Selected Topics of Pervasive Computing

Stephan Sigg

Georg-August-University Goettingen, Computer Networks

29.01.2014

Overview and Structure

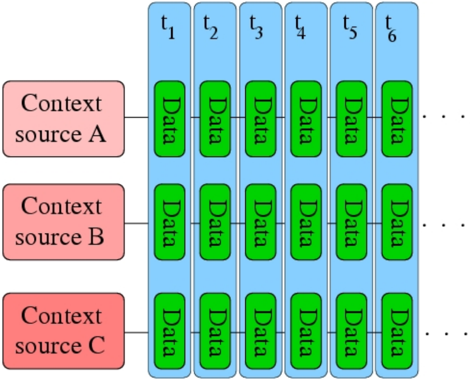
30.10.2013 Organisational
30.10.3013 Introduction
06.11.2013 Classification methods (Feature extraction, Metrics, machine learning)
13.11.2013 Classification methods (Basic recognition, Bayesian, Non-parametric)
20.11.2013 –
27.11.2013 –
04.12.2013 –
11.12.2013 Classification methods (Linear discriminant, Neural networks)
18.12.2013 Classification methods (Sequential, Stochastic)
08.01.2014 Features from the RF channel (Effects of the mobile radio channel)
15.01.2014 Security from noisy data (Encryption schemes, Fuzzy extractors)
22.01.2014 Security from noisy data (Error correcting codes, PUFs, Applications)
29.01.2014 Context prediction (Definitions)
05.02.2014 Context prediction (Algorithms and applications)

Outline

Aspects of prediction algorithms

Multi-dimensional time series

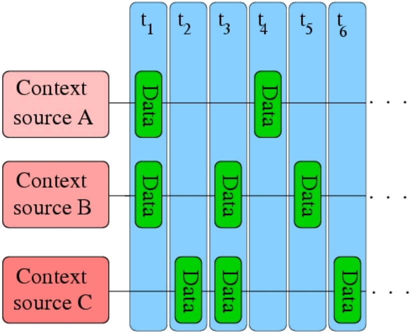
- Idealised: Context data sources synchonised
  - Very unlikely



Aspects of prediction algorithms

Multi-dimensional time series

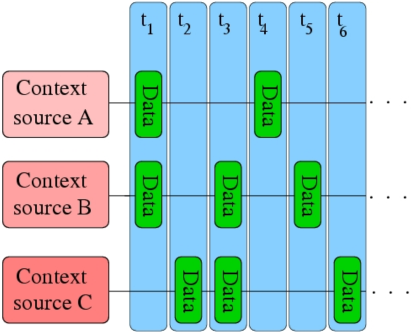
- Realistic sceario: No synchonisation between context sources
  - Context sources push information when specific events occur
  - Duty cycling (time differs between context sources)



Aspects of prediction algorithms

Multi-dimensional time series

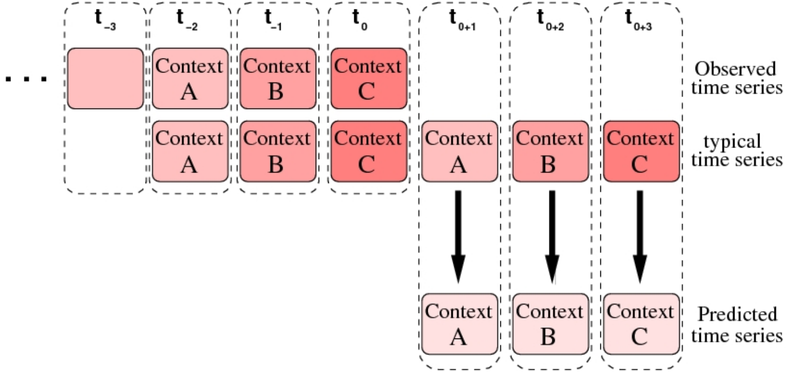
- Question: Which context values for a given time interval?
  - Interpolation of context values?
  - Last value measured?



Aspects of prediction algorithms

Relaxation of typical behaviour patterns

- Exact sequence matching



Aspects of prediction algorithms

Relaxation of typical behaviour patterns

- Exact pattern matching not suited in most ubiquitous scenarios
  - Behaviour patterns do not reoccur 'exactly' but approximately
  - E.g. the route and time to some location will differ slightly for several times the route is taken.
- Approximate matching is more difficult:
  - Where to draw the line?
  - When are two time series considered as approximately matching and when not
  - Inherently dependent on given scenario
  - Typically solved by heuristic approach/metric

Aspects of prediction algorithms

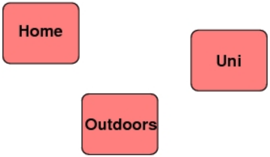
Context data types

- Context can have various data types
  - Nominal
  - Ordinal
  - Hierarchical
  - Numerical
- In multi-dimensional time series also multi-type contexts possible
- Most algorithms can only process some of these data types
  - Not applicable in scenarios where other data types are measured

Aspects of prediction algorithms

Context data types

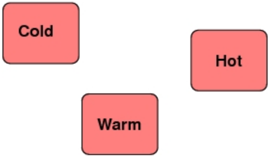
- Nominal contexts
  - $=$
  - $\neq$



Aspects of prediction algorithms

Context data types

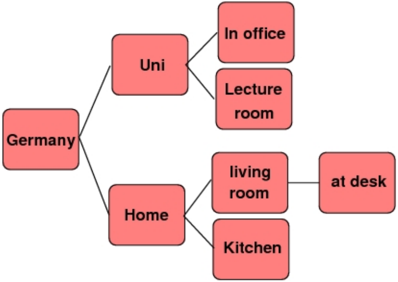
- Ordinal contexts
    - $<$
    - $>$
    - $=$



Aspects of prediction algorithms

Context data types

- Hierarchical contexts
  - Sub-contexts and parent contexts
  - Contexts might be contained in others



Aspects of prediction algorithms

Context data types

- Numerical contexts
  - Real valued, integer valued contexts
  - Complex mathematical operations possible
  - Best suited for context processing

Aspects of prediction algorithms

Context data types

Algorithm	Ordinal contexts	Nominal contexts	Hierarchical contexts	Numerical contexts
BN	+	+	+	+
SVM	-	-	-	+
KM	-	-	-	+
MM	+	+	+	+
NN	+	+	+	+
NNS	-	(+)[7]	(+)	+
SOM	-	(+)[7]	(+)[7]	+
PM	+	+	+	+
AP	(+)[7]	(+)[7]	(+)[7]	+
ARMA	-	-	-	+
Kalman filters	-	-	-	+

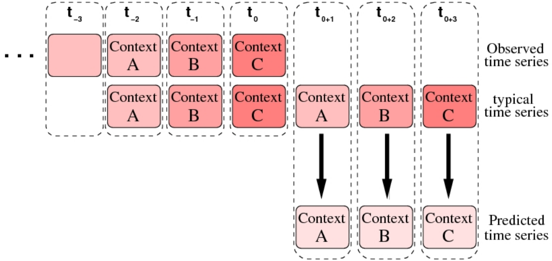
Outline

Simple prediction approaches: ONISI and IPAM

Exact sequence matching

Introduction

- File a given sequence for the exact occurence of a sub-sequence
- 'Pattern Matching' or 'String Matching'[1]
- Easily extended to context prediction:
  - Prediction ≡ continuation of matched sequence



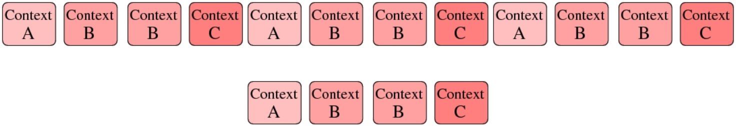
[1] Richard O. Duda, Peter E. Hard and David G. Stork, *Pattern Classification*, Wiley-Interscience, 2nd edition, 2001.

Exact sequence matching

Notation

Strings and patterns

A string is a sequence of letters such as 'AGCTTCGAATC'. Context patterns can be represented as strings when each context is assigned a letter.



Substring

Any contiguous string that is part of another string is called a substring. For example, 'GCT' is a substring of 'AGCTTC'.

Exact sequence matching

Notation

String matching

Given two Strings **x** and **y**, string matching is the problem to determine whether **x** is a substring of **y** and, if so, where it appears.

Edit distance

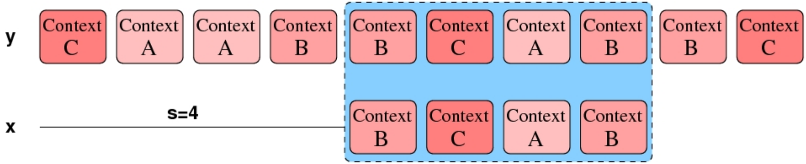
Given two strings **x** and **y**, the edit distance describes the minimum number of basic operations – character insertions, deletions and exchanges – needed to transform **x** into **y**.

Exact sequence matching

String matching

Basic string matching problem

For two strings **x** and **y**, determine whether a shift s at which the string **x** is perfectly matching with each caracter of **y** beginning at position $s + 1$.



Exact sequence matching

String matching

Straightforward approach

Subsequently test each possible shift s

Example

```
begin initialise Σ x,y,n=length[y], m=length[x]
    s ← 0
    while s ≤ n − m
       if x[1..m]=y[s + 1 ··· s + m]
          then print 'pattern occurs at shift' s
             s ← s + 1
    return
end
```

Exact sequence matching

String matching

- The straightforward algorithm is, however, far from optimal
- Worst case runtime:
  - $\Theta((n-m+1)m)$
- Problem: Information known from one candidate shift s is not exploited for the subsequent candidate shift

Exact sequence matching

String matching

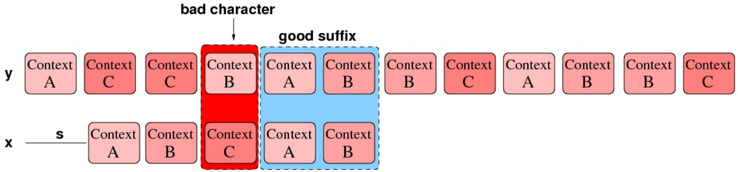
Boyer-Moore string matching

```
begin initialise Σ x,y,n=length[y], m=length[x]
    F(x) ← last-occurrence function
    G(x) ← good-suffic function
    s ← 0
    while s ≤ n − m
       do j ← m
       while j > 0 and x[j] = y[s + j]
          do j ← j − 1
       if j = 0
           then print 'pattern occurs at shift' s
              s ← s + G(0)
           else s ← s + max[G(j), j − F(y[s + j])]
     return
end
```

Exact sequence matching

Last occurrence function

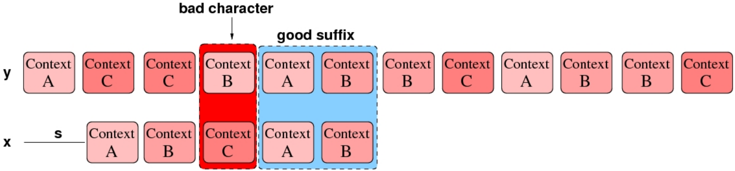
- Table containing every letter in the alphabet
- Plus position of its rightmost occurrence in **x**
- Example:
  - A, 4
  - B, 5
  - C, 3
- Computation only once
  - Does not significantly impact the runtime



Exact sequence matching

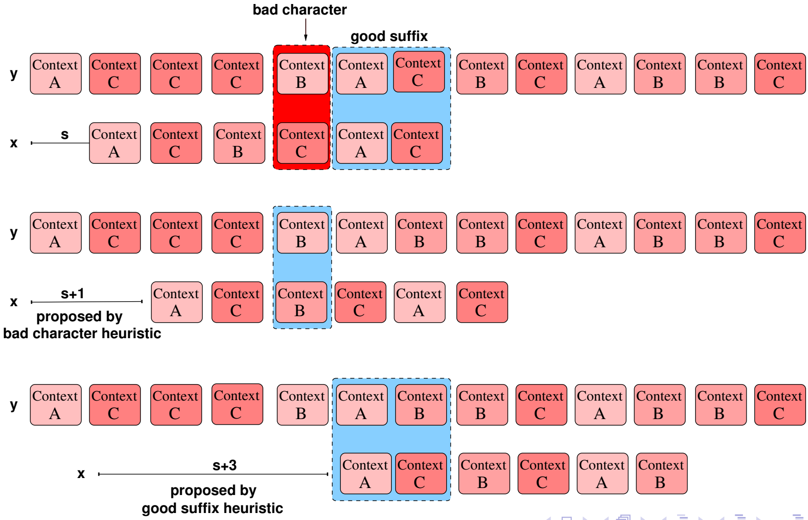
Good suffix function

- Creates table that for each suffix gives location of second right-most occurrence in **x**
- Example:
  - B, 2
  - AB, 1
  - CAB, -
  - BCAB, -
  - ABCAB, -
- Computation only once
  - Does not significantly impact the runtime



Exact sequence matching

Bad character heuristic and good suffix heuristic



Outline

Simple prediction approaches: ONISI and IPAM

IPAM

Introduction and scenario

Scenario

Predict the next command in a series of command line inputs to a UNIX shell

Prediction of next command on a UNIX shell

```
...
96102513:34:49 cd
96102513:34:49 ls
96102513:34:49 emacs
96102513:34:49 exit
96102513:35:32 BLANK
96102513:35:32 cd
96102513:35:32 cd
96102513:35:32 rlogin
96102513:35:32 exit
96102514:25:46 BLANK
96102514:25:46 cd
96102514:25:46 telnet
96102514:25:46 ps
96102514:25:46 kill
96102514:25:46 emasc
96102514:25:46 emacs
96102514:25:46 cp
```

IPAM

Algorithmic approach - operation principle

Step 1:

	$\cdots$	c_i	$\cdots$	c_{i+1}	$\cdots$

Step 2:

	$\cdots$	c_i	$\cdots$	c_{i+1}	$\cdots$
c_i		$\frac{1}{n}$	$\frac{1}{n}$	$\frac{1}{n}$	

Step 3:

	$\cdots$	c_i	$\cdots$	c_{i+1}	$\cdots$
c_i		$\frac{1}{n} \cdot \alpha + (1 - \alpha)$	$\frac{1}{n} \cdot \alpha$	$\frac{1}{n} \cdot \alpha$	
c_{i+1}		$\frac{1}{n}$	$\frac{1}{n}$	$\frac{1}{n}$	

IPAM

Example operation

Events: A,B,C

α : 0.8

- Step 1 – Input: A

	A	B	C
A	0.3333	0.3333	0.3333
Default	0.3333	0.3333	0.3333

- Prediction: A (0.3333), B (0.3333), C (0.3333)

IPAM

Example operation

Events: A,B,C

α : 0.8

- Step 2 – Input: A

	A	B	C
A	0.4667	0.2667	$0,2667$
Default	0.4667	0.2667	$0,2667$

- Prediction: A (0.4667), B (0.2667), C (0.2667)

IPAM

Example operation

Events: A,B,C

α : 0.8

- Step 3 – Input: B

	A	B	C
A	0.3733	0.4133	0.2133
B	0.3333	0.3333	0.3333
Default	0.3733	0.4133	0, 2133

- Prediction: A (0.3733), B (0.4133), C (0.2133)

IPAM

Example operation

Events: A,B,C

α : 0.8

- Step 4 – Input: A

	A	B	C
A	0.3733	0.4133	0.2133
B	0.4667	0.2667	0.2667
Default	0.4986	0.3306	0, 1706

- Prediction: A (0.3733), B (0.4133), C (0.2133)

IPAM

Example operation

Events: A,B,C

α : 0.8

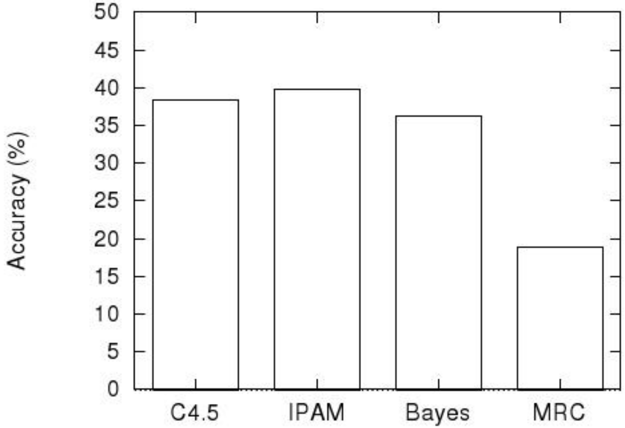
- Step 5 – Input: C

	A	B	C
A	0.2986	0.3306	0.3706
B	0.4667	0.2667	0.2667
C	0.3333	0.3333	0.3333
Default	0.3989	0.2645	0, 3706

- Prediction: A (0.3989), B (0.2645), C (0.3706)

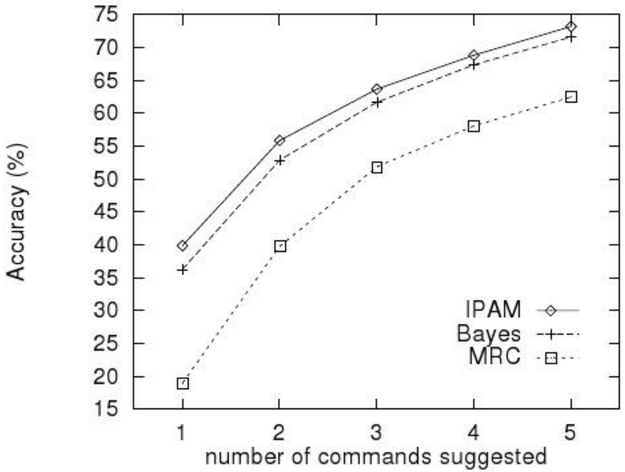
IPAM

Results and figures



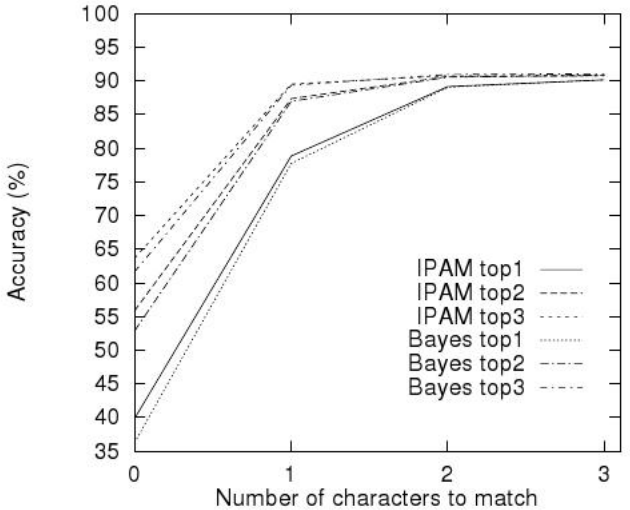
IPAM

Results and figures



IPAM

Prediction accuracy



Outline

ONISI

Algorithmic approach – Extraction of observed pattern

- Length of patterns automatically varied
  - Longer patterns are deemed more important
  - Patterns are chosen to be longest sequences in histroy that match immediate history

Measure 1: Length

Sequences that predict action a are computed by $l_t(s, a)$

Average of lenghts of k longest sequences that end with action a in state s and match history sequence immediately prior to time t

- Possible actions are ranked according to $l_t(s, a)$
- $\frac{l_t(s,a)}{\sum_i l_t(s,a_i)}$

ONISI

Algorithmic approach – Extraction of observed pattern

- Length of patterns automatically varied
  - More frequent patterns are deemed more important

Measure 2: Frequency

Sequences that prediction action a are computed by $f_t(s, a)$

Frequency at which a sequence is observed in history

- Possible actions are ranked according to $f_t(s, a)$
- $\frac{l_t(s,a)}{\sum_i l_t(s,a_i)}$

ONISI

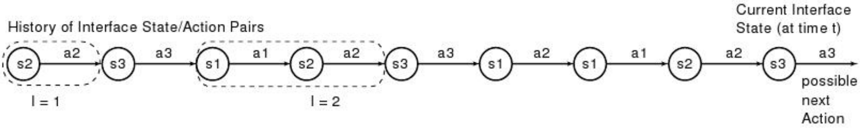
Algorithmic operation

- Compare immediate history with state-action pair (s, a)
  - Running backwards through recorded history
  - Find k longest sequences that match immediate history
- Average length of sequences: $l_t(s, a)$
- Count number of times a sequence has occurred: $f_t(s, a)$
- Return ranking

$$R_t(s, a) = \alpha \frac{l_t(s, a)}{\sum_i l_t(s, a_i)} + (1 - \alpha) \frac{f(s, a)}{\sum_i f(s, a_i)} \qquad (1)$$

ONISI

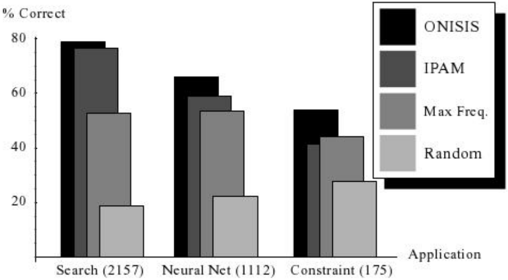
Algorithmic operation



- Assume:
  - $\alpha = 0.9$
  - All actions provide a sum $\sum_i l_t(s, a) = 5$
  - a_3 has occured 50 times, s_3 has been visited 100 times
- Set of maximum length sequences: $\{2,1,0\}$
- $$l_t(s_3, a_3) = \frac{0 + 1 + 2}{3} = 1 \quad (2)$$
- $$R_t(s_3, a_3) = 0.9\frac{1}{5} + 0.1\frac{50}{100} = 0.18 + 0.05 = 0.23 \quad (3)$$

ONISI

Prediction accuracy – Performance



Outline

Alignment prediction

Basic definitions

Alignment

Let $s = s_1 \dots s_m$ and $t_1 \dots t_n$ be two strings over an alphabet Σ and $- \notin \Sigma$ a gap symbol. Let $\Sigma' = \Sigma \cup \{-\}$. Let $h : (\Sigma')^* \to \Sigma^*$ be a homomorphism defined by $h(a) = a$ for all $a \in \Sigma$ and $h(-) = \lambda$.

- An alignment between s and t is a pair (s', t') of length $l \geq \max\{m, n\}$ over Σ' that follows the constraints
  - $|s'| = |t'| \geq \max\{|s|, |t|\}$
  - $h(s') = s$
  - $h(t') = t$
  - $\forall i \in \{1 \dots l\} : s'_i \neq -$ or $t'_i \neq -$

Alignment prediction

Basic definitions

- Example
  - $s = GACGGATTATG$
  - $t = GATCGGAATAG$
  - One possible alignment:
    - $s' = GA\_CGGA\underline{T}TATG$
    - $t' = GATCGGA\underline{A}TA\_G$

Alignment prediction

Basic definitions

- Example
  - $s = GACGGATTATG$
  - $t = GATCGGAATAG$
  - One possible alignment:
    - $s' = GA\_CGGA\underline{T}TATG$
    - $t' = GATCGGA\underline{A}TA\_G$
- Possible operations:

Insertion	The first string contains a gap in this column
Deletion	The second string contains a gap in this column
Match	Both strings are identical in this column
Mismatch	The strings do not match but the column also does not contain a gap.

Alignment prediction
Basic definitions

Alignment score

Let $p(a,b) \in \mathbb{Q}$ for all $a,b \in \Sigma$ and $g \in \mathbb{Q}$. The alignment score $\delta(s',t')$ for $s' = s'_1 \dots s'_l$ and $t'_1 \dots t'_l$ is defined as

$$\delta(s',t') = \sum_{i=1}^{l} \delta(s'_i, t'_i) \tag{4}$$

With

$$\delta(x,y) = \begin{cases} p(x,y) & x,y \in \Sigma \\ g & x = - \\ g & y = - \end{cases} \tag{5}$$

The optimasation goal is $goal_\delta \in \{\min, \max\}$

Alignment prediction

Global alignment

Global alignment problem

Input Two strings s and t over Σ and an alignment score δ with the optimisation aim $goal_\delta$

Valid solutions All alignments of s and t

Cost For each alignment $A = (s', t')$: $cost(A) = \delta(A)$

Optimisation aim $goal_\delta$

Alignment prediction

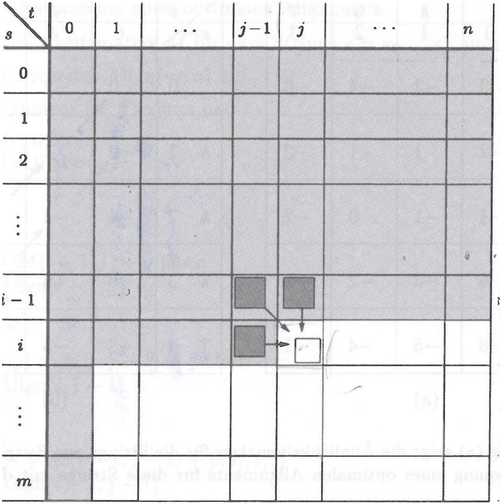
Global alignment

- Calculation of the global alignment between two strings s and t by integer programming:

$$sim(s_1 \ldots s_i, t_1 \ldots t_j) = goal_\delta \begin{cases} \underbrace{sim(s_1 \ldots s_{i-1}, t_1 \ldots t_j) + g}_{insertion} \\ \underbrace{sim(s_1 \ldots s_i, t_1 \ldots t_{j-1}) + g}_{deletion} \\ \underbrace{sim(s_1 \ldots s_{i-1}, t_1 \ldots t_{j-1}) + p(s_i, t_j)}_{Match/Mismatch} \end{cases}$$

Alignment prediction

Global alignment



Alignment prediction

Global alignment

Calculation of similarity

Input: $s = s_1 \dots s_m$, $t = t_1 \dots t_n$

Output: $sim(s,t) = M(m,n)$

for $i = 0$ to m do — *Initialisation*

    for $j = 0$ to n do

        $M(i,j) := 0$

for $i = 0$ to m do — *Initialise borders*

    $M(i,0) = i \cdot g$

for $j = 0$ to n do

    $M(0,j) = j \cdot g$

for $i = 1$ to m do — *Fill out matrix*

    for $j = 1$ to n do

        $M(i,j) := \max\{M(i-1,j)+g, M(i,j-1)+g,$
        $M(i-1,j-1)+p(s_i,s_j)\}$

Alignment prediction

Global alignment

Calculation of an optimum alignment

Input: Similarity matrix M

Output: Alignment (s', t')

Align (i,j) –Recursive procedure

if $i = j = 0$ then
    $s' := t' := \lambda$
else if $M(i,j) = M(i-1,j) + g$ then
    $(\overline{s}, \overline{t}) := Align(i-1, j)$
    $s' := \overline{s} \cdot s_i$; $t' := \overline{t} \cdot -$
    else if $M(i,j) = M(i,j-1) + g$ then
        $(\overline{s}, \overline{t}) := Align(i, j-1)$
        $s' := \overline{s} \cdot -$; $t' := \overline{t} \cdot t_j$
    else $\{M(i,j) = M(i-1,j-1) + p(s_i, t_j)\}$
        $(\overline{s}, \overline{t}) := Align(i-1, j-1)$
        $s' := \overline{s} \cdot s_i$; $t' := \overline{t} \cdot t_j$
return (s',t')

Alignment prediction

Global alignment

s \ t	0	A 1	G 2	T 3
0	0	−2	−4	−6
A 1	−2	1	−1	−3
A 2	−4	−1	0	−2
A 3	−6	−3	−2	−1
T 4	−8	−5	−4	−1

s \ t	0	A 1	G 2	T 3
0	0	−2	−4	−6
A 1	−2	1	−1	−3
A 2	−4	−1	0	−2
A 3	−6	−3	−2	−1
T 4	−8	−5	−4	−1

Alignment prediction

Global alignment

- Computational complexity to calclate global alignment
  - Time to compute the similarity matrix: $O(nm)$
  - Calculation of the optimum alignment: $O(n+m)$
  - Overall computation time: $O(nm)$
- Algorithm can also be extended to compute all alignments
  - Worst case: count of optimum alignments is exponential
  - Consequently, the WC runtime is also exponential.

Alignment prediction

Local and semiglobal alignments

Local alignment problem

Input Two strings s and t over Σ and an alignment score δ with the optimisation aim $goal_\delta$

Valid solutions All local alignments of s and t

Cost For a local alignment $A = (\bar{s}', \bar{t}')$: $cost(A) = \delta(A)$

Optimisation aim Maximisation

- For local alignments, the optimisation aim is always maximisation.
- If the optimisation aim were minimisation, the resulting alignment were often very short (i.e. only one symbol)

Alignment prediction

Local and semiglobal alignments

- Example
  - $s = AAAAACTCTCTCT$
  - $t = GCGCGCGCAAAAA$
  - $\delta = \begin{cases} 1 & x = y \\ -1 & x \neq y \\ -2 & x = -; y = - \end{cases}$

Alignment prediction

Local and semiglobal alignments

- Example
  - $s = AAAAACTCTCTCT$
  - $t = GCGCGCGCAAAAA$
  - $\delta = \begin{cases} 1 & x = y \\ -1 & x \neq y \\ -2 & x = -; y = - \end{cases}$
- Optimum local alignment
  - AAAAA(CTCTCTCT)
  - (GCGCGCGC)AAAAA
  - Alignment score: 5

Alignment prediction

Local and semiglobal alignments

- Example
  - $s = AAAAACTCTCTCT$
  - $t = GCGCGCGCAAAAA$
  - $\delta = \begin{cases} 1 & x = y \\ -1 & x \neq y \\ -2 & x = -; y = - \end{cases}$
- Optimum global alignment
  - AAAAACTCTCTCT
  - GCGCGCGCAAAAA
  - Alignment score: -11

Alignment prediction

Local and semiglobal alignments

- We can calculate the optimum local alignment with a modified version of the algorithm for calculating the optimum global alignment
  - $$M(i,j) = max \begin{cases} M(i-1,j)+g, \\ M(i,j-1)+g, \\ M(i-1,j-1)+p(s_i,s_j) \\ 0 \end{cases}$$
- Row 0 and column 0 are initialised with 0
  - Suffix and prefix are disregarded

Alignment prediction

Local and semiglobal alignments

- Semiglobal alignment
  - Align whole strings
  - Gap symbols at the beginning or at the end of the strings are for free

Alignment prediction

Local and semiglobal alignments

- Example
  - $s = ACTTTATGCCTGCT$
  - $t = ACAGGCT$
  - $\delta = \begin{cases} 1 & x = y \\ -1 & x \neq y \\ -2 & x = -; y = - \end{cases}$
- Optimum global alignment
  - ACTTTATGCCTGCT
  - AC_ _ _ A_ G_ _ _GCT
  - Alignment score: -7

Alignment prediction

Local and semiglobal alignments

- Example
  - $s = ACTTTATGCCTGCT$
  - $t = ACAGGCT$
  - $\delta = \begin{cases} 1 & x = y \\ -1 & x \neq y \\ -2 & x = -; y = - \end{cases}$
- Optimum semiglobal alignment
  - ACTTTAT_ GCCTGCT
  - _ _ _ _ _ACAGGCT_ _ _
  - Alignment score: 0

Alignment prediction

Local and semiglobal alignments

- Example
  - $s = ACTTTATGCCTGCT$
  - $t = ACAGGCT$
  - $\delta = \begin{cases} 1 & x = y \\ -1 & x \neq y \\ -2 & x = -; y = - \end{cases}$
- Optimum local alignment
  - (ACTTTATGCCT)GCT
  - (ACAG)GCT
  - Alignment score: 3
  - But: Only short sequence aligned compared to the semiglobal alignment

Alignment prediction

Local and semiglobal alignments

- Types of semiglobal alignments
  - Variants can be combined with each other

Gap symbols for free	Modification of the algorithm
Beginning of first string	Initialise first row of M with 0
End of first string	Similarity corresponds to the maximum of the last row
Beginning of second string	Initialise first column of M with 0
End of second string	Similarity corresponds to the maximum of the last column

Alignment prediction

Local and semiglobal alignments

- Example
  - $s = AAAT$
  - $t = AGTA$
  - $\delta = \begin{cases} 1 & x = y \\ -1 & x \neq y \\ -2 & x = -; y = - \end{cases}$

Alignment prediction

Local and semiglobal alignments

s \ t		0	A 1	G 2	T 3	A 4
	0	0	−2	−4	−6	−8
A	1	0	1	−1	−3	−5
A	2	0	1	0	−2	−2
A	3	0	1	0	−1	−1
T	4	0	−1	0	1	−1

Alignment prediction

Local and semiglobal alignments

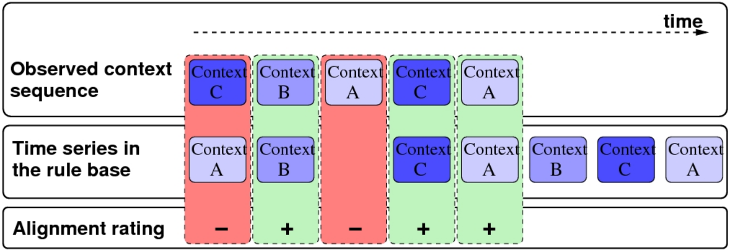
- Example
  - $s = AAAT$
  - $t = AGTA$
  - $\delta = \begin{cases} 1 & x = y \\ -1 & x \neq y \\ -2 & x = -; y = - \end{cases}$
- Optimum semiglobal alignment
  - AAAT_
  - _ AGTA
  - Alignment score: 1

Outline

Alignment prediction approaches

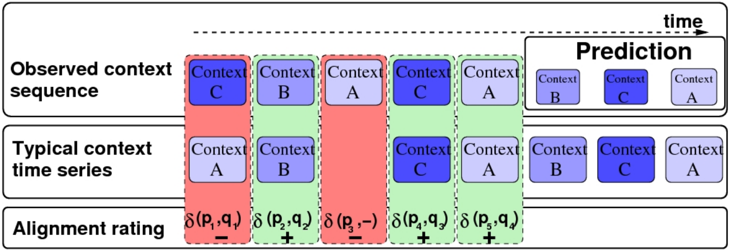
Prediction with alignment methods

Prediction procedure



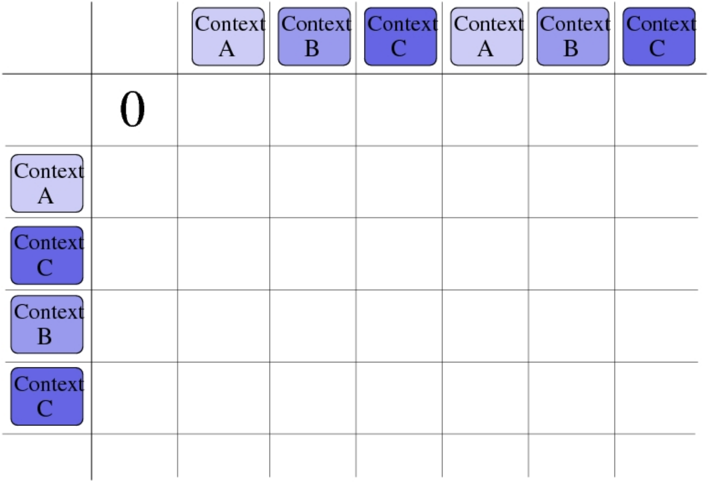
Prediction with alignment methods

Prediction procedure



Prediction with alignment methods

Example



Prediction with alignment methods

Example

		Context A	Context B	Context C	Context A	Context B	Context C
	0	0	0	0	0	0	0
Context A	0						
Context C	0						
Context B	0						
Context C	0						

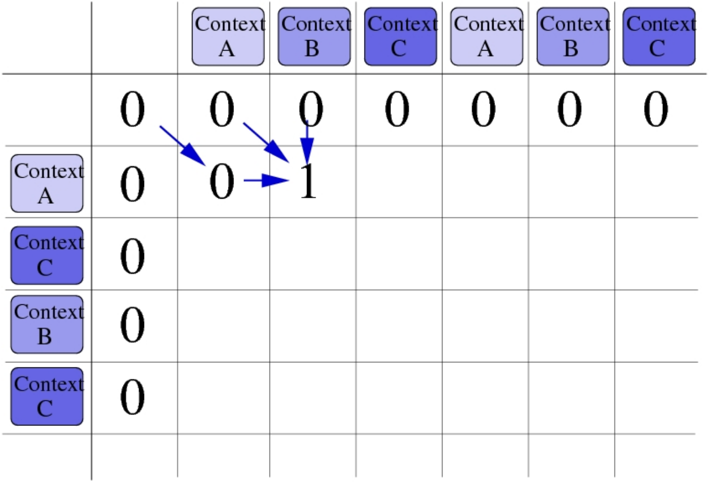
Prediction with alignment methods

Example

		Context A	Context B	Context C	Context A	Context B	Context C
	0	0	0	0	0	0	0
Context A	0	0					
Context C	0						
Context B	0						
Context C	0						

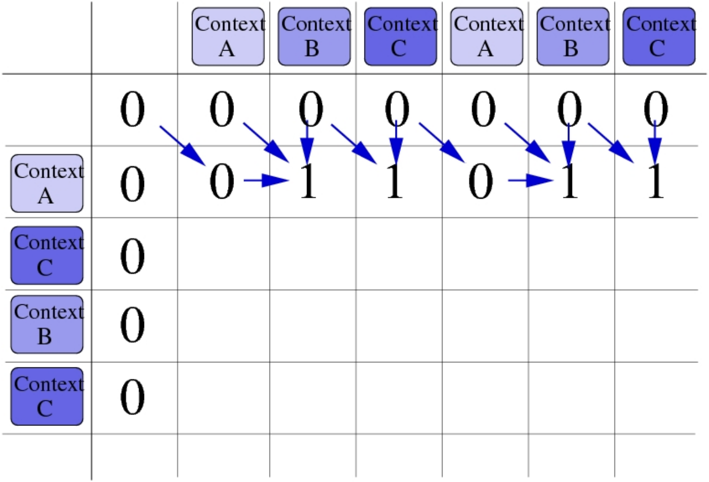
Prediction with alignment methods

Example



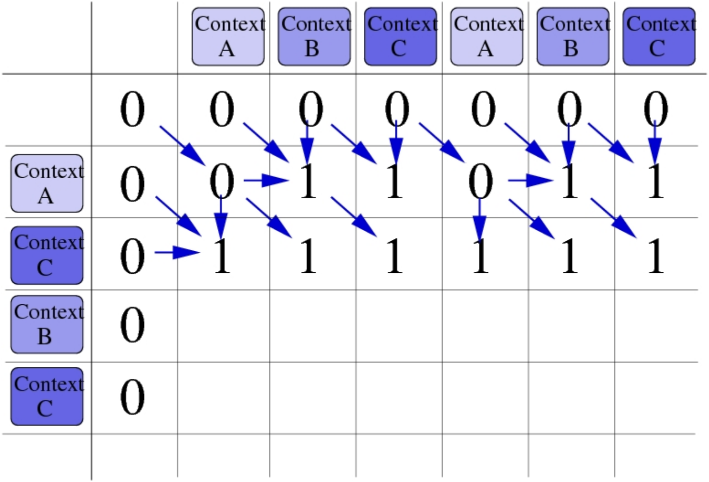
Prediction with alignment methods

Example



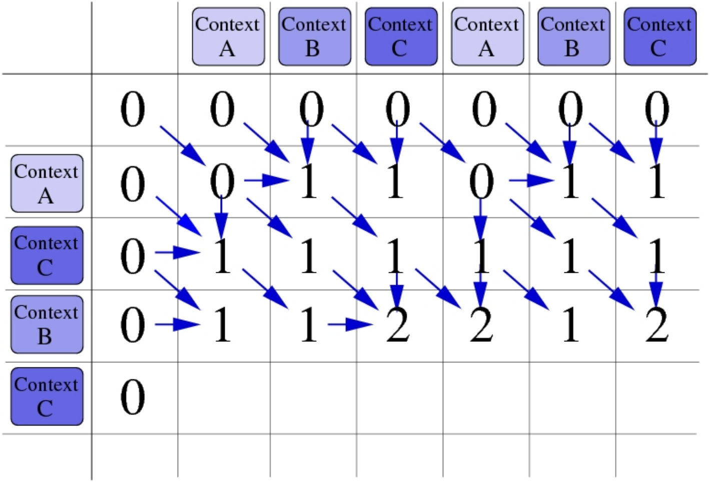
Prediction with alignment methods

Example



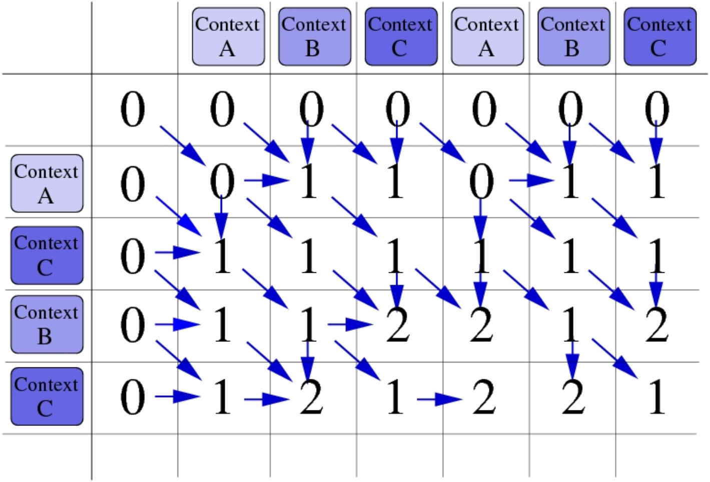
Prediction with alignment methods

Example



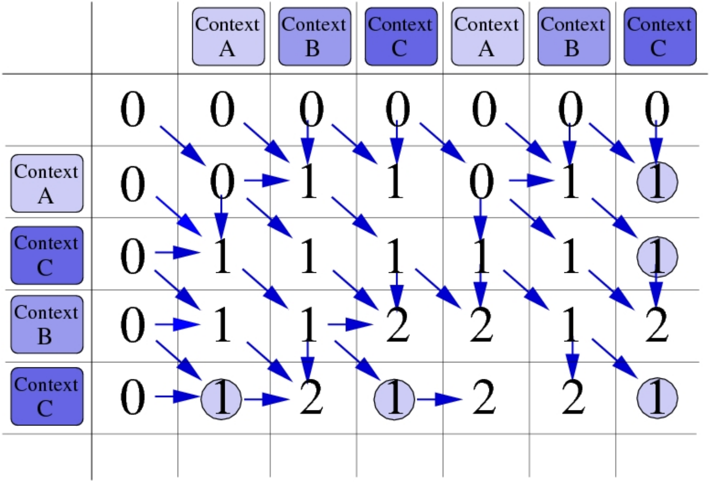
Prediction with alignment methods

Example



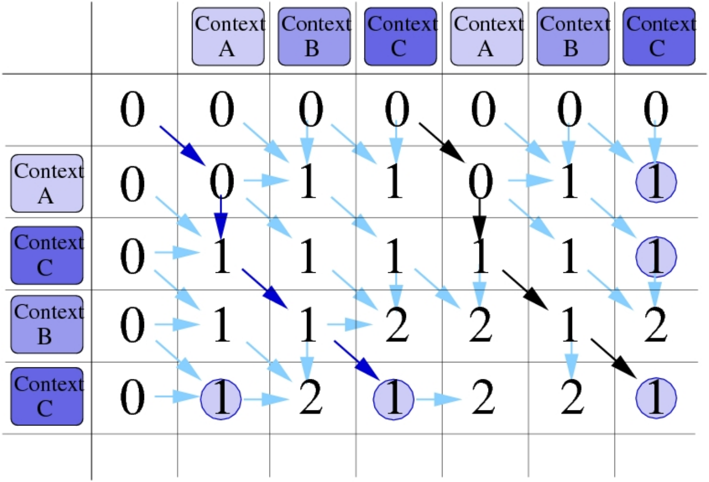
Prediction with alignment methods

Example



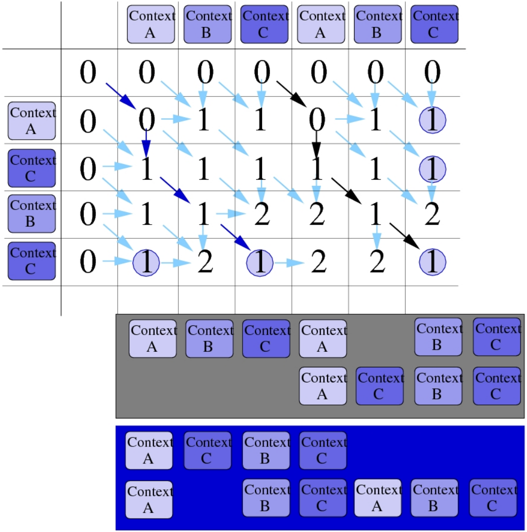
Prediction with alignment methods

Example



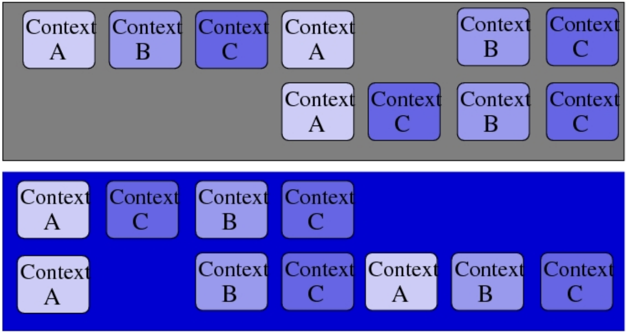
Prediction with alignment methods

Example



Prediction with alignment methods

Example



Questions?

Stephan Sigg
sigg@nii.ac.jp

Literature

- C.M. Bishop: Pattern recognition and machine learning, Springer, 2007.
- P. Tulys, B. Skoric, T. Kevenaar: Security with Noisy Data – On private biometrics, secure key storage and anti-counterfeiting, Springer, 2007.
- R.O. Duda, P.E. Hart, D.G. Stork: Pattern Classification, Wiley, 2001.