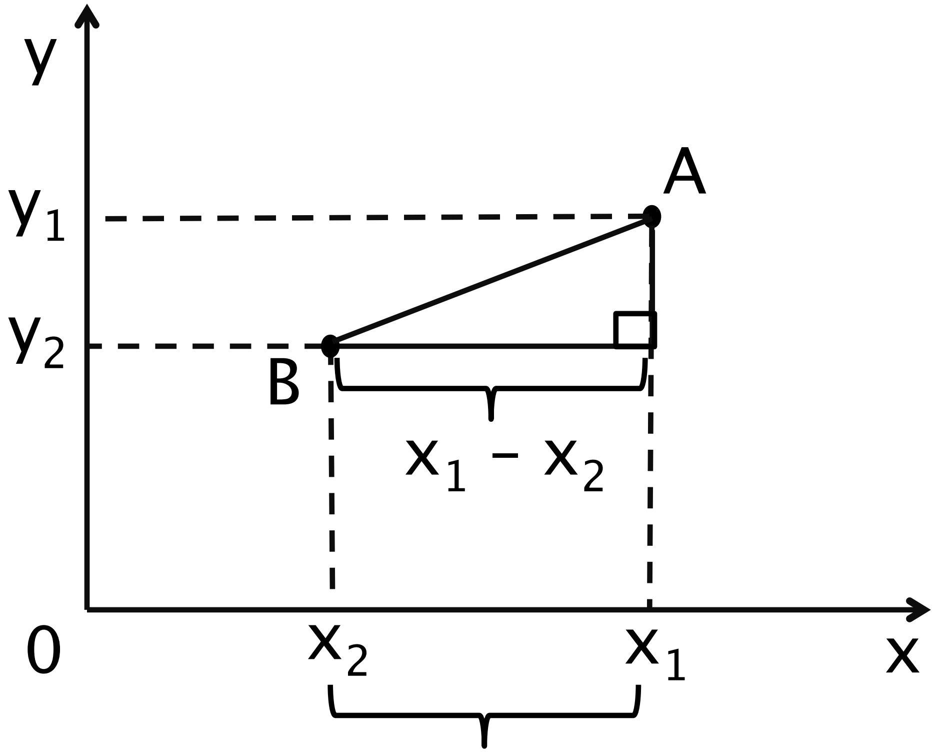
Computing Euclidean distance



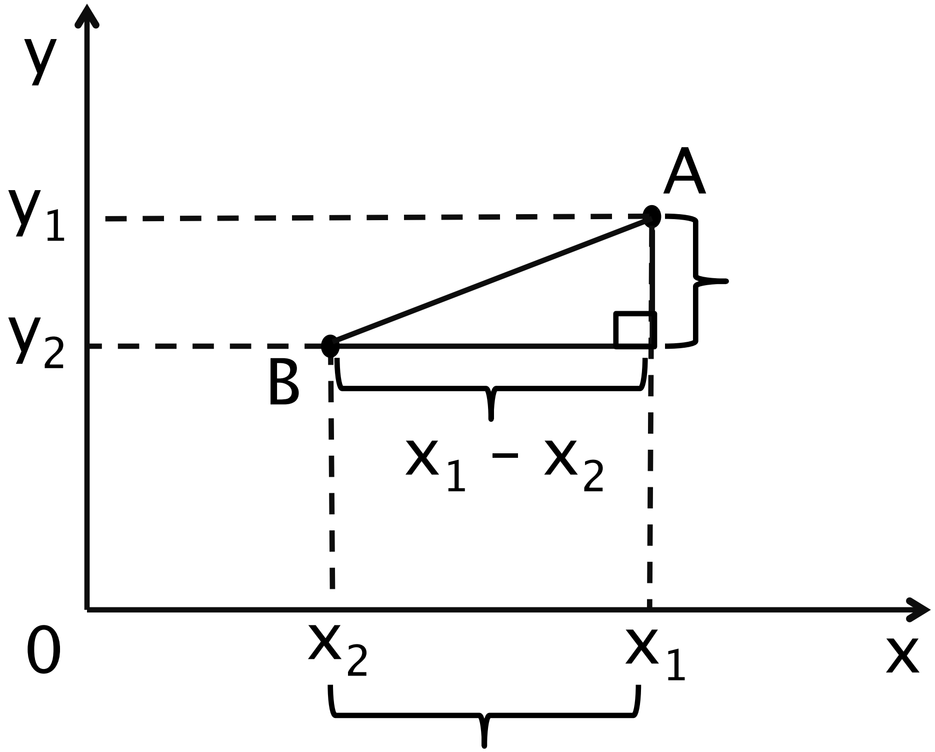
Computing Euclidean distance



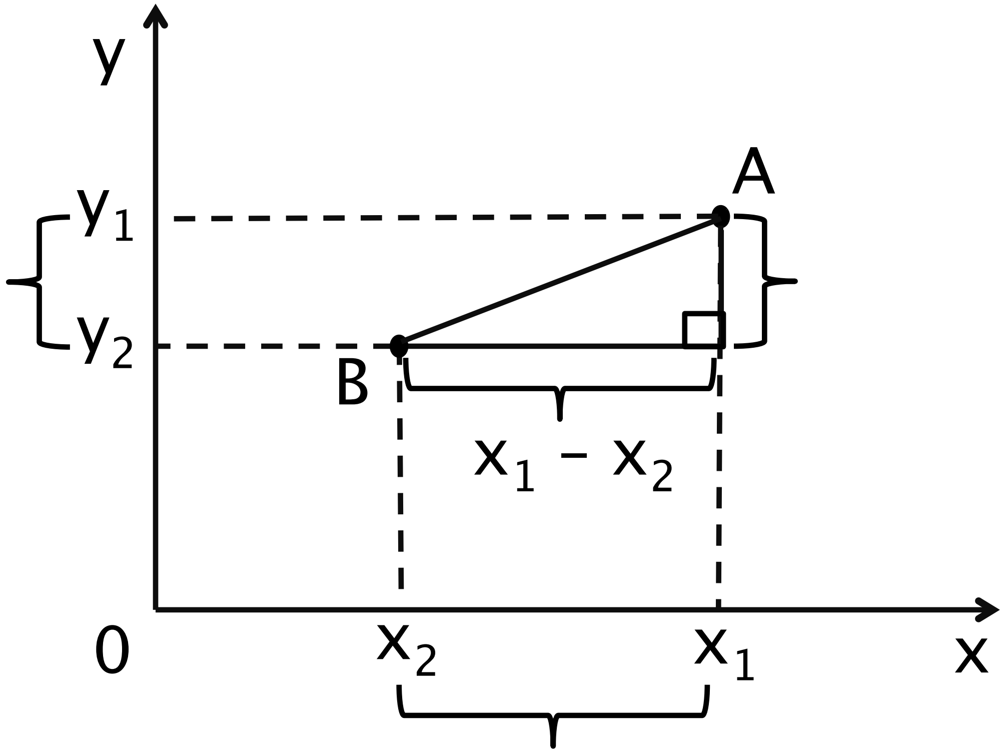
Computing Euclidean distance



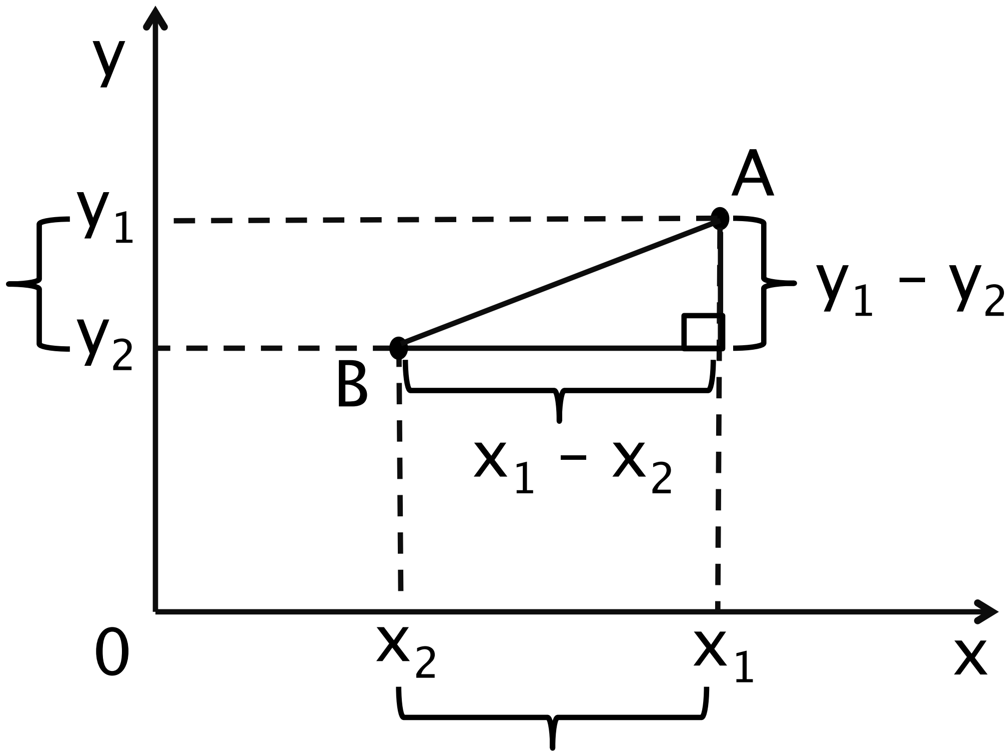
Computing Euclidean distance



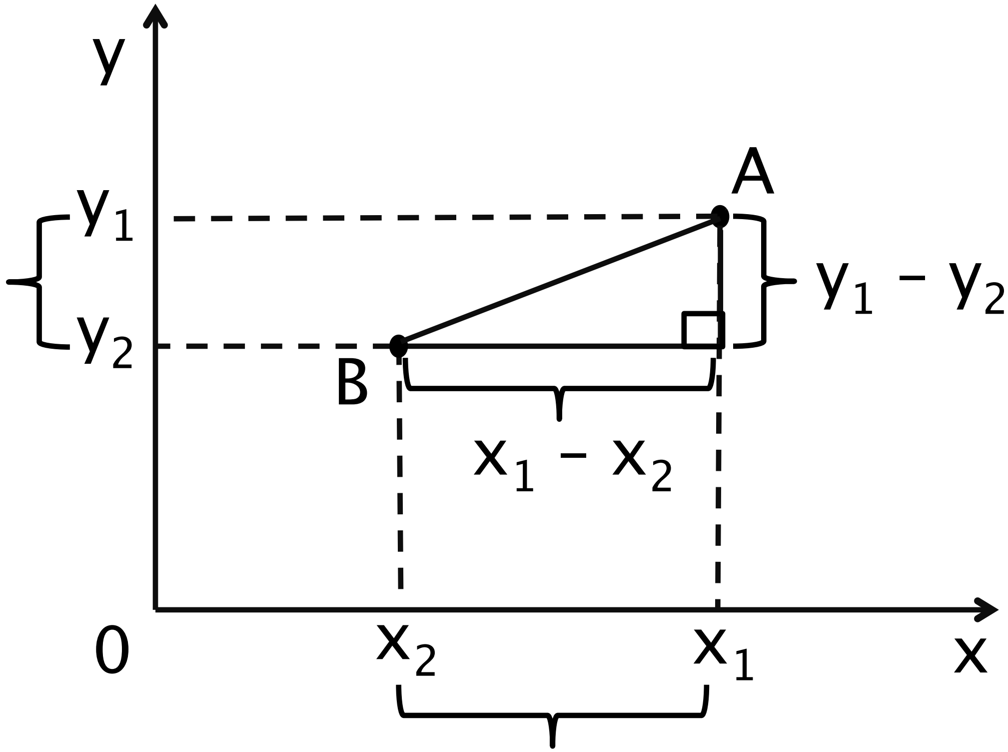
Computing Euclidean distance



Computing Euclidean distance

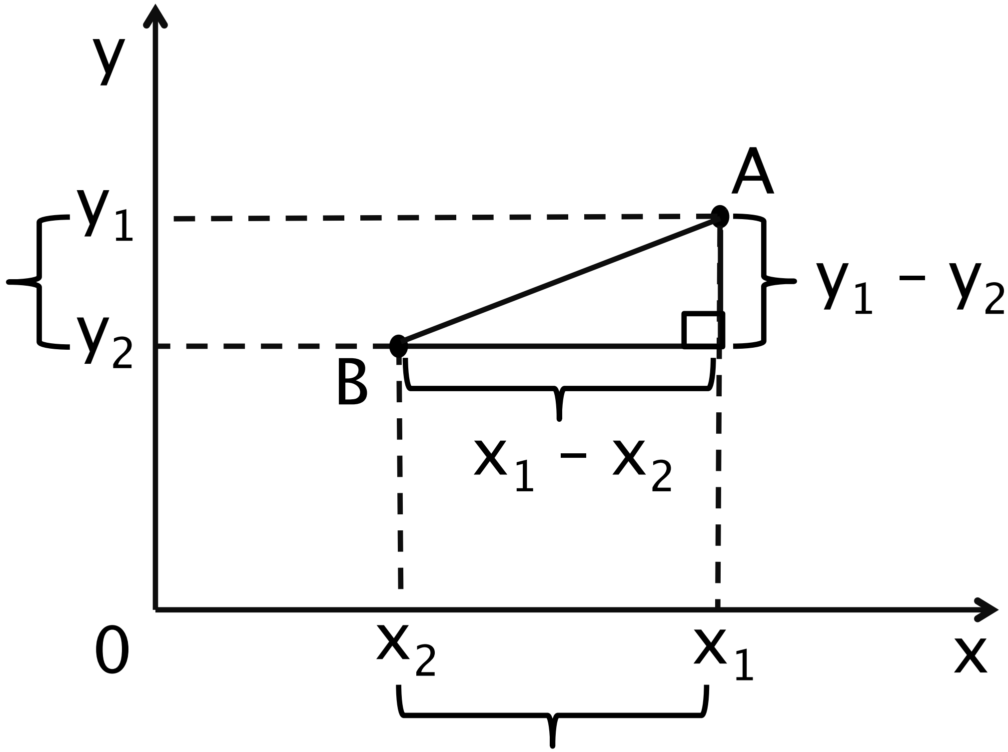


Computing Euclidean distance



- using Pythagorean theorem:

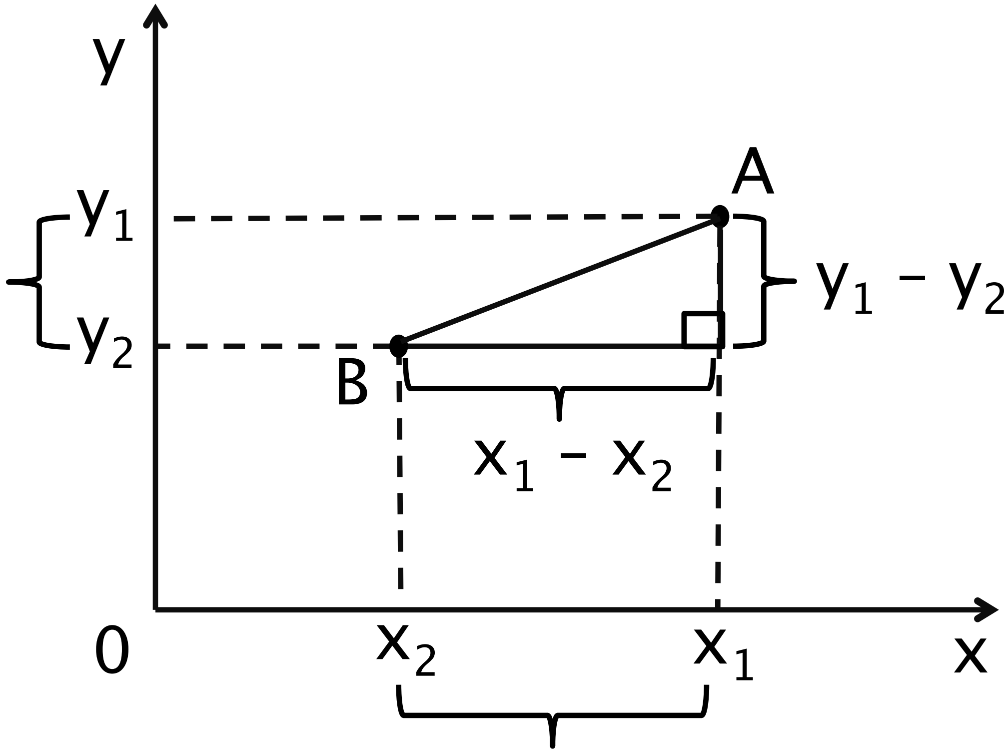
Computing Euclidean distance



- using Pythagorean theorem:

$$d(A,B)^2 = (x_1 - x_2)^2 + (y_1 - y_2)^2$$

Computing Euclidean distance



- using Pythagorean theorem:

$$d(A,B)^2 = (x_1 - x_2)^2 + (y_1 - y_2)^2$$

$$d(A,B) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD

Euclidian distance

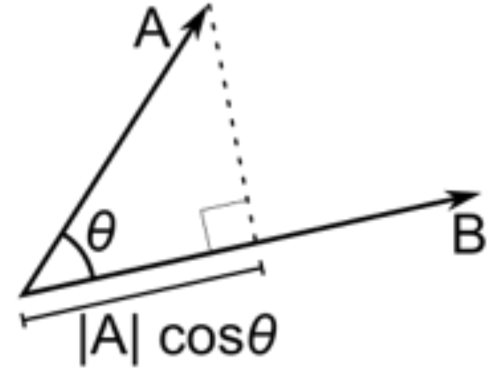
- Euclidian distance in an n-dimensional space between $X = (x_1, x_2, \ldots, x_n)$ and $Y = (y_1, y_2, \ldots, y_n)$:

$$d(X,Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

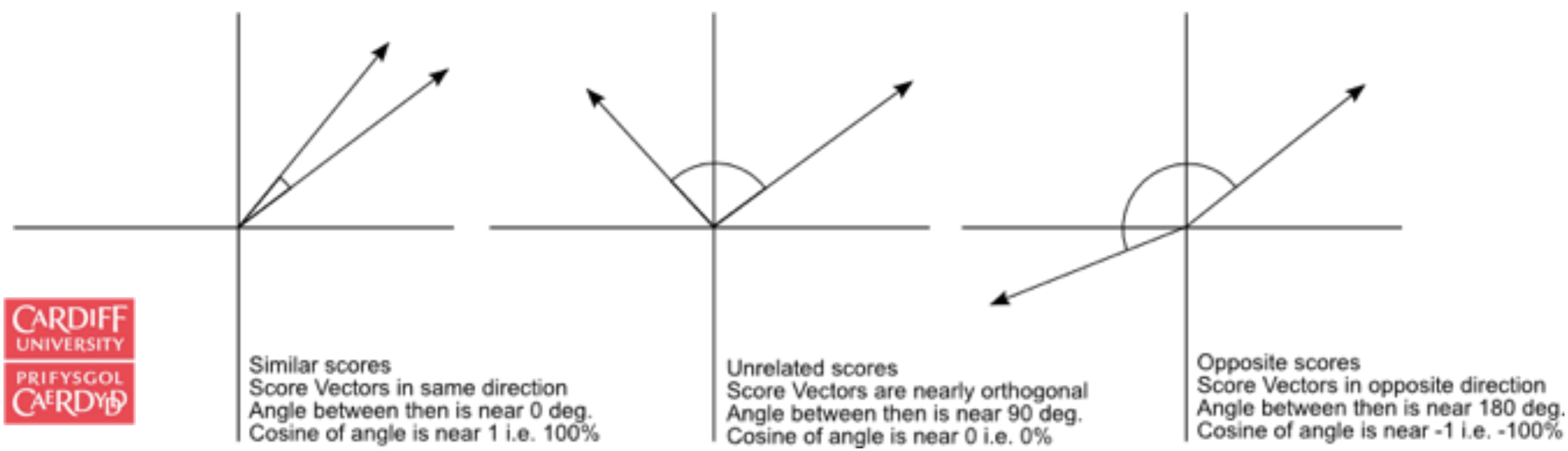
1. square the distance in each dimension
2. sum the squares
3. take the square root

Cosine similarity

Cosine similarity



- the cosine similarity between two vectors in a Euclidean space is a measure that calculates the cosine of the angle between them
- this metric is a measurement of orientation and **not** magnitude



CARDIFF UNIVERSITY PRIFYSGOL CAERDYDD

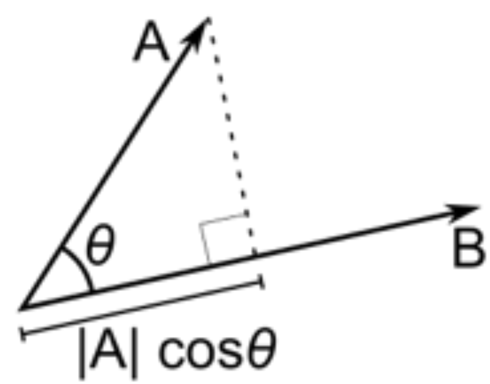
Cosine similarity

- we can calculate the cosine similarity as follows:

$$sim(A,B) = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^{n} a_i \cdot b_i}{\sqrt{\sum_{i=1}^{n} a_i^2} \sqrt{\sum_{i=1}^{n} b_i^2}}$$

- cosine similarity ranges from –1 to 1

Value	Meaning
–1	exactly opposite
1	exactly the same
0	orthogonal
in between	intermediate similarity

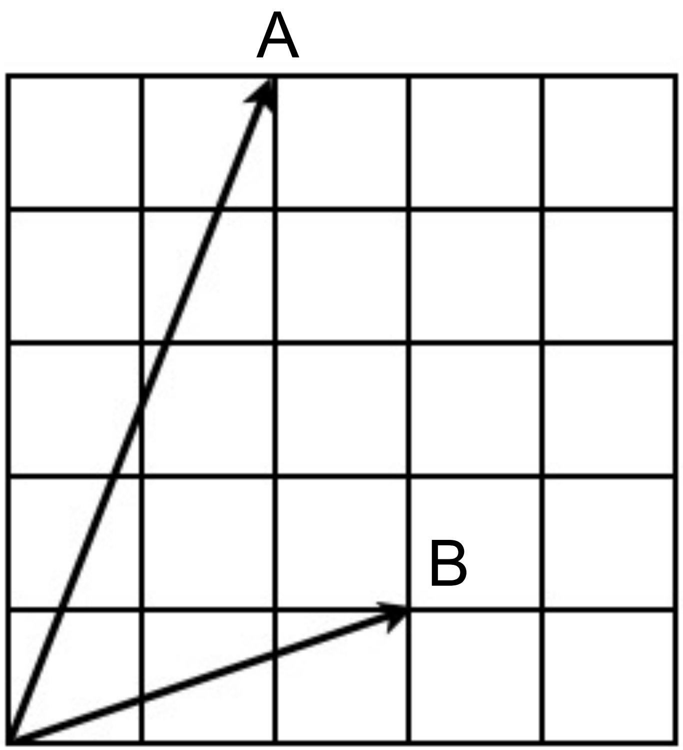


CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD

Cosine similarity

- we can calculate the cosine similarity as follows:

$$sim(A,B) = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^{n} a_i \cdot b_i}{\sqrt{\sum_{i=1}^{n} a_i^2} \sqrt{\sum_{i=1}^{n} b_i^2}}$$



$$\frac{2 \cdot 3 + 5 \cdot 1}{\sqrt{2^2 + 5^2}\sqrt{3^2 + 1^2}} = 0.646$$

Cosine distance

- cosine distance is a term often used for the measure defined by the following formula:

$$d(A,B) = 1 - sim(A,B)$$

- it is important to note that this is **not** a proper distance measure!
  - it does not satisfy the triangle inequality property
  - it violates the coincidence property

Edit distance

Edit distance

- edit distance has been widely applied in natural language processing for approximate string matching, where:

1. the distance between identical strings is equal to zero
2. the distance increases as the string s get more dissimilar with respect to:
   - the symbols they contain, and
   - the order in which they appear

CARDIFF UNIVERSITY PRIFYSGOL CAERDYDD

Edit distance

- informally, edit distance is defined as the minimal number (or cost) of changes needed to transform one string into the other

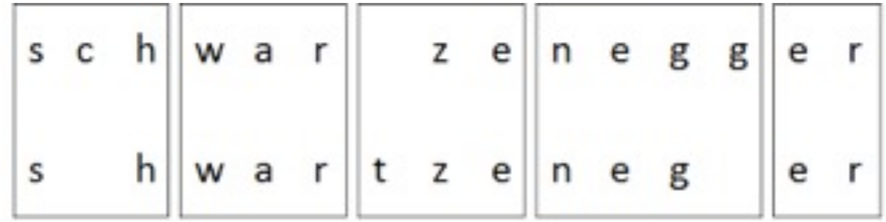
- these changes may include the following edit operations:

1. insertion of a single character
2. deletion of a single character
3. replacement (substitution) of two corresponding characters in the two strings being compared
4. transposition (reversal or swap) of two adjacent characters in one of the strings

Edit operations

▪ insertion	... ac ... ... a**b**c ...
▪ deletion	... a**b**c ... ... ac ...
▪ replacement	... a**b**c ... ... a**d**c ...
▪ transposition	... a**bc** ... ... a**cb** ...

Applications

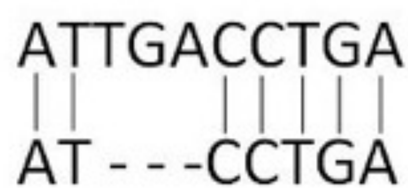


- successfully utilised in NLP applications to deal with:
  - alternate spellings
  - misspellings
  - the use of white spaces as means of formatting
  - UPPER- and lower-case letters
  - other orthographic variations
- e.g. 80% of the spelling mistakes can be identified and corrected automatically by considering a single omission, insertion, substitution or reversal

CARDIFF UNIVERSITY PRIFYSGOL CAERDYDD

Applications

- apart from NLP, the most popular application area of edit distance is molecular biology
- edit distance is used to compare DNA sequences in order to infer information about:
  - common ancestry
  - functional equivalence
  - possible mutations
  - etc.



Notation and terminology

- let $\mathbf{x} = x_1 \dots x_m$ and $\mathbf{y} = y_1 \dots y_n$ be two strings of lengths m and n respectively, where $0 < m \leq n$
- a sequence of edit operations transforming **x** into **y** is referred to as an alignment (also edit sequence, edit script or edit path)
- the cost of an alignment is calculated by summing the costs of individual edit operations it is comprised of
- when all edit operations have the same costs, then the cost of an alignment is equivalent to the total number of operations in the alignment

Edit distance

- formally, the value of edit distance between **x** and **y**, ed(**x**, **y**), corresponds to the minimal alignment cost over all possible alignments for **x** and **y**
- when all edit operations incur the same cost, edit distance is referred to as **simple** edit distance
  - simple edit distance is a distance measure, i.e. $ed(x, y) \geq 0$, $ed(x, y) = 0$ iff $x = y$, $ed(x, y) = ed(y, x)$, $ed(x, z) \leq ed(x, y) + ed(y, z)$
- **general** edit distance permits different costs for different operations or even symbols
  - the choice of the operation costs influences the "meaning" of the corresponding alignments, and thus they depend on a specific application

Variants

- depending on the types of edit operations allowed, a number of specialised variants of edit distance have been identified
  - Hamming distance allows only replacement
  - longest common subsequence problem allows only insertion and deletion, both at the same cost
  - Levenshtein distance allows insertion, deletion and replacement of individual characters, where individual edit operations may have different costs
  - Damerau distance extends Levenshtein distance by permitting the transposition of two adjacent characters

CARDIFF UNIVERSITY PRIFYSGOL CAERDYDD

Variants

- depending on the types of edit operations allowed, a number of specialised variants of edit distance have been identified
  - Hamming distance allows only replacement
  - longest common subsequence problem allows only insertion and deletion, both at the same cost
  - Levenshtein distance allows insertion, deletion and replacement of individual characters, where individual edit operations may have different costs
  - Damerau distance extends Levenshtein distance by permitting the transposition of two adjacent characters

Levenshtein distance

- well suited for a number of practical applications
- most of the existing algorithms have been developed for the simple Levenshtein distance
- many of them can easily be adapted for:
  - general Levenshtein distance, where different costs are used for different operations
  - Damerau distance, where transposition is an allowed edit operation
- transposition is important in some applications such as text searching, where transpositions are typical typing errors
- note that the transposition could be simulated by using an insertion followed by a deletion, but the total cost would be different in that case!

CARDIFF UNIVERSITY PRIFYSGOL CAERDYDD

Dynamic programming

- a class of algorithms based on the idea of:
  - breaking a problem down into sub-problems so that optimal solutions can be obtained for sub-problems
  - combining sub-solutions to produce an optimal solution to the overall problem
- the same idea is applied incrementally to sub-problems
- by saving and re-using the results obtained for the sub-problems, unnecessary re-computation is avoided for recurring sub-problems, thus facilitating the computation of the overall solution

Wagner–Fischer algorithm

- a dynamic programming approach to the computation of Levenshtein distance, which relies on the following reasoning:
- at each step of an alignment, i.e. after aligning two leading substrings of the two strings, there are only three possibilities:

1. delete the next symbol from the first string (delete)
2. delete the next symbol from the second string (insert)
3. match the next symbol in the first string to the first symbol in the second string (exact match or replace otherwise)

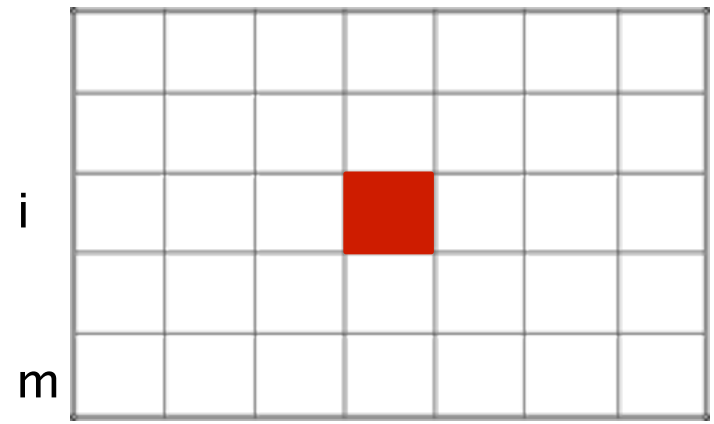
Wagner–Fischer algorithm

- for the cost $C(i, j)$ of aligning the leading substrings $x_1 \ldots x_i$ and $y_1 \ldots y_j$, the cost of their alignment is calculated as follows:
- $1 \le i \le m$, $j = 0$: $C(i, 0) = C(i - 1, 0) + IC(x_i)$
- $i = 0$, $1 \le j \le n$: $C(0, j) = C(0, j - 1) + DC(y_j)$
- $1 \le i \le m$, $1 \le j \le n$: $C(i, j) = \min \begin{cases} C(i-1, j) + IC(x_i) \\ C(i, j-1) + DC(y_j) \\ C(i-1, j-1) + RC(x_i, y_j) \end{cases}$
- where $C(0, 0) = 0$ and IC, DC and RC are the costs of insert, delete and replace operations

CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD

Wagner–Fischer algorithm

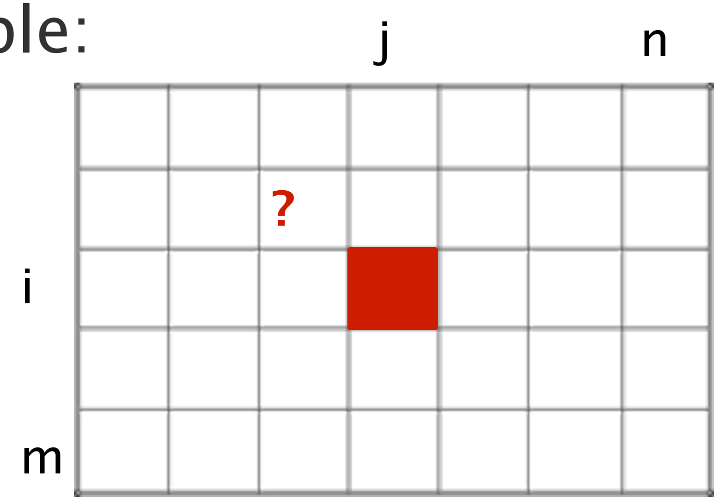
- if the cost values are represented by a cost matrix, then the matrix needs to be filled so that the values needed for the calculation of C(i, j) are available:
  - C(i – 1, j – 1)
  - C(i – 1, j)
  - C(i, j – 1)



- it suffices to fill the cost matrix row-wise left-to-right, column-wise top-to-bottom, or diagonally upper-left to lower-right
- edit distance between **x** and **y** is then obtained as C(m, n)

Wagner–Fischer algorithm

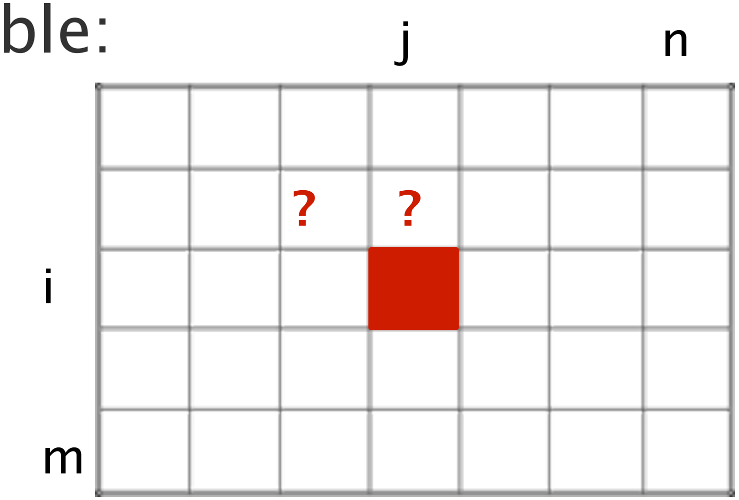
- if the cost values are represented by a cost matrix, then the matrix needs to be filled so that the values needed for the calculation of C(i, j) are available:
  - C(i – 1, j – 1) upper-left
  - C(i – 1, j)
  - C(i, j – 1)



- it suffices to fill the cost matrix row-wise left-to-right, column-wise top-to-bottom, or diagonally upper-left to lower-right
- edit distance between **x** and **y** is then obtained as C(m, n)

Wagner–Fischer algorithm

- if the cost values are represented by a cost matrix, then the matrix needs to be filled so that the values needed for the calculation of C(i, j) are available:
  - C(i – 1, j – 1) upper–left
  - C(i – 1, j) upper
  - C(i, j – 1)

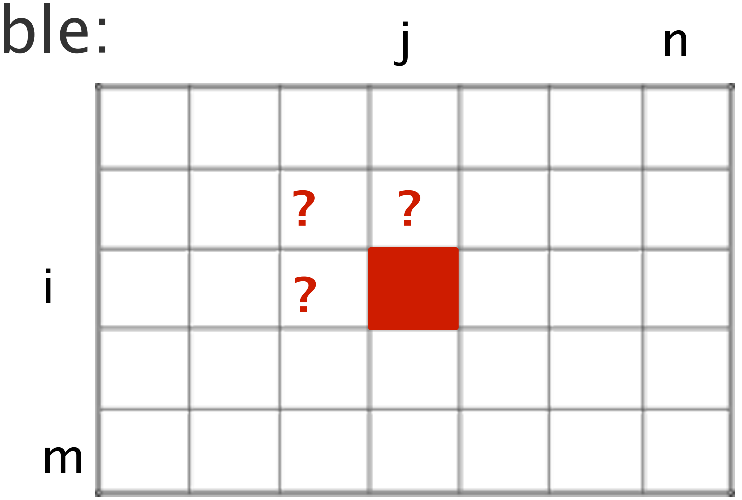


- it suffices to fill the cost matrix row–wise left–to–right, column–wise top–to–bottom, or diagonally upper–left to lower–right
- edit distance between **x** and **y** is then obtained as C(m, n)

CARDIFF UNIVERSITY PRIFYSGOL CAERDYDD

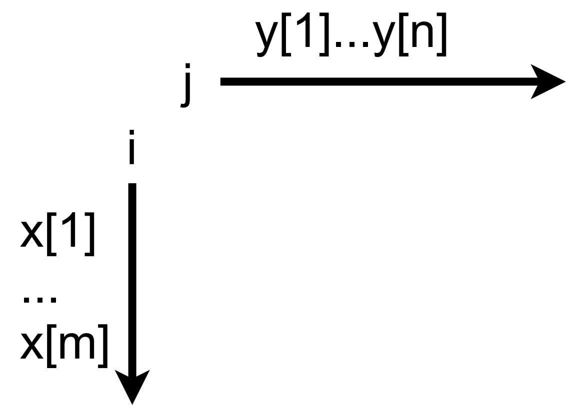
Wagner–Fischer algorithm

- if the cost values are represented by a cost matrix, then the matrix needs to be filled so that the values needed for the calculation of C(i, j) are available:
  - C(i – 1, j – 1) upper-left
  - C(i – 1, j) upper
  - C(i, j – 1) left



- it suffices to fill the cost matrix row-wise left-to-right, column-wise top-to-bottom, or diagonally upper-left to lower-right
- edit distance between **x** and **y** is then obtained as C(m, n)

Wagner-Fischer algorithm



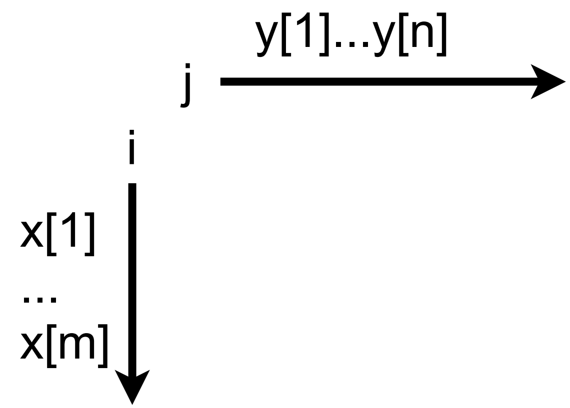

```
let x[1..m], y[1..n] be two arrays of char
let ed[0..m, 0..n] be a 2D array of int

// distance to an empty string
for i in [0..m] ed[i, 0] = i;
for j in [0..n] ed[0, j] = j;

for j in [1..n]
for i in [1..m]
  if x[i] = y[j]   // match, so no operation required
  then ed[i, j] = ed[i-1, j-1];
  else ed[i, j] = minimum of (
    ed[i-1, j] + 1,
    ed[i, j-1] + 1,
    ed[i-1, j-1] + 1
  );

return ed[m,n];
```

Wagner-Fischer algorithm




```
let x[1..m], y[1..n] be two arrays of char
let ed[0..m, 0..n] be a 2D array of int

// distance to an empty string
for i in [0..m] ed[i, 0] = i;
for j in [0..n] ed[0, j] = j;

for j in [1..n]
for i in [1..m]
  if x[i] = y[j]   // match, so no operation required
  then ed[i, j] = ed[i-1, j-1];
  else ed[i, j] = minimum of (
    ed[i-1, j] + 1,
    ed[i, j-1] + 1,
    ed[i-1, j-1] + 1
  );

return ed[m,n];
```

Exercise: compute the distance between "*sweeter*" and "*feathers*"

Extracting an optimal alignment

- start at bottom right, (i,j)=(m,n)
- if x[i]=y[j] then match (x[i],y[j]) & continue at (i-1,j-1)
- else min=minimum(ed[i-1,j], ed[i-1,j-1], ed[i,j-1])
  // ignore cells that don't exist & break ties arbitrarily
- if min=ed[i-1,j] then delete x[i] & continue at (i-1,j)
- if min=ed[i-1,j-1] then replace x[i] by y[j] & continue at (i-1,j-1)
- if min=ed[i,j-1] then insert y[j] & continue at (i,j-1)

(i-1,j-1)	(i-1,j)
(i,j-1)	(i,j)

CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD

String similarity

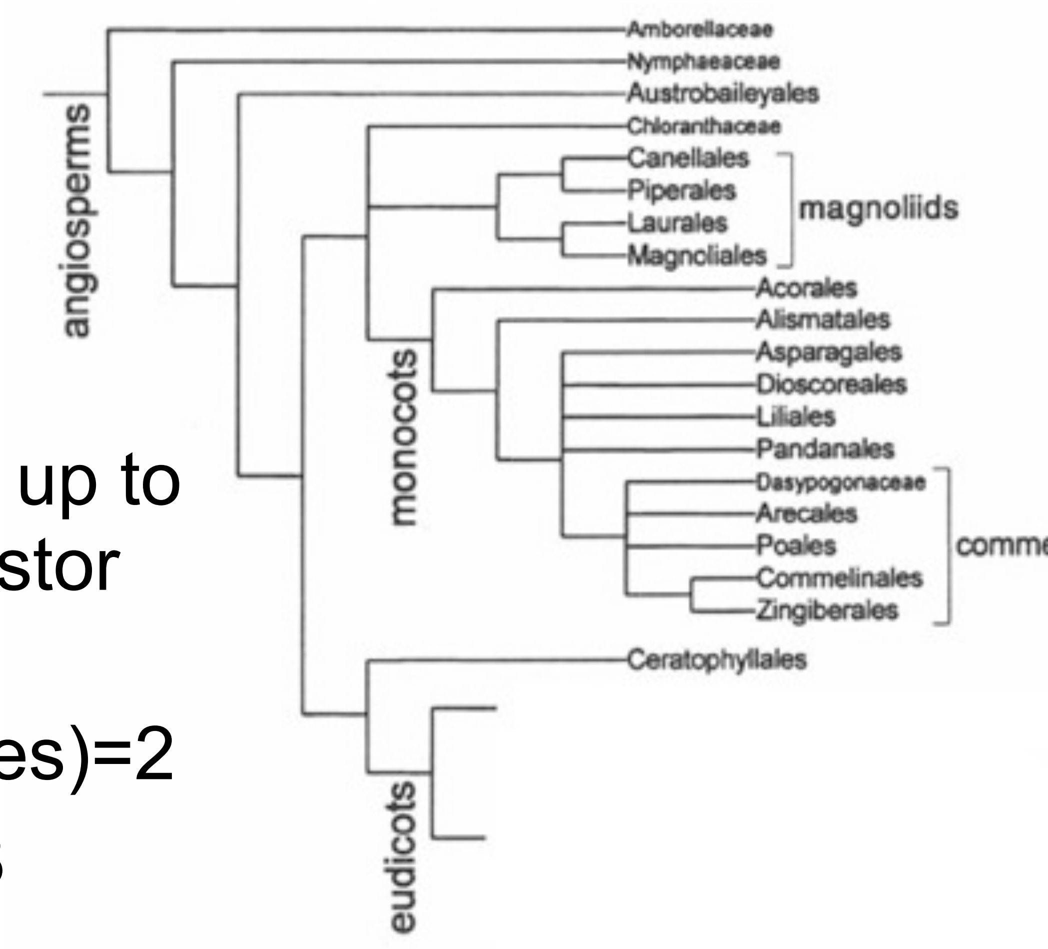
- two strings are regarded similar if their edit distance is lower than a certain threshold: $ed(\mathbf{x}, \mathbf{y}) \leq t \rightarrow \mathbf{x} \sim \mathbf{y}$
- an absolute threshold t does not take into account the lengths m and n ($m \leq n$) of the strings **x** and **y**
- the same threshold should not be used for very long strings and very short ones, e.g.

d o p p e l g ä – n g e r d o p p e l g a e n g e r	vs.	h o t d o g

- a relative threshold $r = t \times m$ proportional to the length of the shorter string is suggested instead
- otherwise, short strings would be erroneously regarded similar to non-similar long strings

Semantic distance

Distance based on Taxonomy



- closeness in tree
- e.g., number of steps up to lowest common ancestor and down again
- d(Laurales,Magnoliales)=2
- d(Laurales,Poales)=8

CARDIFF UNIVERSITY PRIFYSGOL CAERDYDD

Summary

- Similarity plays key role in many applications
- Similarity of sets, points, strings, concepts,...
- Distance vs similarity