

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Introduction

COMP90049 COMP30018 Knowledge Technologies

Jeremy Nicholson and Justin Zobel and Karin Verspoor

Semester 2, 2018



THE UNIVERSITY OF
MELBOURNE

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Week 3:

- Approximate String Search and Matching
- Common Applications
- Methods:
 - Neighbourhood Search
 - Edit Distance
 - N-Gram Distance
 - [Phonetic methods]
- Evaluation
- [Genomics]

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact

Approximate

Application

Methods

Neighbourhood

Edit Distance

N-Gram Distance

Phonetics

Evaluation

References

Genomics

Consider:

- Given a string, is some substring contained within it?
- Given a string (document), find all occurrences of some substring

For example, find Exxon in:

In exes for foxes rex dux mixes a pox of waxed luxes.

An axe, and an axon, to exo Exxon max oxen.

Grexit or Brexit as quixotic haxxers with buxom rex taxation.

Not (really) a Knowledge Technology!

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact

Approximate

Application

Methods

Neighbourhood

Edit Distance

N-Gram Distance

Phonetics

Evaluation

References

Genomics

Find exon in:

In exes for foxes rex dux mixes a pox of waxed luxes.

An axe, and an axon, to exo Exxon max oxen.

Grexit or Brexit as quixotic haxxers with buxom rex taxation.

Not present!

...But what is the “closest” or “best” match?

This is a Knowledge Technology!

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact

Approximate

Applications

Methods

Neighbourhood

Edit Distance

N-Gram Distance

Phonetics

Evaluation

References

Genomics

Two main applications for Approximate String Search:

- Spelling correction
- Computational Genomics

Spelling Correction

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact

Approximate

Application

Methods

Neighbourhood

Edit Distance

N-Gram Distance

Phonetics

Evaluation

References

Genomics



Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact

Approximate

Application

Methods

Neighbourhood

Edit Distance

N-Gram Distance

Phonetics

Evaluation

References

Genomics

Need the notion of a **dictionary**:

- Here, a list of entries that are “correct”
- We can break our input into substrings that we wish to match, and compare each of them against the entries in the dictionary
- An item in the input which *doesn't* appear in the dictionary is *misspelled*
- An item in the input which *does* appear in the dictionary might be correctly spelled *or* misspelled (probably slightly beyond the scope of this subject)

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact

Approximate

Application

Methods

Neighbourhood

Edit Distance

N-Gram Distance

Phonetics

Evaluation

References

Genomics

Therefore, the problem here:

Given some item of interest — which does not appear in our dictionary
— which entry from the dictionary was truly intended?

Depends on the person who wrote the original string!

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact

Approximate

Application

Methods

Neighbourhood

Edit Distance

N-Gram Distance

Phonetics

Evaluation

References

Genomics

- Computational Genomics (later, if we have time)
- Name matching
- Query repair
- Phonetic matching (later, if we have time)
- Data cleaning
- ...

What's a “best” match?

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Find approximate match(es) for exon in:

In exes for foxes rex dux mixes a pox of waxed luxes.

An axe, and an axon, to exo Exxon max oxen.

Grexit or Brexit as quixotic haxxers with buxom rex taxation.

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

For a given string w of interest:

- Generate all variants of w that utilise at most k changes (Insertions/Deletions/Replacements) — **neighbours**
- Check whether generated variants exist in dictionary
- **All** results found in dictionary are returned

Unix command-line utility `agrep` is an efficient mechanism for finding these.

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

For example:

... proceed if you can see no **ther** option ...

Intended word: other

Requires 1 insertion (o) so intended word will be found using neighbourhood search (and some unintended words...)

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact

Approximate

Application

Methods

Neighbourhood

Edit Distance

N-Gram Distance

Phonetics

Evaluation

References

Genomics

Neighbourhood search is suprisingly fast!

Consider: alphabet size is Σ , length of string is $|w|$:

For k edits, roughly $\mathcal{O}(\Sigma^k \cdot |w|^k)$ neighbours

...But Σ is a small constant, string of interest is usually short, and k is usually small

For each neighbour, need a dictionary read (dict has D entries):
Binary search yields $\mathcal{O}(|w|^k \log D)$ string comparisons

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

So, efficiency isn't our problem.

(agrep example)

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Alternative methods:

Scan through each dictionary entry looking for the “best” match

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Global Edit Distance:

Transform the string of interest into each dictionary entry, using the operations Insert, Delete, Replace, and Match (character)

Each operation is associated with a score;
Best match is the dictionary entry with best aggregate **score**

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

For example:

Item of interest: `crat`

Dictionary: `cart`, `arts`

Score: Match +1, Insert -1, Delete -1, Replace -1

`crat` → `cart`:

Match `c` (+1), Delete `r` (-1), Match `a` (+1), Insert `r` (-1), Match `t` (+1) = +1

`crat` → `arts`:

Replace `c` with `a` (-1), Match `r` (+1), Delete `a` (-1), Match `t` (+1), Insert `s` (-1) = -1

`cart` is the better match

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Confusingly, Global Edit Distance isn't a "distance"

...But depends on parameter

Match (0), Insert (+1), Delete (+1), Replace (+1)

This is the Levenshtein Distance (which is a "distance"): it counts the number of edits required to transform one string into the other

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Hypothetically, any parameter is possible!

But some choices make no sense, e.g.:

Match (+4), Insert (-2), Delete (+8), Replace (0)

aba: Which corresponds to best match?

- foo: Insert, Delete, Insert, Delete, Insert, Delete = +18
- aba: Match, Match, Match = +12
- cbc: Replace, Match, Replace = +4

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Often, “direction” doesn’t matter: Insert = Delete (“Indel”)

Sometimes, score of Replace depends on which character is being replaced:

Consider:

Is *faxing* more likely to be *facing* or *faking*?

Global Edit Distance Algorithm

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact

Approximate

Application

Methods

Neighbourhood

Edit Distance

N-Gram Distance

Phonetics

Evaluation

References

Genomics

From string f to string t , given array A of $|f| + 1$ columns and $|t| + 1$ rows, we can solve using the Needleman–Wunsch algorithm:

```
lf = strlen(f); lt = strlen(t);
A[0][0]=0;
for (j=1; j<=lt; j++) A[j][0] = j * i;
for (k=1; k<=lf; k++) A[0][k] = k * d;

for (j=1; j<=lt; j++)
    for (k=1; k<=lf; k++)
        A[j][k] = max3( //Or min3 if m<i,d,r
            A[j][k-1] + d, //Deletion
            A[j-1][k] + i, //Insertion
            A[j-1][k-1] + equal(f[k-1],t[j-1])); //Replace or match
```

`equal()` returns m if characters match, r otherwise

Final score is at $A[|t|][|f|]$

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

In action: from crat to arts, Match (+1), Insert/Delete/Replace (-1)

	ϵ	c	r	a	t
ϵ	0	-1	-2	-3	-4
a	-1	-1	-2	-1	-2
r	-2	-2	0	-1	-2
t	-3	-3	-1	-1	0
s	-4	-4	-2	-2	-1

Global Edit Distance: -1 (Replace, Match, Delete, Match, Insert)

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Algorithm actually depends on parameter!

```
A[j][k] = max3(  
    A[j][k-1] + d, //Deletion  
    A[j-1][k] + i, //Insertion  
    A[j-1][k-1] + equal(f[k-1],t[j-1])); //Replace or match
```

→ Match score greater than Insert/Delete/Replace

e.g. Match (+1), Insert/Delete/Replace (-1)

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact

Approximate

Application

Methods

Neighbourhood

Edit Distance

N-Gram Distance

Phonetics

Evaluation

References

Genomics

Algorithm actually depends on parameter!

```
A[j][k] = min3(  
    A[j][k-1] + d, //Deletion  
    A[j-1][k] + i, //Insertion  
    A[j-1][k-1] + equal(f[k-1],t[j-1])); //Replace or match
```

→ Match score less than Insert/Delete/Replace

e.g. Match (0), Insert/Delete/Replace (+1)

(Levenshtein Distance)

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Local Edit Distance is like Global Edit Distance, but we are searching for the best substring match

Particularly suitable when comparing two strings of very different lengths, e.g. a word and a sentence

Local Edit Distance Algorithm

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact

Approximate

Application

Methods

Neighbourhood

Edit Distance

N-Gram Distance

Phonetics

Evaluation

References

Genomics

From string f to string t , given array A of $|f| + 1$ columns and $|t| + 1$ rows, we can solve using the Smith–Waterman algorithm:

```
lf = strlen(f); lt = strlen(t);
A[0][0]=0;
for (j=1; j<=lt; j++) A[j][0] = 0;
for (k=1; k<=lf; k++) A[0][k] = 0;

for (j=1; j<=lt; j++)
    for (k=1; k<=lf; k++)
        A[j][k] = max4( //0r min4 if m<i,d,r
            0,
            A[j][k-1] + d, //Deletion
            A[j-1][k] + i, //Insertion
            A[j-1][k-1] + equal(f[k-1],t[j-1])); //Replace or match
```

`equal()` returns m if characters match, r otherwise

Final score is greatest value in the entire table (or least value, if $m < i, d, r$)

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

In action: from cart to arts, Match (+1), Insert/Delete/Replace (-1)

	ϵ	c	a	r	t
ϵ	0	0	0	0	0
a	0	0	1	0	0
r	0	0	0	2	1
t	0	0	0	1	3
s	0	0	0	0	2

Best match: art with art (+3); ties are possible.

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

For strings f and t , Both algorithms above are $\mathcal{O}(|f||t|)$ in both space and time. (Space can be improved, but time (probably) cannot.)

When approximate matching, we have a constant string f which we want to compare to each string t in the dictionary D :

$$\mathcal{O}(|f| \sum_{t \in D} |t|)$$

Hence, integer comparisons are roughly the number of characters in the dictionary. Whether this is feasible depends on the size of the dictionary.

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance

N-Gram Distance

Phonetics

Evaluation

References

Genomics

N-Gram Distance has same goal as Edit Distance: compare two strings to determine “best” match

(character) n -gram: substring of length n

2-grams of crat: #c, cr, ra, at, t#

2-grams of cart: #c, ca, ar, rt, t#

2-grams of arts: #a, ar, rt, ts, s#

N-Gram Distance between n -grams of string s ($G_n(s)$) and t ($G_n(t)$):
 $|G_n(s)| + |G_n(t)| - 2 \times |G_n(s) \cap G_n(t)|$

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance

N-Gram Distance

Phonetics

Evaluation

References

Genomics

n -gram: substring of length n

2-grams of crat: #c, cr, ra, at, t#

2-grams of cart: #c, ca, ar, rt, t#

2-grams of arts: #a, ar, rt, ts, s#

2-Gram Distance between crat and cart:

$$|G_2(\text{crat})| + |G_2(\text{cart})| - 2 \times |G_2(\text{crat}) \cap G_2(\text{cart})| \\ = 5 + 5 - 2 \times 2 = 6 \text{ (better)}$$

2-Gram Distance between crat and arts:

$$|G_2(\text{crat})| + |G_2(\text{arts})| - 2 \times |G_2(\text{crat}) \cap G_2(\text{arts})| \\ = 5 + 5 - 2 \times 0 = 10$$

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance

N-Gram Distance

Phonetics

Evaluation

References

Genomics

Occasionally useful as a simpler variant of Edit Distance

More sensitive to long substring matches, less sensitive to relative ordering of strings (matches can be anywhere!)

Despite its simplicity, takes roughly the same time to compare entire dictionary

Quite useless for very long strings and/or very small alphabets (Why?)

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

In English (and some other languages), **orthography** (spelling) isn't a good predictor of **phonetics** (sounds)

Salient concern in speech-to-text systems, e.g.:

Georgia Conal

George O'Connell

Also relevant in spelling correction (English can be very difficult to spell correctly!)

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

One mechanism: Soundex

Translation table:	aehiouwy	→	0 (vowels)
	bpfv	→	1 (labials)
	cgjkqszx	→	2 (misc: fricatives, velars, etc.)
	dt	→	3 (dentals)
	l	→	4 (lateral)
	mn	→	5 (nasals)
	r	→	6 (rhotic)

Four step process:

- 1 Except for initial character, translate string characters according to table
- 2 Remove duplicates (e.g. 4444 → 4)
- 3 Remove 0s
- 4 Truncate to four symbols

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

One mechanism: Soundex

Translation table:	aehiouwy	→	0 (vowels)
	bpfv	→	1 (labials)
	cgjkqsxz	→	2 (misc: fricatives, velars, etc.)
	dt	→	3 (dentals)
	l	→	4 (lateral)
	mn	→	5 (nasals)
	r	→	6 (rhotic)

Four step process:

king	kyngge
k052	k05220
k052	k0520
k52	k52
k52	k52

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Better phonetic methods make use of the fact that some letters sounds alike in certain contexts, and different in other contexts

Editex uses the Edit Distance to compare strings based on a similar translation table to Soundex

Ipadist uses a text-to-sound algorithm to represent tokens according to the International Phonetic Alphabet (but context matters a lot)

There are also worse variants, like Phonix.

Evaluating an Approximate Matching System

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Evaluation: consider whether the system is effective at solving the user's problem

In this case: for a misspelled word, does the system identify the correct word?

To evaluate, we need:

- A number of cases of misspelled words
- The intended (correct) word for each case
- An **evaluation metric**

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

We have some cases:

Misspelled Word	Correct Word	Predicted Word	Right/Wrong?
ther	other	there	×
corridr	corridor	corridor	✓
cracheyt	crotchety	cachet	×
...

Accuracy: fraction of correct responses ($\frac{1}{3}$)

Evaluation Metrics for Spelling Correction

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

More realistic situation:

Misspelled Word	Correct Word	Predicted Word	Right/Wrong?
ther	other	there	×
		other	✓
		their	×
corridr	corridor	corridor	✓
		carrier	×
cracheyt	crotchety	???	—
...	

Precision: fraction of correct responses among attempted responses
($\frac{2}{5}$)

Recall: proportion of words with a correct response (somewhere) ($\frac{2}{3}$)

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Typically, the value of the evaluation metric has little intrinsic meaning

“This system gets 81% accuracy” — useful for users, or not?

“The system based on the Global Edit Distance gets 81% accuracy, whereas the system based on the N-Gram Distance gets 84% accuracy”

“The basic system gets 81% accuracy, but after making some changes, the accuracy becomes 74%”

“System A gets 45% precision and 80% recall;
System B gets 95% precision and 10% recall”
— Which one should we use? (Also: why?)

The answer depends on the problem (and the user)!

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

- What is approximate string search?
- What are some common applications of approximate string search; why are they hard?
- What are some methods for finding an approximate match to a string? What do we need to generate them?
- How can we evaluate a typical approximate matching system?

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Needleman, Saul B. and Wunsch, Christian D. (1970). "A general method applicable to the search for similarities in the amino acid sequence of two proteins". *Journal of Molecular Biology* 48 (3): 44353. doi:10.1016/0022-2836(70)90057-4

(Originally in Russian, published in English as:) Levenshtein, Vladimir I. (1966). "Binary codes capable of correcting deletions, insertions, and reversals". *Soviet Physics Doklady* 10 (8): 707710.

Smith, Temple F. and Waterman, Michael S. (1981). "Identification of Common Molecular Subsequences". *Journal of Molecular Biology* 147: 195197. doi:10.1016/0022-2836(81)90087-5

Kondrak, Grzegorz (2005). "N-Gram Similarity and Distance". In *Proceedings of the 12th international conference on String Processing and Information Retrieval (SPIRE'05)*, pp. 115-126, Buenos Aires, Argentina.

Zobel, Justin and Dart, Philip (1996). "Phonetic String Matching: Lessons from Information Retrieval". In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR'96)*, pp. 166-172, New York, USA.

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Whitelaw, Casey and Hutchison, Ben and Chung, Grace Y and Ellis, Gerard (2009). "Using the Web for Language Independent Spellchecking and Autocorrection". In Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP 2009), pp. 890-899, Singapore, Singapore.

Ahmad, Farooq and Kondrak, Grzegorz (2005). "Learning a Spelling Error Model from Search Query Logs". In Proceedings of the Human Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005), pp. 955-962, Vancouver, Canada.

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Typical Genomics problem:

- Given a nucleotide/amino acid sequence (substring)
- Find whether the sequence occurs within a larger sequence (string)
- Possibly with “errors” (nucleotide/amino acid changes)

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Typical Genomics problem:

- Given a substring, find whether the sequence occurs within a larger string, possibly with “errors”
- Almost the same as spelling correction
- But **much** larger strings: a small genomics problem might involve comparing perhaps 1K character sequence against several 100K character sequences; alphabet is smaller

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Recall: we have a “short” ($\sim 1\text{K}$ character) nucleotide/amino acid sequence to compare against many long ($\sim 100\text{K}$ character) chromosomes/genes/proteins/etc.

For example, if some member of the population has 99% of the sequence of interest, they might be susceptible to some medical condition

We're allowed ~ 10 errors; alphabet is ~ 4 or ~ 20 characters

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Neighbourhood search:

Roughly $4^{10} \times 1000^{10}$ possible neighbours.

... Forget it.

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Global Edit Distance:

One string is $\sim 1\text{K}$ characters, other is $\sim 100\text{K}$ characters.

... Every string comparison involves $\sim 99\text{K}$ insertions.

→ Prefers shorter chromosomes (not intended behaviour)

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Local Edit Distance:

One string is $\sim 1\text{K}$ characters, other is $\sim 100\text{K}$ characters.

... Seems like the right idea.

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

Local Edit Distance:

One string is $\sim 10K$ characters, other is $\sim 1G$ characters.

... Can't fit table into memory.

... Requires approximate solutions with heuristics, e.g. BLAST, FASTA

Introduction

COMP90049
COMP30018
Knowledge
Technologies

String Search

Exact
Approximate
Application

Methods

Neighbourhood
Edit Distance
N-Gram Distance

Phonetics

Evaluation

References

Genomics

N-Gram Distance:

With huge n (e.g. 80% of length of shorter string) can (almost) work!

Tends to prefer shorter chromosomes like Global Edit Distance

But better methods for using n -gram information, e.g. de Bruijn graphs