Edit Distance with Single-Symbol Combinations and Splits

Manolis Christodoulakis[1] and Gerhard Brey[2]

[1]School of Computing & Technology,
University of East London
m.christodoulakis@uel.ac.uk

[2]Centre for Computing in the Humanities,
Kings College London
gerhard.brey@kcl.ac.uk

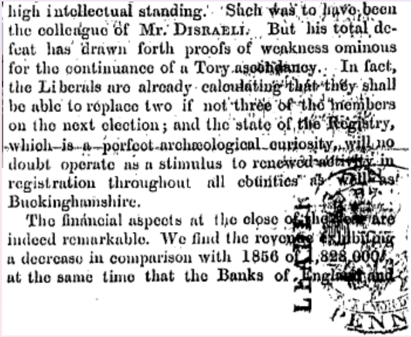
Prague Stringology Conference 2008

NCSE – Our Data Repository

- **Nineteenth-Century Serials Edition (NCSE)**
  `http://www.ncse.ac.uk`
- Collaborative project
  - Birkbeck College, London
  - British Library
  - King's College London (Centre for Computing in the Humanities)
  - Olive Software
- Digital edition of 6 newspapers / periodicals published between 1806–1890
- Microfilmed versions of the newspapers were scanned and OCR'd by Olive Software

OCR Quality

- OCR quality varies widely across corpus



high intellectual standing. Such was to have been the colleague of Mr. DISRAELI. But his total defeat has drawn forth proofs of weakness ominous for the continuance of a Tory ascendancy. In fact, the Liberals are already calculating that they shall be able to replace two if not three of the members on the next election; and the state of the Registry, which is a perfect archæological curiosity, will no doubt operate as a stimulus to renewed activity in registration throughout all counties as well as Buckinghamshire.

The financial aspects at the close of the year are indeed remarkable. We find the revenue exhibiting a decrease in comparison with 1856 of 1,828,000l., at the same time that the Banks of England and

high intollectual standing.' 'Such ^as '' ,to jja^been
the colleague 'of Mr.' Disraeli. But his tp^al,
defeat has' drawn forth proofs of wcAkness ominous
for tho continuance of a Tory.as,o^jlanay.: In fact,
the Liberals are alroady ealoidaWj! *^£fr'tnfc y; shall
bo able to replace two if notHhreje ofi-hd ^e '^oi-s
on tho next election; and the state o^ttftfyiiglfriry,
».wluoU-i&;.~a<-~porfoot,w
doubt oporato as a stimulus to rcnewc\1^tt#fcj1aK : _
registration throughout all counties"' aV'mUEfea?
Buckinghamshire. ^^f^'mwfc)
The financial aspects at the clpso oJ^1 ' i^|y|g'W ,
indeed remarkable. We find the roYOXJMjjgi ffflbflGngj1 '
u decrease in comparison with 1856 °S» ^^ rp|0^3
at tho same time that tho Banks of ^ngl^^n^

The Heart of the Problem

Approximate Pattern Matching

How to distinguish good approximate matches from random ones?

The Heart of the Problem

Approximate Pattern Matching

How to distinguish good approximate matches from random ones?

Example

Let the pattern be `Billington`.

The Heart of the Problem

Approximate Pattern Matching

How to distinguish good approximate matches from random ones?

Example

Let the pattern be `Billington`.

Matches	Edit Distance	True match?
`Billmgton`	2	Must be ✓

The Heart of the Problem

Approximate Pattern Matching

How to distinguish good approximate matches from random ones?

Example

Let the pattern be `Billington`.

Matches	Edit Distance	True match?
`Billmgton`	2	Must be ✓
`Wellington`	2	Not really ✗

The Heart of the Problem

Approximate Pattern Matching

How to distinguish good approximate matches from random ones?

Example

Let the pattern be `Billington`.

Matches	Edit Distance	True match?
`Billmgton`	2	Must be ✓
`Wellington`	2	Not really ✗

Remarks

- `m` optically resembles `in`
- `m` is essentially a merge of `i` and `n`

Problem Requirements

Ideas

An enhanced edit-distance algorithm that allows:

Problem Requirements

Ideas

An enhanced edit-distance algorithm that allows:

- symbols that look similar to get smaller penalty

Problem Requirements

Ideas

An enhanced edit-distance algorithm that allows:

- symbols that look similar to get smaller penalty
- groups of symbols to be combined and matched against a symbol that looks similar to the combination

Problem Requirements

Ideas

An enhanced edit-distance algorithm that allows:

- symbols that look similar to get smaller penalty
- groups of symbols to be combined and matched against a symbol that looks similar to the combination

We introduce a new edit-distance operation:

Definition

The single-symbol combination (hereafter: a combination) operation allows a string, e.g. "in" to be matched against a single symbol, e.g. "m". The dual of a combination is the split operation.

Combination Operation in DP

		0	1	2	3	4	5	6	7	8	9
			B	i	l	l	m	g	t	o	n
0		0	10	20	30	40	50	60	70	80	90
1	B	10	0	10	20	30	40	50	60	70	80
2	i	20	10	0	10	20	30	40	50	60	70
3	l	30	20	10	0	10	20	30	40	50	60
4	l	40	30	20	10	0	10	20	30	40	50
5	i	50	40	30	20	10	10	20	30	40	50
6	n	60	50	40	30	20					
7	g										
8	t										
9	o										
10	n										

$d_{sub} = 10, \ d_{indel} = 10, \ d_{comb} = 1$

Combination Operation in DP

		0	1	2	3	4	5	6	7	8	9
			B	i	l	l	m	g	t	o	n
0		0	10	20	30	40	50	60	70	80	90
1	B	10	0	10	20	30	40	50	60	70	80
2	i	20	10	0	10	20	30	40	50	60	70
3	l	30	20	10	0	10	20	30	40	50	60
4	l	40	30	20	10	0	10	20	30	40	50
5	i	50	40	30	20	10	10	20	30	40	50
6	n	60	50	40	30	20					
7	g										
8	t										
9	o										
10	n										

$d_{sub} = 10,\ d_{indel} = 10,\ d_{comb} = 1$

Combination Operation

Find the suffix (if any) of $x[1..i]$ that can be combined into symbol $y[j]$ with the minimum cost.

Combination Operation in DP

		0	1	2	3	4	5	6	7	8	9
			B	i	l	l	m	g	t	o	n
0		0	10	20	30	40	50	60	70	80	90
1	B	10	0	10	20	30	40	50	60	70	80
2	i	20	10	0	10	20	30	40	50	60	70
3	l	30	20	10	0	10	20	30	40	50	60
4	l	40	30	20	10	0	10	20	30	40	50
5	i	50	40	30	20	10	10	20	30	40	50
6	n	60	50	40	30	20					
7	g										
8	t										
9	o										
10	n										

$d_{sub} = 10, \; d_{indel} = 10, \; d_{comb} = 1$

Combination Operation

Find the suffix (if any) of $x[1..i]$ that can be combined into symbol $y[j]$ with the minimum cost.

Combination Operation in DP

		0	1	2	3	4	5	6	7	8	9
			B	i	l	l	m	g	t	o	n
0		0	10	20	30	40	50	60	70	80	90
1	B	10	0	10	20	30	40	50	60	70	80
2	i	20	10	0	10	20	30	40	50	60	70
3	l	30	20	10	0	10	20	30	40	50	60
4	l	40	30	20	10	0	10	20	30	40	50
5	i	50	40	30	20	10	10	20	30	40	50
6	n	60	50	40	30	20					
7	g										
8	t										
9	o										
10	n										

$d_{sub} = 10, \; d_{indel} = 10, \; d_{comb} = 1$

Combination Operation

Find the suffix (if any) of $x[1..i]$ that can be combined into symbol $y[j]$ with the minimum cost.

Problem!

$O(n)$ suffixes per cell!

Combination Operation in DP

		0	1	2	3	4	5	6	7	8	9
			B	i	l	l	m	g	t	o	n
0		0	10	20	30	40	50	60	70	80	90
1	B	10	0	10	20	30	40	50	60	70	80
2	i	20	10	0	10	20	30	40	50	60	70
3	l	30	20	10	0	10	20	30	40	50	60
4	l	40	30	20	10	0	10	20	30	40	50
5	i	50	40	30	20	10	10	20	30	40	50
6	n	60	50	40	30	20					
7	g										
8	t										
9	o										
10	n										

$d_{sub} = 10, \; d_{indel} = 10, \; d_{comb} = 1$

Combination Operation

Find the suffix (if any) of $x[1..i]$ that can be combined into symbol $y[j]$ with the minimum cost.

Problem!

$O(n)$ suffixes per cell!

Idea

Preprocess the valid combinations.

Combination Lists

Assumptions

For every symbol α, a (possibly empty) list is provided, call it C_α (the *combination list* of α), that contains all the strings that resemble α.

Combination Lists

Assumptions

For every symbol α, a (possibly empty) list is provided, call it C_α (the *combination list* of α), that contains all the strings that resemble α.

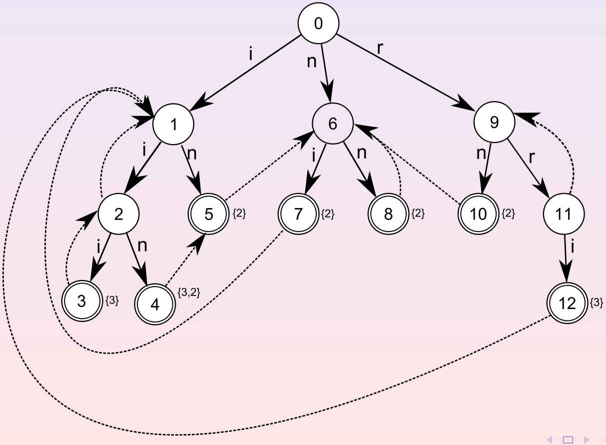
Preprocessing Algorithm

For each symbol α

1. Build Aho-Corasick automaton, T_α, from the list C_α
2. Replace the output values of final nodes with the lengths of the strings matched at those nodes

Combination Lists

T_m built from the combination list $C_m = \{\text{iii,iin,in,ni,nn,rn,rri}\}$



Incorporating the Combination Lists

		0	1	2	3	4	5	6	7	8	9
			B	i	l	l	m	g	t	o	n
0		0	10	20	30	40	50	60	70	80	90
1	B	10	0	10	20	30	40	50	60	70	80
2	i	20	10	0	10	20	30	40	50	60	70
3	l	30	20	10	0	10	20	30	40	50	60
4	l	40	30	20	10	0	10	20	30	40	50
5	i	50	40	30	20	10	10	20	30	40	50
6	n	60	50	40	30	20					
7	g										
8	t										
9	o										
10	n										

$d_{sub} = 10,\ d_{indel} = 10,\ d_{comb} = 1$

Incorporating the Combination Lists

		0	1	2	3	4	5	6	7	8	9
			B	i	l	l	m	g	t	o	n
0		0	10	20	30	40	50	60	70	80	90
1	B	10	0	10	20	30	40	50	60	70	80
2	i	20	10	0	10	20	30	40	50	60	70
3	l	30	20	10	0	10	20	30	40	50	60
4	l	40	30	20	10	0	10	20	30	40	50
5	i	50	40	30	20	10	10	20	30	40	50
6	n	60	50	40	30	20					
7	g										
8	t										
9	o										
10	n										

$d_{sub} = 10,\ d_{indel} = 10,\ d_{comb} = 1$

Remark 1

For cell $D(6,5)$ we need to spell out "Billin" on T_m

Incorporating the Combination Lists

		0	1	2	3	4	5	6	7	8	9
			B	i	l	l	m	g	t	o	n
0		0	10	20	30	40	50	60	70	80	90
1	B	10	0	10	20	30	40	50	60	70	80
2	i	20	10	0	10	20	30	40	50	60	70
3	l	30	20	10	0	10	20	30	40	50	60
4	l	40	30	20	10	0	10	20	30	40	50
5	i	50	40	30	20	10	10	20	30	40	50
6	n	60	50	40	30	20					
7	g										
8	t										
9	o										
10	n										

$d_{sub} = 10, \ d_{indel} = 10, \ d_{comb} = 1$

Remark 1

For cell $D(6,5)$ we need to spell out "Billin" on T_m

Remark 2

For cell $D(7,5)$ we need to spell out "Billing" on T_m (only one letter longer than the previous)

Incorporating the Combination Lists

		0	1	2	3	4	5	6	7	8	9
			B	i	l	l	m	g	t	o	n
0		0	10	20	30	40	50	60	70	80	90
1	B	10	0	10	20	30	40	50	60	70	80
2	i	20	10	0	10	20	30	40	50	60	70
3	l	30	20	10	0	10	20	30	40	50	60
4	l	40	30	20	10	0	10	20	30	40	50
5	i	50	40	30	20	10	10	20	30	40	50
6	n	60	50	40	30	20					
7	g										
8	t										
9	o										
10	n										

$d_{sub} = 10,\ d_{indel} = 10,\ d_{comb} = 1$

Remark 1

For cell $D(6,5)$ we need to spell out "Billin" on T_m

Remark 2

For cell $D(7,5)$ we need to spell out "Billing" on T_m (only one letter longer than the previous)

Idea

Store a pointer to the node of T_m where "Billin" ended. When processing $D(7,5)$ proceed from that node following the letter 'g' only.

Incorporating the Combination Lists

		0	1	2	3	4	5	6	7	8	9
			B	i	l	l	m	g	t	o	n
0		0	10	20	30	40	50	60	70	80	90
1	B	10	0	10	20	30	40	50	60	70	80
2	i	20	10	0	10	20	30	40	50	60	70
3	l	30	20	10	0	10	20	30	40	50	60
4	l	40	30	20	10	0	10	20	30	40	50
5	i	50	40	30	20	10	10	20	30	40	50
6	n	60	50	40	30	20					
7	g										
8	t										
9	o										
10	n										

1	2	3	4	5	6	7	8	9
B	i	l	l	m	g	t	o	n
root TB	root Ti	root Tl	root Tl	root Tm	root Tg	root Tt	root To	root Tn

$d_{sub} = 10,\ d_{indel} = 10,\ d_{comb} = 1$

Incorporating the Combination Lists

		0	1	2	3	4	5	6	7	8	9
			B	i	l	l	m	g	t	o	n
0		0	10	20	30	40	50	60	70	80	90
1	B	10	0	10	20	30	40	50	60	70	80
2	i	20	10	0	10	20	30	40	50	60	70
3	l	30	20	10	0	10	20	30	40	50	60
4	l	40	30	20	10	0	10	20	30	40	50
5	i	50	40	30	20	10	10	20	30	40	50
6	n	60	50	40	30	20					
7	g										
8	t										
9	o										
10	n										

$d_{sub} = 10,\ d_{indel} = 10,\ d_{comb} = 1$

1	2	3	4	5	6	7	8	9
B	i	l	l	m	g	t	o	n
"B" TB	"B" Ti	"B" Tl	"B" Tl	"B" Tm	"B" Tg	"B" Tt	"B" To	"B" Tn

Incorporating the Combination Lists

		0	1	2	3	4	5	6	7	8	9
			B	i	l	l	m	g	t	o	n
0		0	10	20	30	40	50	60	70	80	90
1	B	10	0	10	20	30	40	50	60	70	80
2	i	20	10	0	10	20	30	40	50	60	70
3	l	30	20	10	0	10	20	30	40	50	60
4	l	40	30	20	10	0	10	20	30	40	50
5	i	50	40	30	20	10	10	20	30	40	50
6	n	60	50	40	30	20					
7	g										
8	t										
9	o										
10	n										

1	2	3	4	5	6	7	8	9
B	i	l	l	m	g	t	o	n
"Bi" TB	"Bi" Ti	"Bi" Tl	"Bi" Tl	"Bi" Tm	"Bi" Tg	"Bi" Tt	"Bi" To	"Bi" Tn

$d_{sub} = 10,\ d_{indel} = 10,\ d_{comb} = 1$

Running Time and Space

Preprocessing Time

$$O\Big(\sum_{\forall\alpha} \ell_\alpha\Big) \qquad \text{where } \ell_\alpha = \sum_{x \in C_\alpha} |x|$$

Running Time and Space

Preprocessing Time

$$O\left(\sum_{\forall\alpha} \ell_\alpha\right) \qquad \text{where } \ell_\alpha = \sum_{x \in C_\alpha} |x|$$

Edit Distance Algorithm Time

$O(mnk)$ where k is an upper bound on $|C_\alpha|$ over all α

Running Time and Space

Preprocessing Time

$$O\Big(\sum_{\forall\alpha} \ell_\alpha\Big) \qquad \text{where } \ell_\alpha = \sum_{x \in C_\alpha} |x|$$

Edit Distance Algorithm Time

$O(mnk)$ where k is an upper bound on $|C_\alpha|$ over all α

Space Requirements

$$O\Big(mn + \sum_{\forall\alpha} \ell_\alpha\Big)$$

Open Problems

Current Algorithm

Single-symbol combinations allow a string (e.g. "in") to be matched against a *single* symbol (e.g. "m").

Open Problems

Current Algorithm

Single-symbol combinations allow a string (e.g. “in”) to be matched against a *single* symbol (e.g. “m”).

Definition (Edit Distance with Re-Combinations)

How about allowing a string (e.g. “in”) to be matched against another string (e.g. “ni”)?

Open Problems

Current Algorithm

Single-symbol combinations allow a string (e.g. “in”) to be matched against a *single* symbol (e.g. “m”).

Definition (Edit Distance with Re-Combinations)

How about allowing a string (e.g. “in”) to be matched against another string (e.g. “ni”)?

Definition (Edit Distance with Transitive Combinations)

Let “in” $\in C_m$ and “ri” C_n. Write an algorithm that can infer that “iri” resembles “m” despite the fact that “iri” $\notin C_m$.