Algorithms for context prediction in Ubiquitous Systems

Prediction by alignment methods

Stephan Sigg

Institute of Distributed and Ubiquitous Systems
Technische Universität Braunschweig

January 6, 2009

Overview and Structure

- Introduction to context aware computing
- Basics of probability theory
- Algorithms
  - Simple prediction approaches: ONISI and IPAM
  - Markov prediction approaches
  - The State predictor
  - Alignment prediction
  - Prediction with self organising maps
  - Stochastic prediction approaches: ARMA and Kalman filter
  - Alternative prediction approaches
    - Dempster shafer
    - Evolutionary algorithms
    - Neural networks
    - Simulated annealing

Overview and Structure

- Introduction to context aware computing
- Basics of probability theory
- **Algorithms**
  - Simple prediction approaches: ONISI and IPAM
  - Markov prediction approaches
  - The State predictor
  - **Alignment prediction**
  - Prediction with self organising maps
  - Stochastic prediction approaches: ARMA and Kalman filter
  - Alternative prediction approaches
    - Dempster shafer
    - Evolutionary algorithms
    - Neural networks
    - Simulated annealing

Outline

Alignment prediction approaches

1. Introduction to alignment methods
   - Alignment of two strings
   - Heuristical approaches for database search
   - Multiple alignments
2. Prediction with alignment methods
   - Application scenarios
3. Properties of the alignment prediction approach

Introduction to alignment methods

Introduction

- Approximative string matching
  - Is a string approximatively contained in another string?
  - Required: Similarity between strings
  - Align strings to each other and add some gaps so that the remaining positions are maximally matching
  - Similarity metric defines similarity between sub-sequences of strings

Introduction to alignment methods

Introduction

- Application domains
  - Computational biology
  - Matching of DNA or Proteine sequences
  - Compare several sequences of the same genome (e.g. from various experiments or laboratories)
  - Search for a string as substring in a string data base

Introduction to alignment methods

Alignment of two strings

- We will discuss approaches to calculate an optimal alignment between two strings[1]
  - What is an alignment
  - Efficient algorithm to calculate an optimal alignment between two strings
  - Variants and Extensions of the approach

[1] Hans-Joachim Böckenhauer and Dirk Bongartz, *Algorithmische Grundlagen der Bioinformatik*, Teubner, 2003

Introduction to alignment methods

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 $h : (\Sigma')^* \rightarrow \Sigma^*$ 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 $t'_i \neq -$

Introduction to alignment methods

Basic definitions

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

Introduction to alignment methods

Basic definitions

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

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.

Introduction to alignment methods

Basic definitions

- Possible columns:

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.

- Interpretation
  - Alignment as sequence of insertion and deletion operands on the first string to obtain the second string.
  - Origin of this interpretation in computational biology

Introduction to alignment methods

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

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

With

$$\delta(x, y) = \begin{cases} p(x, y) & x, y \in \Sigma \\ g & x = - \\ g & y = - \end{cases} \tag{2}$$

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

Introduction to alignment methods

Basic definitions

Alignment similarity

The similarity sim_δ of two strings s and t with regard to δ is the score of an optimal alignment:

$$sim_\delta(s,t) = goal_\delta\{\delta(s',t')|(s',t')\text{ist ein Alignment von s und t}\} \tag{3}$$

Introduction to alignment methods

Global alignment

- Global alignment
  - Alignment of two strings s and t
- Local alignment
  - Alignment of substrings from s and t
- Semiglobal alignment

Introduction to alignment methods

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	$goal_\delta$

Introduction to alignment methods

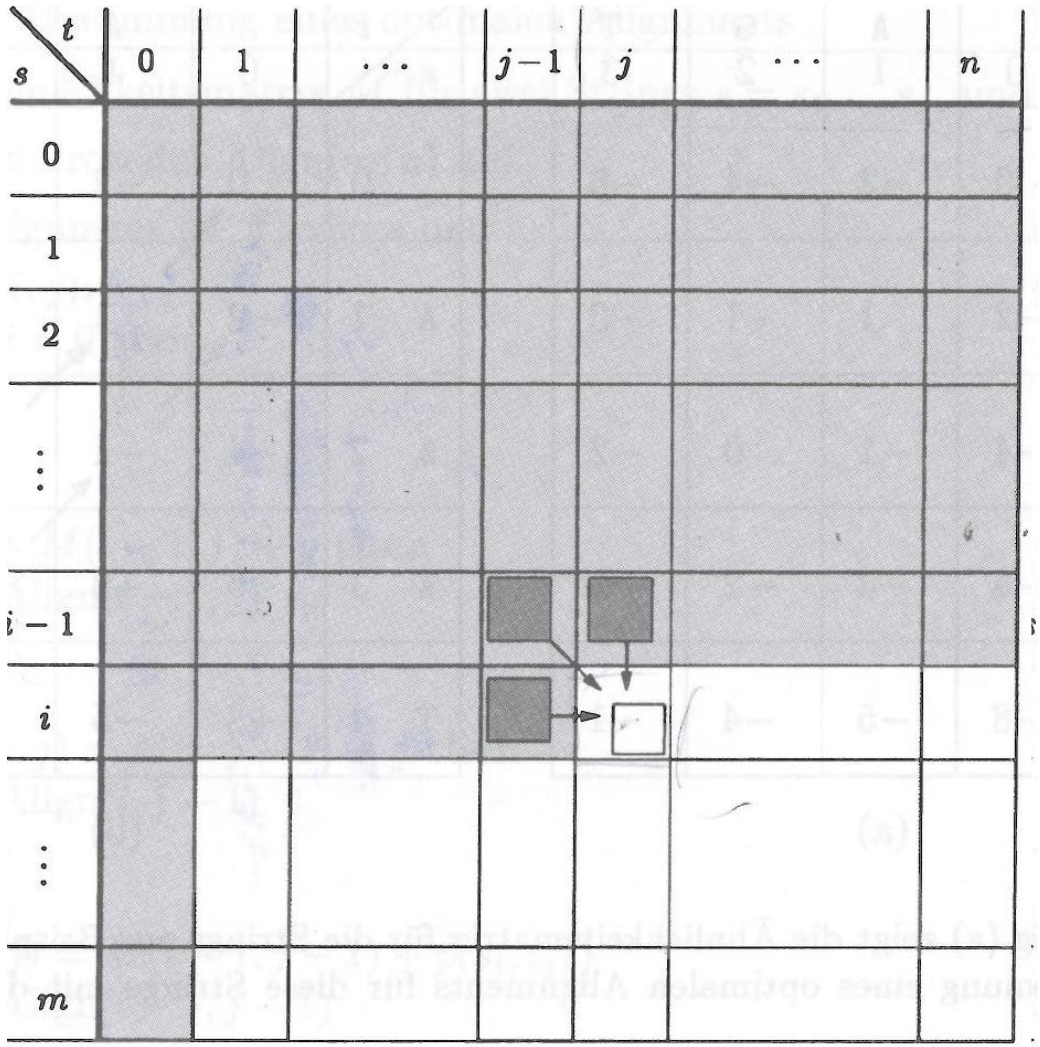
Global alignment

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

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

Introduction to alignment methods

Global alignment



Introduction to alignment methods

Global alignment

- Initialisation: Row 0 and column 0
  - Multiples of g
- Fill the matrix by integer programming
  - row after row or column after column
- Every possible path from (0,0) to (m,n) is one possible alignment between s and t
- The Algorithm calculates the cheapest path or the optimum alignment[2]

[2] S.B. Needleman and C.D. Wunsch, *A general method applicable to the search for similarities in the amino acid sequence of two proteins*, Journal of Molecular biology 48, pp443-453, 1970.

Introduction to alignment methods

Global alignment

Calculation of similarity

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

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

```
1  for i = 0 to m do                                   Initialisation
2      for j = 0 to n do
3          M(i,j) := 0
4  for i = 0 to m do                                   Initialise borders
5      M(i,0) = i · g
6  for j = 0 to n do
7      M(0,j) = j · g
8  for i = 1 to m do                                   Fill out matrix
9      for j = 1 to n do
10         M(i,j) := max{M(i−1,j) + g, M(i,j−1) + g,
                        M(i−1,j−1) + p(s_i, s_j)}
```

Introduction to alignment methods

Global alignment

Calculation of an optimum alignment

```
Input:  Similarity matrix M
Output:  Alignment (s', t')
1 for i = j = 0 then                          Align (i,j) –Recursive procedure
2     for s' := t' := λ
3 else if M(i,j) = M(i-1,j) + g then
4     (s̄, t̄) := Align(i-1, j)
5     s' := s̄ · s_i ;  t' := t̄ · –
6     else if M(i,j) = M(i,j-1) + g then
7         (s̄, t̄) := Align(i, j-1)
8         s' := s̄ · –  ;  t' := t̄ · t_j
9     else {M(i,j) = M(i-1,j-1) + p(s_i, t_j)}
10        (s̄, t̄) := Align(i-1, j-1)
11        s' := s̄ · s_i  ;  t' := t̄ · t_j
12 return (s',t')
```

Introduction to alignment methods

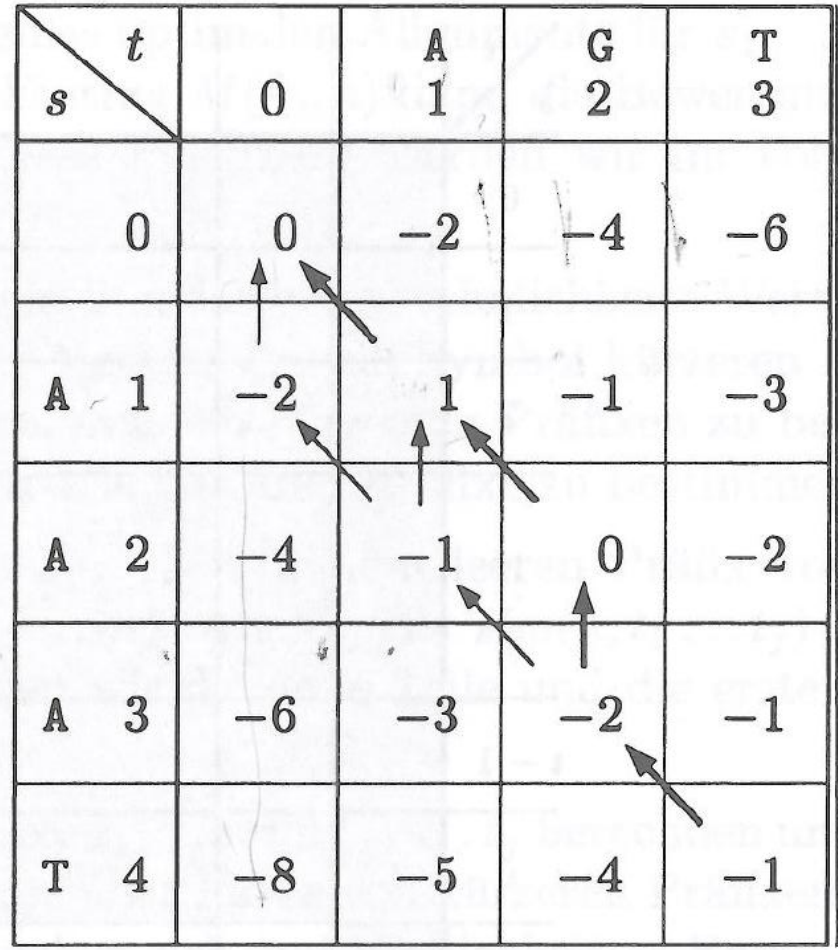
Global alignment

- Example
  - $s = AAAT$
  - $t = AGT$
  - $p(x, y) = \begin{cases} 1 & x = y \\ -1 & x \neq y \\ -2 & x = -; y = - \end{cases}$

Introduction to alignment methods

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



Introduction to alignment methods

Global alignment

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

Introduction to alignment methods

Local and semiglobal alignments

- Alignment of substrings of two input strings s and t.
- Generalisation of the global alignment problem

Introduction to alignment methods

Local and semiglobal alignments

Local alignment

A local alignment of two strings s and t is a global alignment of the substrings $\overline{s} = s_{i_1} \ldots s_{i_2}$ and $\overline{t} = t_{i_1} \ldots t_{i_2}$.
an Alignment $A = (\overline{s}', \overline{t}')$ of the substrings $\overline{s}$, $\overline{t}$ is an optimal local alignment of s and t, if

$$\delta(A) = \max\{sim(\overline{s}, \overline{t}) | \overline{s} \text{ is a substring of s}, \overline{t} \text{ is a substring of t}\}$$

Introduction to alignment methods

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 $A = (\overline{s}', \overline{t}')$: $cost(A) = \delta(A)$
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)

Introduction to alignment methods

Local and semiglobal alignments

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

Introduction to alignment methods

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

Introduction to alignment methods

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

Introduction to alignment methods

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

Introduction to alignment methods

Local and semiglobal alignments

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

Introduction to alignment methods

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

Introduction to alignment methods

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

Introduction to alignment methods

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

Introduction to alignment methods

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

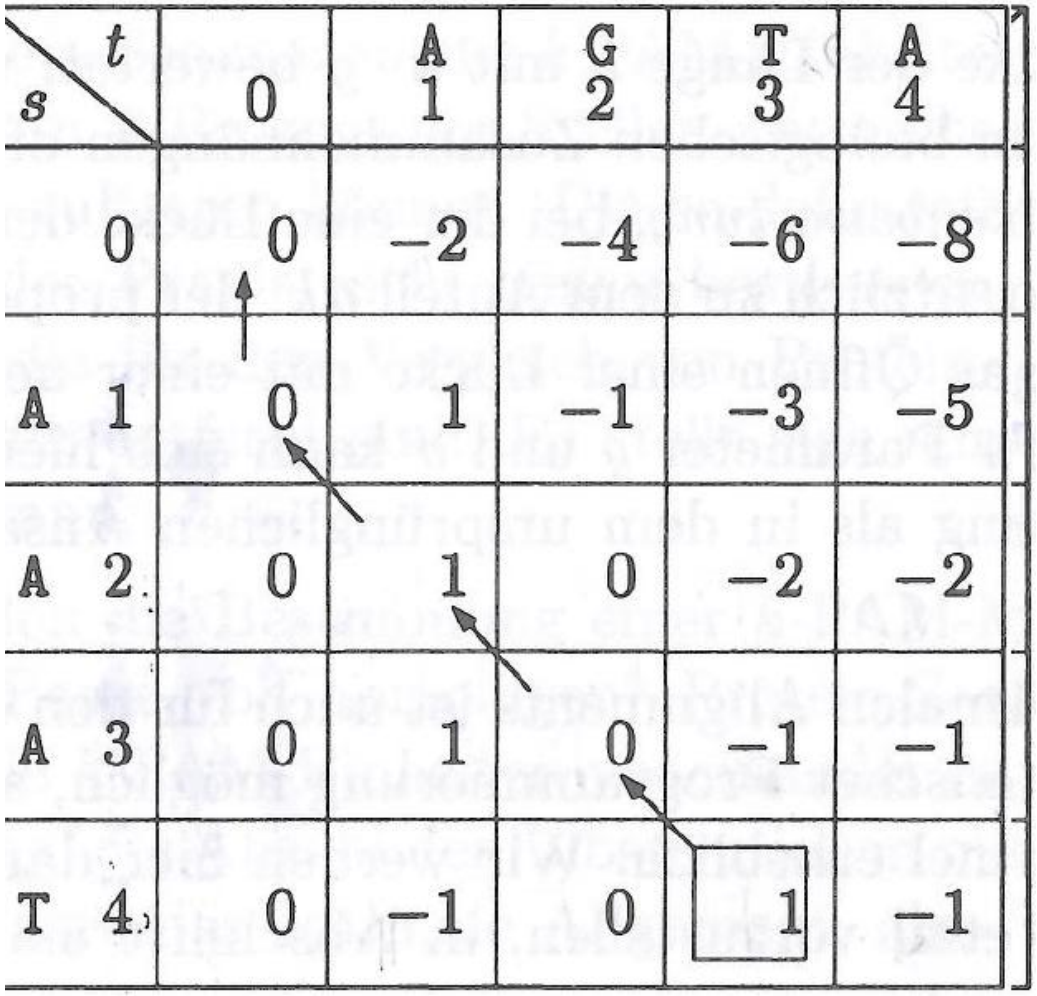
Introduction to alignment methods

Local and semiglobal alignments

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

Introduction to alignment methods

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

Introduction to alignment methods

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

Introduction to alignment methods

Generalised scoring function

- The scoring function defines the properties of a given scenario
  - By modifying the scoring function, the alignment cost is adapted to the scenario
- Cost for gap symbols
- Scoring matrices

Introduction to alignment methods

Generalised scoring function

- Cost for gap symbols
  - Various gaps that occur en block model the case that a novel context sequence was interleaved
  - The observed context sequence differs from the typical sequence
    - Interruption
    - Seldom action sequence
  - Blocks of gap symbols should then have fewer costs than single gap symbols

Introduction to alignment methods

Generalised scoring function

Gap of length k

A substring $s'_{i+1} \dots s'_{i+k} = -^k$ with $s'_i, s'_{i+k+1} \neq -$ is called a gap of length k[a]

[a] In the literature, a singe gap is often referred to as a space while a gap of any length is called a gap

Introduction to alignment methods

Generalised scoring function

- Recently, a gap of lenght k had cost $k \cdot g$
- Another approach is the affine gap score:
  - A gap of length k has cost $-(\varrho + \sigma k)$
  - $\varrho, \sigma > 0$
  - In addition to the cost σk for the lenght of the gap, the opening of the gap has cost ϱ
- The calculation of affine gap symbols is possible by the algorithms detailed above.
- The recursion, however, becomes more complicated

Introduction to alignment methods

Generalised scoring function

- Scoring matrices
  - Transitions between some contexts might be more probable than between others
  - The scoring function might respect this aspect by weighting the cost for various transitions differently

Introduction to alignment methods

Generalised scoring function

- Scoring matrices
  - $\Sigma \times \Sigma$ scoring matrix
  - Every transition between context has distinct cost
  - Two approaches in the literature[3]
    - PAM-Matrices
    - BLOSUM-Matrices

[3] The notation have their origin in computational biology

Introduction to alignment methods

Generalised scoring function

- PAM Matrices

Accepted mutation

An accepted mutationis a mutation that only slightly modifies a context

Introduction to alignment methods

Generalised scoring function

- PAM Matrices

PAM-interval

Two sequences s and t are one PAM-interval apart, if s can be modified to t by accepted point mutations (substitution of contexts, no insertion or deletion), so that on average one accepted mutation occurs per 100 symbols

Introduction to alignment methods

Generalised scoring function

- PAM Matrices

k-PAM matrix

A k-PAM matrix is a scoring matrix to compare sequences that are k PAM-intervals apart.

- How do we create a k-PAM matrix?

Introduction to alignment methods

Generalised scoring function

- PAM Matrices
- How to create k-PAM matrices (cont.)
  - Assumptions
    - Set of training sequence pairs given, that are k PAM intervals apart
    - For every pair, the optimum alignment is known/given
  - A: Set of optimum alignments
  - $Sp(A)$: Multiset of all columns that do not contain a gap symbol

Introduction to alignment methods

Generalised scoring function

- PAM Matrices
- How to create k-PAM matrices
  - Relative frequency of columns with (a_i, a_j) or (a_j, a_i) from all columns in $Sp(A)$: $freq(a_i, a_j)$

$$freq(a_i, a_j) = \frac{\text{count of pairs } (a_i, a_j) \text{ or } (a_j, a_i) \text{ in } Sp(A)}{2 \cdot |Sp(A)|} \quad (4)$$

  - Relative frequency of a_i in all alignments:

$$freq(a_i) = \frac{\text{count of occurences of } a_i \text{ in all alignments}}{\text{overall length of all sequences}} \quad (5)$$

  - The PAM matrix is then defined by

$$PAM_k(i,j) = \log \frac{freq(a_i, a_j)}{freq(a_i) \cdot freq(a_j)} \quad (6)$$

Introduction to alignment methods

Generalised scoring function

- PAM Matrices
- Explanation

$$PAM_k(i,j) = \log \frac{freq(a_i, a_j)}{freq(a_i) \cdot freq(a_j)} \tag{7}$$

- Entry (i,j) in the PAM matrix describes the relation between the probability to mutate a_i to a_j and the probability that this pair occurs in an alignment uniformly random.
- For ease of calculation with these values, the logarithm is applied to this result

Introduction to alignment methods

Generalised scoring function

- PAM Matrices
- Problem
  - Typically no such training sequences available
- Idea
  - Take a set of very similar sequences that most probably constitute one typical context sequence
  - Assume that these sequences are only one PAM interval apart
  - Calculate $freq(a_i, a_j)$ and $freq(a_i)$
  - Then: Calculation of PAM matrix for greater values possible

Introduction to alignment methods

Generalised scoring function

- PAM Matrices
- Calculate k-PAM matrices
  - Let $F(i,j)$ be the probability that a_i mutates in one PAM interval to a_j
  - $F^k = \underbrace{F \cdot F \cdot \ldots \cdot F}_{\text{k times}}$
  - Derive k-PAM-matrix as

$$PAM_k(i,j) = \log \frac{freq(a_i) \cdot F^k(i,j)}{freq(a_i) freq(a_j)} = \log \frac{F^k(i,j)}{freq(a_j)} \qquad (8)$$

Introduction to alignment methods

Generalised scoring function

- BLOSUM Matrices
  - Calculation of matrix based on data base
  - Data base contains information about similar and typical sequences
  - BLOSUM matrix therefore based on empirically derived sequences

Outline

Alignment prediction approaches

1. Introduction to alignment methods
   - Alignment of two strings
   - Heuristical approaches for database search
   - Multiple alignments
2. Prediction with alignment methods
   - Application scenarios
3. Properties of the alignment prediction approach

Heuristical approaches for database search

Introduction

- Exact calculation of alignments polynomial runtime but too slow.
- Practical implementations utilise fast heuristics
  - Faster than exact approaches
  - Optimum solution not guaranteed
- Popular approaches
  - FASTA[4]
  - BLAST[5]

[4] W.R. Pearson and D.J. Lipman, *Improved tools for biological sequence comparison*, Proceedings of the National academy of sciences of the U.S.A. 85, pp 2444-2448, 1988.

[5] S.F. Altschul, T.L. Madden, A. Zhang, Z. Zhang, W. Miller and D.J. Lipman, *Gapped BLAST and PSI-BLAST: A new generation of protein database search programs*, Nucleic acids research 25, pp 3389-3402, 1997.

Heuristical approaches for database search

The FASTA approach

- FASTA
  - Origin: Computational biology
  - Short for 'fast all'
  - Referenece to FASTP
    - Only applicable to alignment of Proteine sequences

Heuristical approaches for database search
The FASTA approach

- FASTA – Operation principle
  - Pattern iteratively compared with all sequences stored in the data base
  - Comparison with the pattern is obtained in four steps
    - Search for 'Hot-spots'
    - Combine 'Hot-spots' and find sub-alignment
    - Consider sub-alignments that exceed threshold
    - Calculate alternative alignment

Heuristical approaches for database search

The FASTA approach

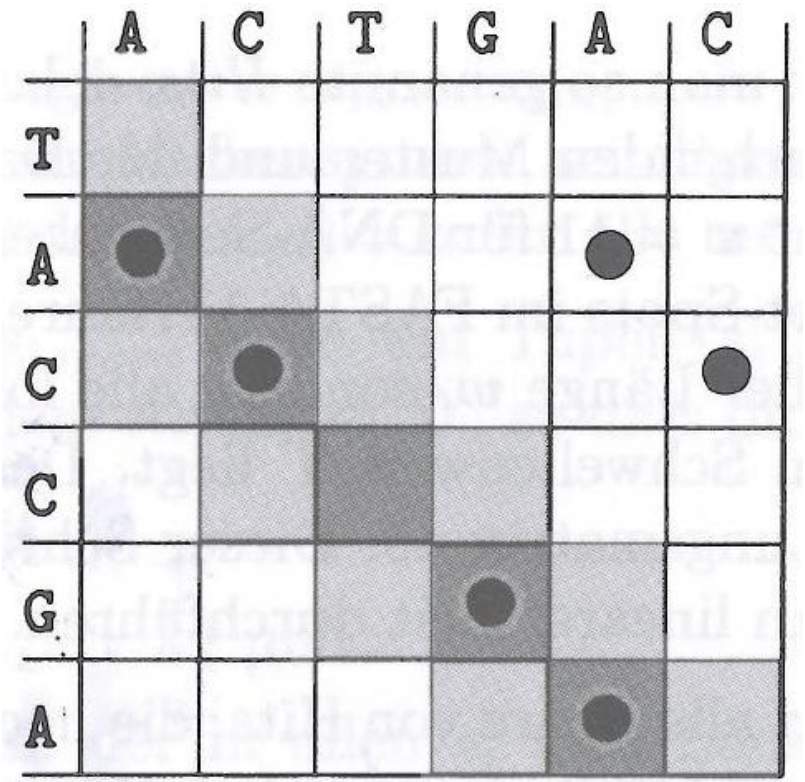
- FASTA – Operation principle
  - Step 1 Search for 'Hot spots'
    - Choose parameter k and each exact match of length k (Hot Spots)
    - Identified by starting-positions in both strings
    - Typical values of k: 2, 6
    - Simple pattern matching approaches feasible for these short sequences (e.g. Boyer-Moore)

Heuristical approaches for database search
The FASTA approach

- FASTA – Operation principle
  - Step 2 Combine 'Hot-spots' and find sub-alignments
    - Create 'Hot-spot' matrix
    - Idetify diagonal sub-sequences that contain 'Hot-spots' (sequence starts and ends in 'Hot spot')
    - Calculate local alignments for these sub-seqeunces
    - Optimum local alignment utilised in step 4

Heuristical approaches for database search

The FASTA approach



Heuristical approaches for database search

The FASTA approach

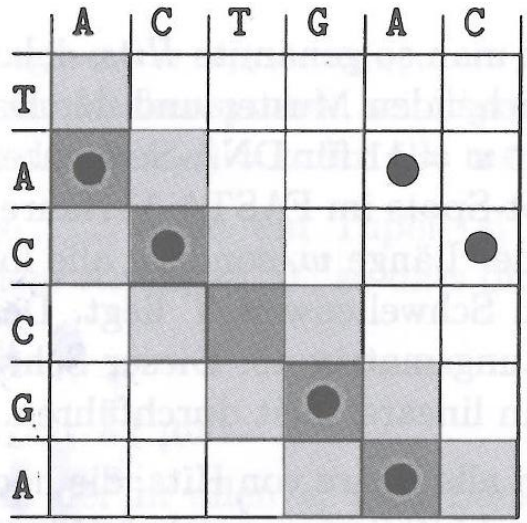
- FASTA – Operation principle

  Step 3 Consider sub-alignments that exceed threshold

  1. Try to enlarge sub-alignments that exceed predefined threshold
  2. Represent sub-alignments as nodes in a graph
     - Sub-alignment u ends at (i, j)
     - Sub-alignment v starts at (i', j')
     - Insert directed edge iff $i < i'$ and $j < j'$
     - (Concatenation of alignments principally possible)
     - (Negative) weight of the edge depends on distance between (i, j) and (i', j')
     - Alignment-cost: Sum of weights along 'alignment-path'
  3. Algorithm outputs this alignment as one possible solution

Heuristical approaches for database search

The FASTA approach



- FASTA – Operation principle
  - Step 4 Calculate alternative alignment
    - Calculate optimal local alignment based on the exact approach
    - Search for optimal local alignment restricted to small corridor around the optimum alignment found in step 2

Heuristical approaches for database search

The BLAST approach

- BLAST (Basic Local Alignment Search Tool)
  - Various implementations and specialised approaches
  - Search in DNA- or proteine databases
  - Two components:
    - Search component
    - Estimation of statistic relevance

Heuristical approaches for database search

The BLAST approach

- BLAST (Basic Local Alignment Search Tool)

  Search component :

  - Search for 'hits'
  - Search for pairs of 'hits'
  - Calculate extensions of 'hits'

Heuristical approaches for database search

The BLAST approach

Step 1 Search for 'hits'

- Search for local alignments without gaps that exceed a given cost threshold (Hits)
- FASTA-approach: Exact matchings (Hot-spots)
- Calculation efficiently possible

Heuristical approaches for database search

The BLAST approach

Step 2 Search for pairs of 'hits'

- Search for pairs of 'hits' with a maximum distance of d
- All other 'hits' are not considered further

Heuristical approaches for database search

The BLAST approach

Step 3 Calculate extensions of 'hits'

- 'Hit'-pairs are extended at their ends until the alignment score does not further increase
- Extended 'Hit'-pairs that exceed a given threshold S are considered 'high scoring pairs' (HSP)
- Algorithms outputs ordered list of HSPs

Heuristical approaches for database search

The BLAST approach

- BLAST (Basic Local Alignment Search Tool)
  Estimation of statistic relevance :
  - Similar to FASTA approach

Outline

Alignment prediction approaches

1. Introduction to alignment methods
   - Alignment of two strings
   - Heuristical approaches for database search
   - Multiple alignments
2. Prediction with alignment methods
   - Application scenarios
3. Properties of the alignment prediction approach

Multiple alignments
Introduction

- Calculation of alignments between more than two strings
- Various approaches in the literature
- Algorithmic solution much harder than alignment between two strings

Multiple alignments

Definition and scoring of multiple alignments

Multiple alignment

Given k strings over an alphabet Σ:

$s_1 = s_{11} \dots s_{1m_1}, \dots, s_k = s_{k1} \dots s_{km_k}$

A multiple alignment of $s_1, \dots, s_k$ is a tupel $(s'_1, \dots, s'_k)$ of strings of length $l \geq \max\{m_i | 1 \leq i \leq k\}$ with

- $|s'_1| = |s'_2| = \dots = |s'_k|$
- $h(s'_i) = s_i$ for all $i \in \{1, \dots, k\}$
- For all $j \in \{1, \dots, l\}$ exists one $i \in \{1, \dots, k\}$ with $s'_{i,j} \neq -$

Multiple alignments

Definition and scoring of multiple alignments

- Problem: How to calculate alignment score?
  - Various solutions feasible to calculate alignment score
  - Alignment score determined by 'Consensus'
  - Alignment score defined by score of pairwise alignment scores

Multiple alignments

Definition and scoring of multiple alignments

- Alignment score determined by 'Consensus'
  - Choose for every column of all alignments the symbol that occurs most often
  - The concatenation of these most common symbols is the 'Consensus'
  - Alignment score determined by distance to the 'Consensus'

Multiple alignments

Definition and scoring of multiple alignments

- Alignment score determined by 'Consensus'

Consensus

Let $(s'_1, \ldots, s'_k)$ be a multiple alignment of length $l = |s'_1|$. A string $c = c_1 \ldots c_l \in \Sigma^l$ is called Consensus for $(s'_1, \ldots, s'_k)$, if

$$c_j = argmax_{a \in \Sigma} |\{s'_{ij} = a | 1 \leq i \leq k\}| \text{ for all } 1 \leq j \leq l \quad (9)$$

- This method to determine a Consensus is called 'Majority voting'.
- In the literature, also other approaches to obtain a consensus are discussed.

Multiple alignments

Definition and scoring of multiple alignments

- Alignment score determined by 'Consensus'

Distance to Consensus

The distance of an alignment $(s'_1, \ldots, s'_k)$ of length $l = |s'_1|$ to a Consensus c is defined as

$$dist(c, (s'_1, \ldots, s'_k)) = \sum_{j=1}^{i} |\{s'_{ij} | 1 \leq i \leq k, s'_{ij} \neq c_j\}| \qquad (10)$$

Multiple alignments

Definition and scoring of multiple alignments

- Alignment score determined by 'Consensus'

Lemma: Distance to consensus strings

Let $(s'_1, \ldots, s'_k)$ be a multiple alignment with two Consensus strings c and $\overline{c}$. Then, the following equation holds

$$dist(c, (s'_1, \ldots, s'_k)) = dist(\overline{c}, (s'_1, \ldots, s'_k)) \quad (11)$$

Multiple alignments

Definition and scoring of multiple alignments

- Example

$$
\begin{array}{rcl}
s_1' & = & AATGCT \\
s_2' & = & A\_\ TTC\_ \\
s_3' & = & \_\ \_\ \_TCC \\
\hline
c & = & AATTCT
\end{array}
$$

- $dist(c, (s_1', s_2', s_3')) = 1 + 2 + 1 + 1 + 0 + 2 = 7$

Multiple alignments

Definition and scoring of multiple alignments

Mult-Consensus-Align-Problem

Input A set $S = \{s_1, \ldots, s_k\}$ of strings over an alphabet Σ

Valid solutions All multiple alignments of S

Costs The cost of a multiple alignment $(s'_1, \ldots, s'_k)$ with a Consensus c is

$$cost((s'_1, \ldots, s'_k)) = dist(c, (s'_1, \ldots, s'_k)) \quad (12)$$

Optimisation aim Minimisation

Multiple alignments

Definition and scoring of multiple alignments

- Alignment score based on pairwise alignment scores

Bewertung δ_{SP}

Let Σ be an alphabet, $- \notin \Sigma$ a gap symbol and δ a scoring function for the alignment of two strings that is also valid for $\delta(-,-)$. The score of a multiple alignment $(s'_1, \ldots, s'_k)$ of length l is given by

$$\delta_{SP}(s'_1, \ldots, s'_k) = \sum_{j=1}^{l} \underbrace{\sum_{i=1}^{k} \sum_{r=i+1}^{k}}_{\text{Column sum}} \delta(s'_{ij}, s'_{rj}) \qquad (13)$$

Multiple alignments

Definition and scoring of multiple alignments

- Example
  - $\delta(x, y) = \begin{cases} 0 & x = y \\ 1 & x \neq y \end{cases}$
  - Alignment calculated in the example above:
  - $s'_1 = AATGCT$
  - $s'_2 = A\_\ TTC\_$
  - $s'_3 = \_\ \_\ \_TCC$

Multiple alignments

Definition and scoring of multiple alignments

- Example
  - $\delta(x,y) = \begin{cases} 0 & x = y \\ 1 & x \neq y \end{cases}$
  - Alignment calculated in the example above:
  - $s'_1 = AATGCT$
  - $s'_2 = A\_\ TTC\_$
  - $s'_3 = \_\ \_\ \_TCC$

$$\begin{aligned} \delta_{SP}(s'_1, s'_2, s'_3) &= \sum_{j=1}^{6} \sum_{i=1}^{3} \sum_{r=i+1}^{3} \delta(s'_{ij}, s'_{rj}) \\ &= \sum_{j=1}^{6} \left(\delta(s'_{1j}, s'_{2j}) + \delta(s'_{1j}, s'_{3j}) + \delta(s'_{2j}, s'_{3j}) \right) \\ &= 11 \end{aligned}$$

Multiple alignments

Definition and scoring of multiple alignments

Mult-SP-Align-Problem

Calculation of a multiple alignment with optimum SP-score

Input A set $S = \{s_1, \ldots, s_k\}$ of strings over an alphabet Σ

Valid solutions All multiple alignments of S

Costs The cost of a multiple alignment $(s'_1, \ldots, s'_k)$ is

$$cost((s'_1, \ldots, s'_k)) = \delta_{SP}(s'_1, \ldots, s'_k) \qquad (14)$$

Optimisation aim Minimisation

Multiple alignments

Exact calculation of multiple alignments

- Exact calculation possible in analogy to alignment of two strings
  - For k strings $s_1, \ldots, s_k$ we denote a k dimensional array.
  - Entry $A(i_1, \ldots, i_k)$ holds the score of an optimal multiple alignment of the prefixes $s_{11} \ldots s_{1i_1}, \ldots, s_{k1} \ldots s_{ki_k}$
- Problem:
  - Computational complexity and size of the array exponential in k
- This problem is NP-hard when k is also part of the input.[6]

[6] L. Wang and T. Jiang, *On the complexity of multiple sequence alignment*, Journal of computational biology 1 (4), pp 337-348, 1994.

Multiple alignments

Merging pairwise alignments

- Since exact calculation of multiple alignments not feasible in practical situations we want to calculate an approximative multiple alignment
- Idea:
  - Construct multiple alignment from a set of pairwise alignments of strings

Multiple alignments

Merging pairwise alignments

Compatibility of multiple alignments

Let $S = \{s_1, \ldots, s_k\}$ be a set of strings and $T = \{s_{i_1}, \ldots, s_{i_m}\}$ be a subset of S. Let $A' = (s'_1, \ldots, s'_k)$ be a multiple alignment of S and $A'' = (s''_1, \ldots, s''_m)$ a multiple alignment of T.
The alignment A' is compatible to A'' if A' is identical to A'' in $i_1, \ldots, i_m$ after removing all gaps.

Multiple alignments

Merging pairwise alignments

- Example
  - $S = \{ACGG, ATG, ATCGG\}$
  - $T_1 = \{ACGG, ATG\}$
  - $T_2 = \{ATG, ATCGG\}$

Multiple alignments

Merging pairwise alignments

- The alignment

$$\begin{array}{l} A\_\ CGG \\ A\_\ \_\ TG \\ ATCGG \end{array}$$

of S is compatible to the alignment

$$\begin{array}{l} ACGG \\ A\_\ TG \end{array}$$

of T_1 since the restriction of S to the first and second row (and deletion of gap column) leads to

$$\begin{array}{l} A\_\ CGG \\ A\_\ \_\ TG \end{array} \rightarrow \begin{array}{l} ACGG \\ A\_\ TG \end{array}$$

Multiple alignments

Merging pairwise alignments

- The alignment

$$A\_\ CGG$$
$$A\_\ \_\ TG$$
$$ATCGG$$

of S is not compatible to the alignment

$$AT\_\ G\_$$
$$ATCGG$$

of T_2 since the restriction of S to the second and third row leads to

$$A\_\ \_\ TG$$
$$ATCGG$$

which is another alignment.

Multiple alignments

Graph algorithms

- A multiple alignment of strings can be calculated by a graph algorithm[7]

[7] D. Feng and R. Doolitle, *Progressive sequence alignment as a prerequisite to correct phylogenetic trees*, Journal of moledular evolution 25, pp 351-360, 1987.

Multiple alignments

Graph algorithms

Alignment Tree

Let $S = \{s_1, \ldots, s_k\}$ be a set of Strings over Σ. A Tree $T = (V, E)$ with $V = \{s_1, \ldots, s_k\}$ and edges $\{s_i, s_j\} \in E$ that are labelled with an optimal alignment (s'_i, s'_j) is called alignment tree for S

Multiple alignments

Graph algorithms

Multiple alignmnets from alignment trees

Given an alignment tree $T = (V, E)$ for a set of strings S, a multiple alignment $(s''_1, \ldots, s''_k)$ that is compatible to the optimal pairwise alignment $(s'_1, \ldots, s'_k)$ can be derived efficiently.

- This assertion was proven by D. Feng and R. Doolittle[8]
- We will introduce an algorithm that calculates multiple alignments from graphs with a star topology.

[8] D. Feng and R. Doolittle, *Progressive sequence alignment as a prerequisite to correct phylogenetic trees*. Journal of Molecular Evolution, 25, pp 351-360, 1987.

Multiple alignments

Graph algorithms

Star alignment

Let T be an alignment tree $T = (V, E)$ for a set of strings $S = (s_1, \ldots, s_k)$. T is a star, when it is composed of a center c and $k - 1$ leaves that are separated from the center with one edge each. This special case is often referred to as star alignment

- The algorithms for the star alignment will first choose one string from the set S to be the center.
- Afterwards, a compatible multiple alignment is created from the optimal pairwise alignments of the remaining strings with the center string.

Multiple alignments

Graph algorithms

- Example: Iteratively compose a compatible alignment
  - The algorithm follows the principle to minimise the count of gap symbols included.
  - $c' = ATG\_\ CATT$
  - $s_1' = A\_\ GTCAAT$
  - $s_2' = \_\ TCTCA\_\_$
  - and
  - $c'' = A\_\ TGCAAT$
  - $s_3'' = ACTGTAAT$
  - The insertion of gap symbols of both strings yields
    - $c''' = A\_\ TG\_\ C\_\ ATT$

Multiple alignments

Graph algorithms

- Example: Iteratively compose a compatible alignment
  - The aligment of all four strings is therfore
  - $c''' = A\_\ TG\_\ C\_\ ATT$
  - $s_1''' = A\_\_\ GTC\_\ AAT$
  - $s_2''' = \_\_\ TCTC\_\ A\_\_$
  - $s_3''' = ACTG\_\ TAAT$
- However, this method does not always yield the optimum alignment since $s_3''' = ACTGT\_\ AAT$ would have been a better solution.

Multiple alignments

Graph algorithms

Algorithm star alignment

```
Input:  A set of strings S = {s_1, ..., s_k}
Output:  A compatible alignment of S on T
1 for i = 1 to k do                                  Calculate star center
2     for j = i to k do
3         Align(s_i, s_j)                            Calculate optimum pairwise alignment
4 c = argmin Σ_{s∈S} sim(t, s)                       Find center of star
5 for i = 2 to k do                                  Determine compatible multiple alignment
6     Calculate compatible multiple alignment
```

Outline

Alignment prediction approaches

1. Introduction to alignment methods
   - Alignment of two strings
   - Heuristical approaches for database search
   - Multiple alignments
2. Prediction with alignment methods
   - Application scenarios
3. Properties of the alignment prediction approach

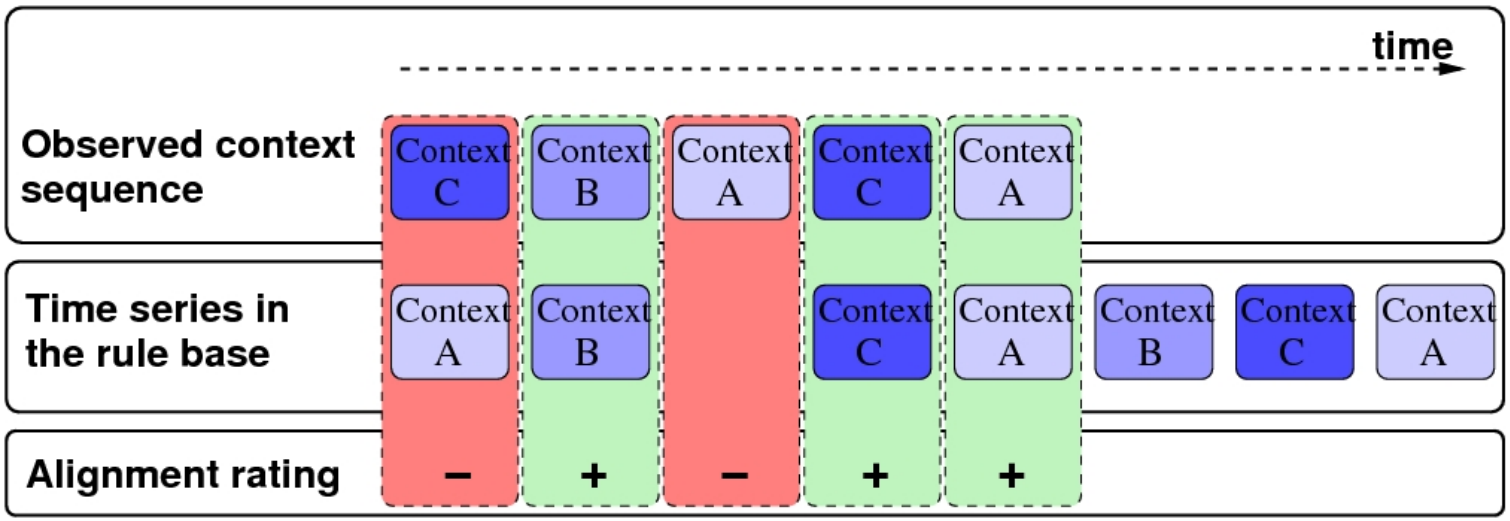
Prediction with alignment methods

Prediction procedure

- Utilise alignment methods for context prediction purposes
  - Compare the end of the observed sequence with typical context patterns
  - Semiglobal alignment between observed and typical patterns
  - Method stores and computes a set of typical context patterns
- Idea
  - When observed pattern is very similar to sub-string in a typical context pattern, we deduce that the continuation of both patterns is also very similar

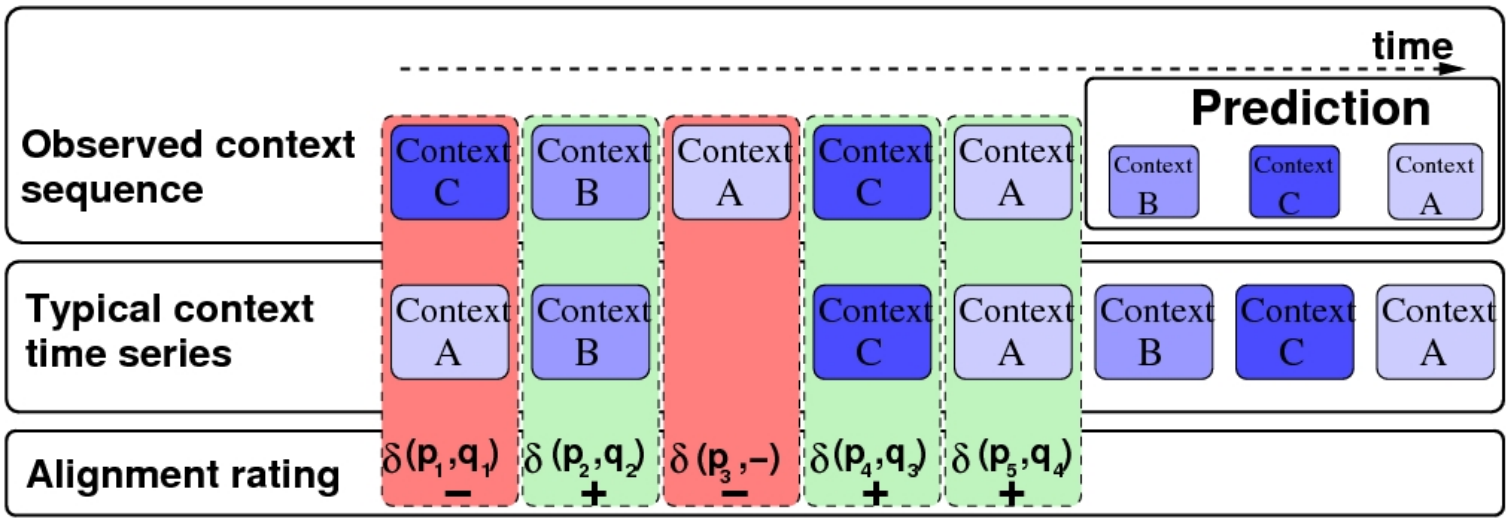
Prediction with alignment methods

Prediction procedure



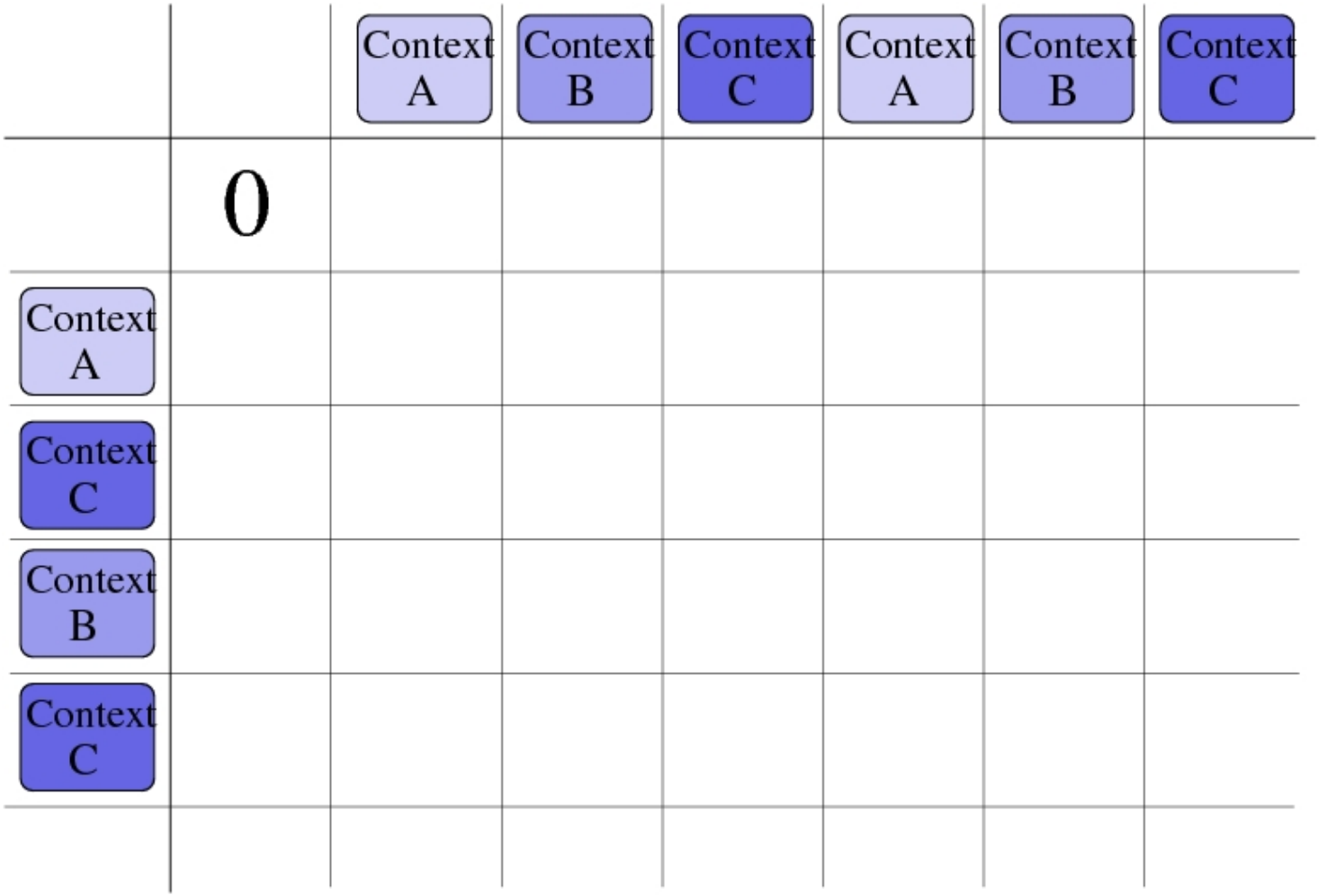
Prediction with alignment methods

Prediction procedure



Prediction with alignment methods

Example



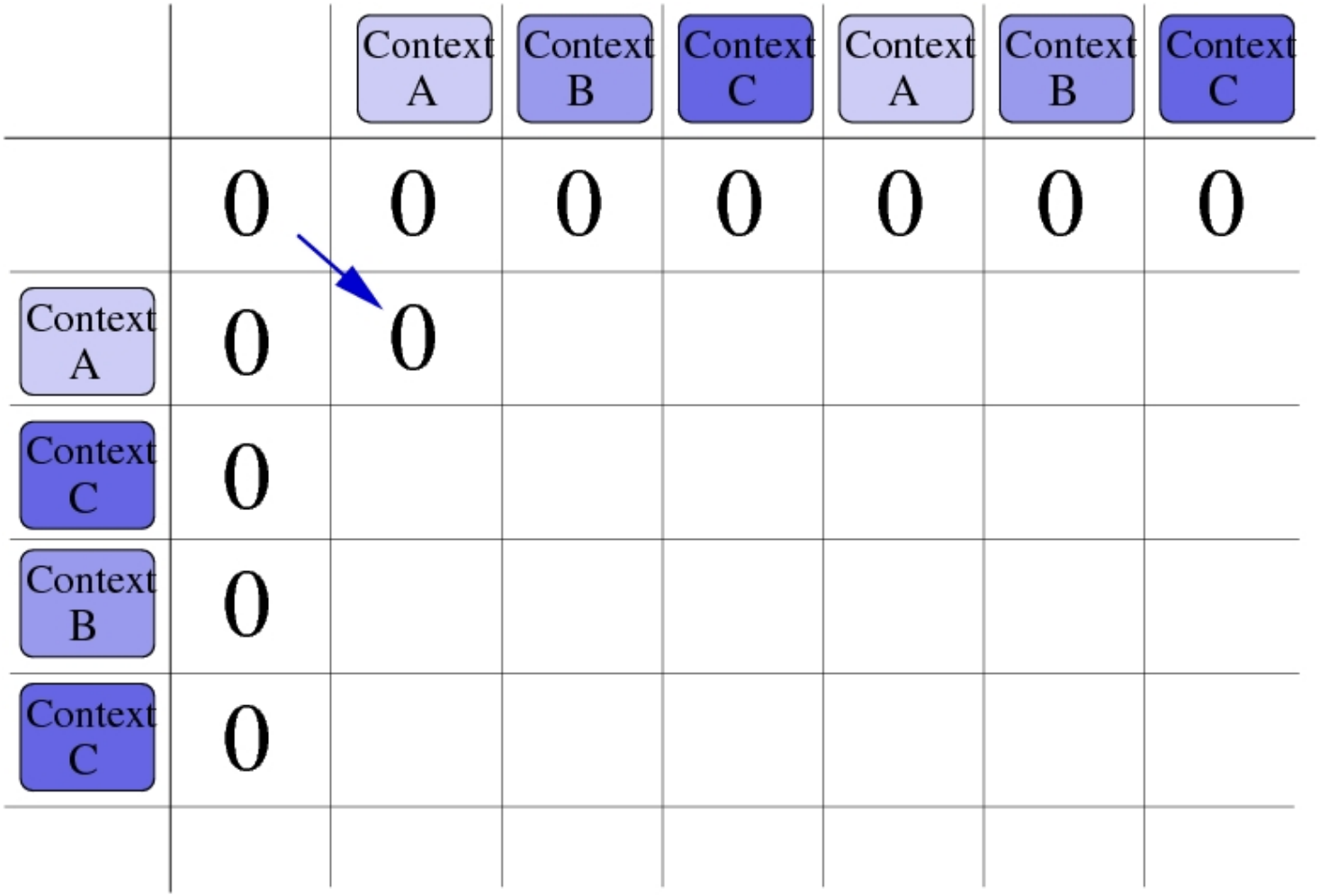
Prediction with alignment methods

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						

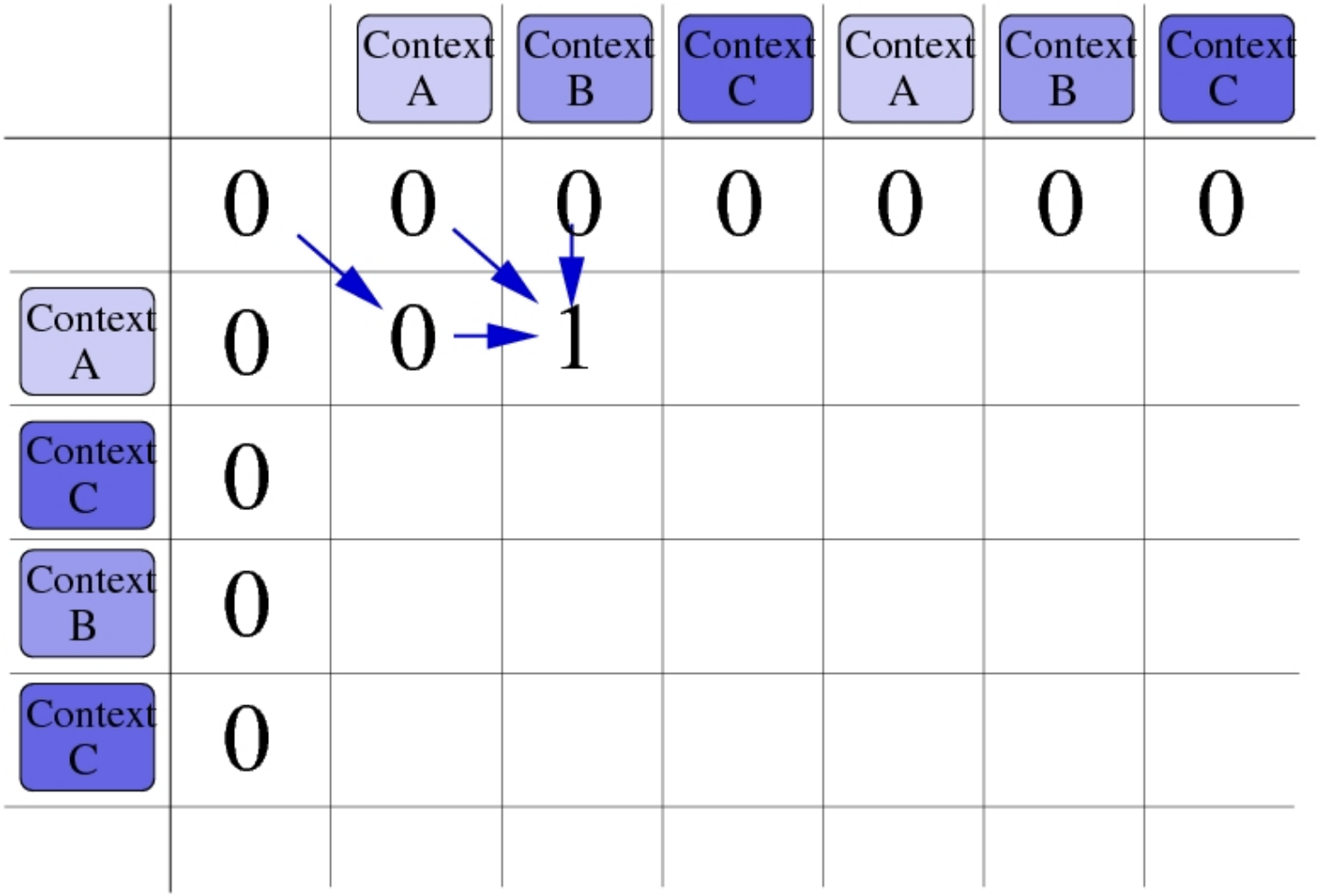
Prediction with alignment methods

Example



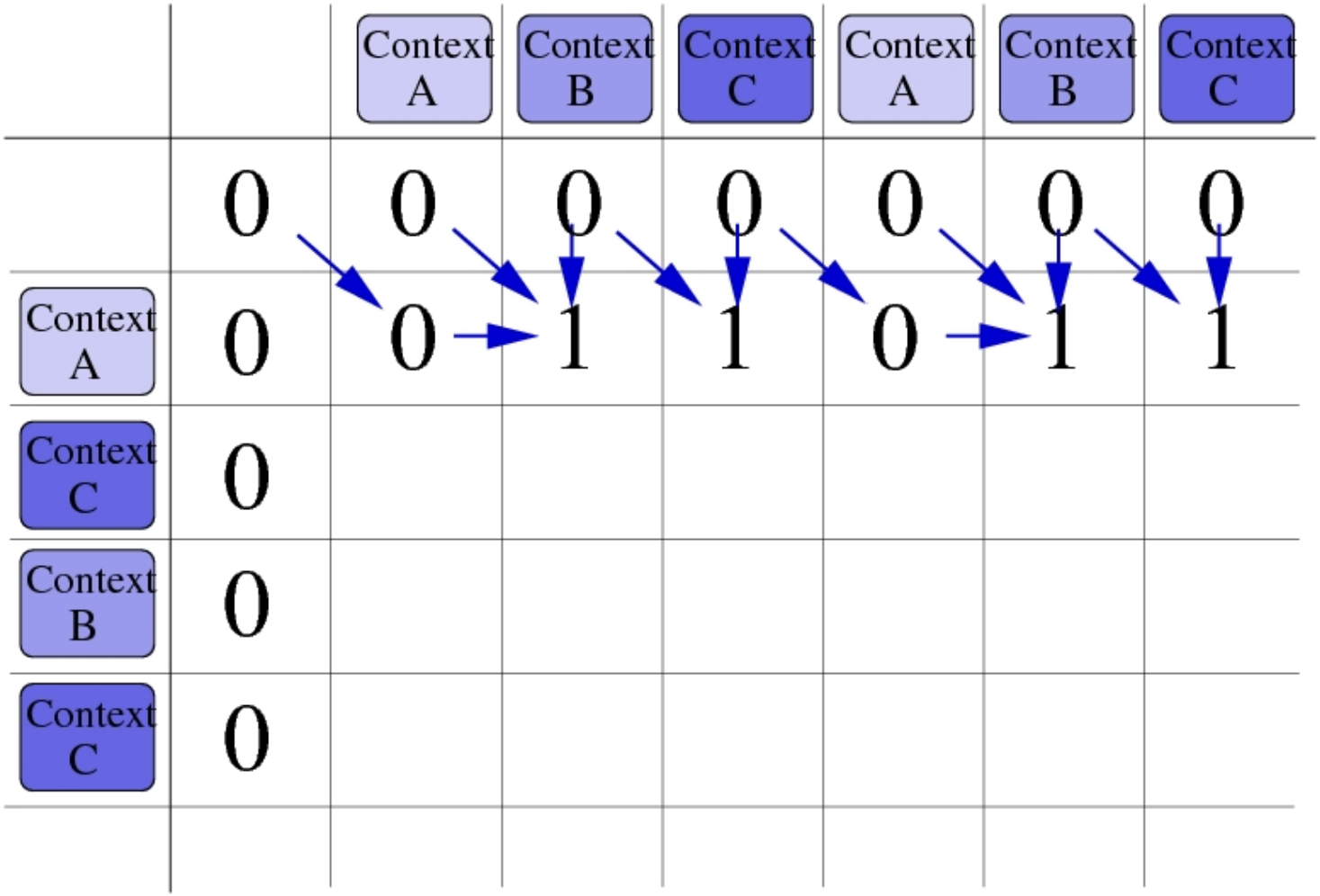
Prediction with alignment methods

Example



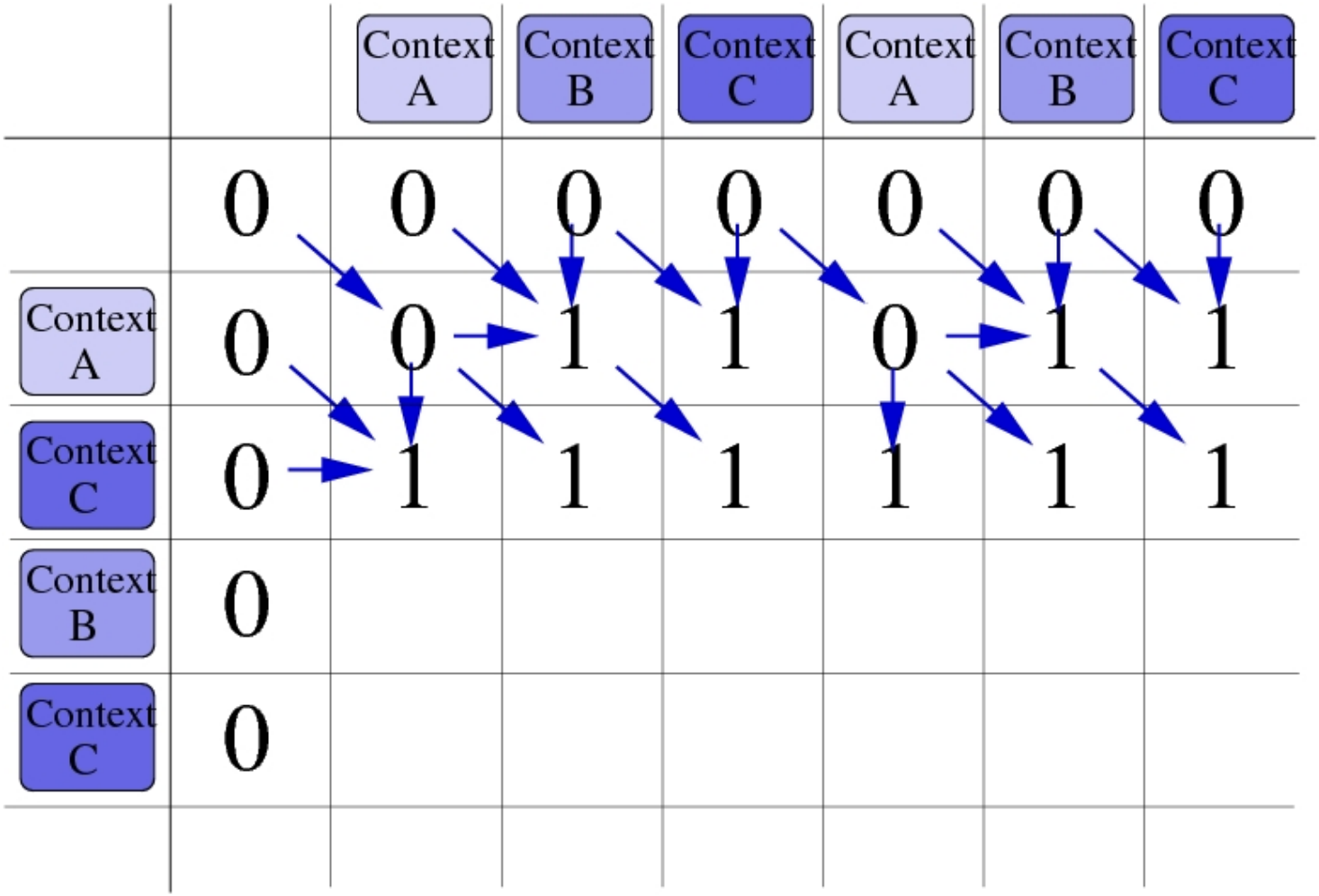
Prediction with alignment methods

Example



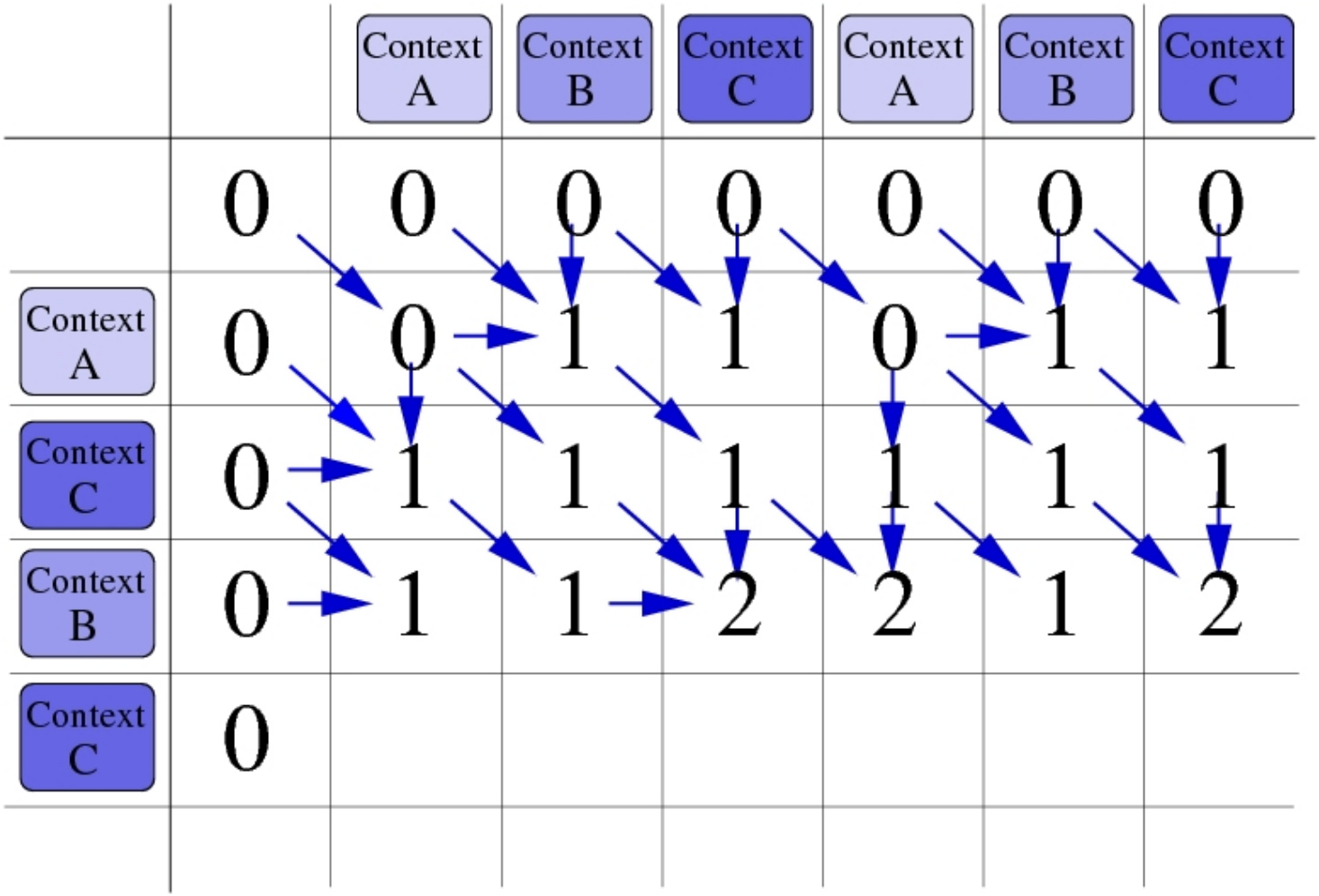
Prediction with alignment methods

Example



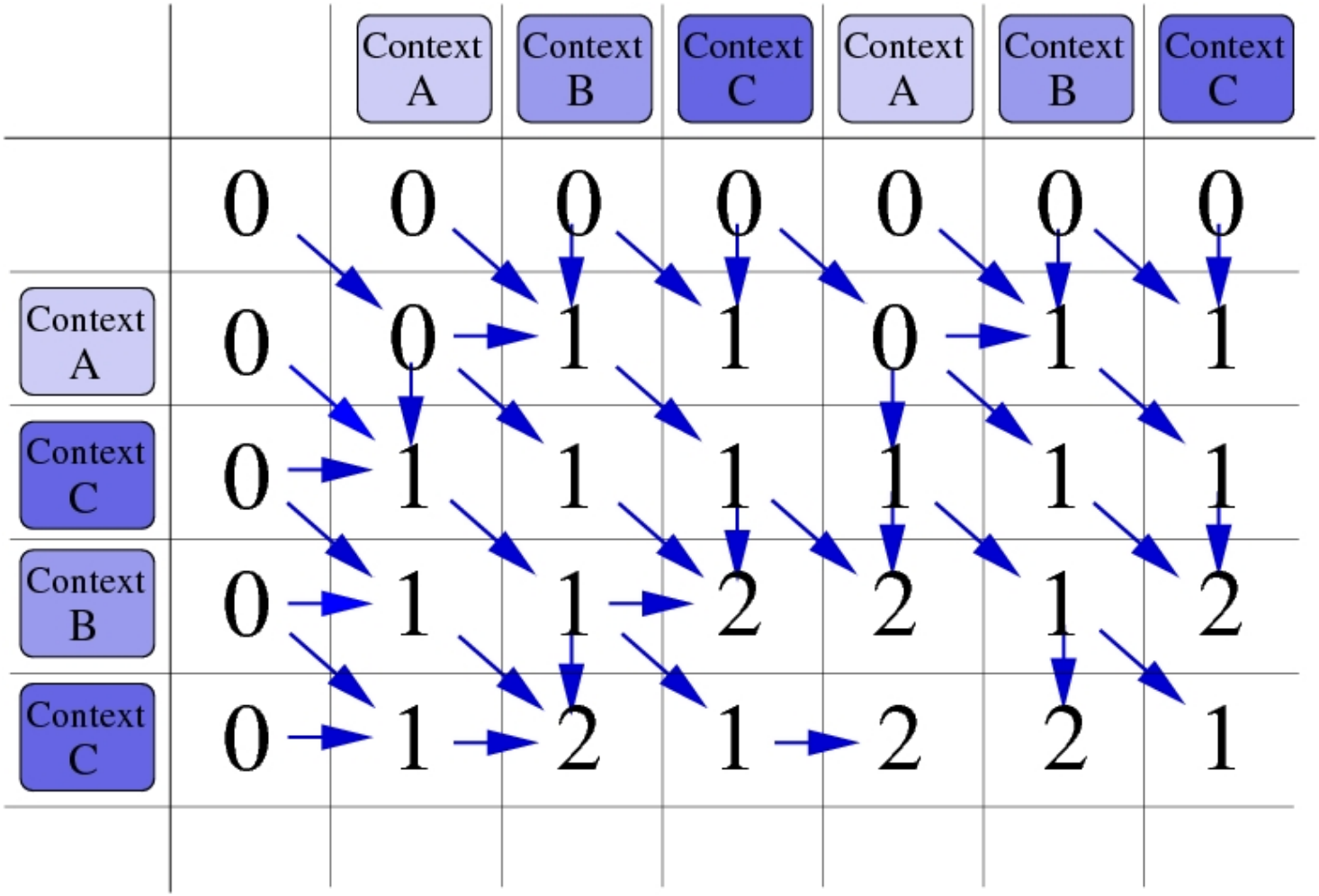
Prediction with alignment methods

Example



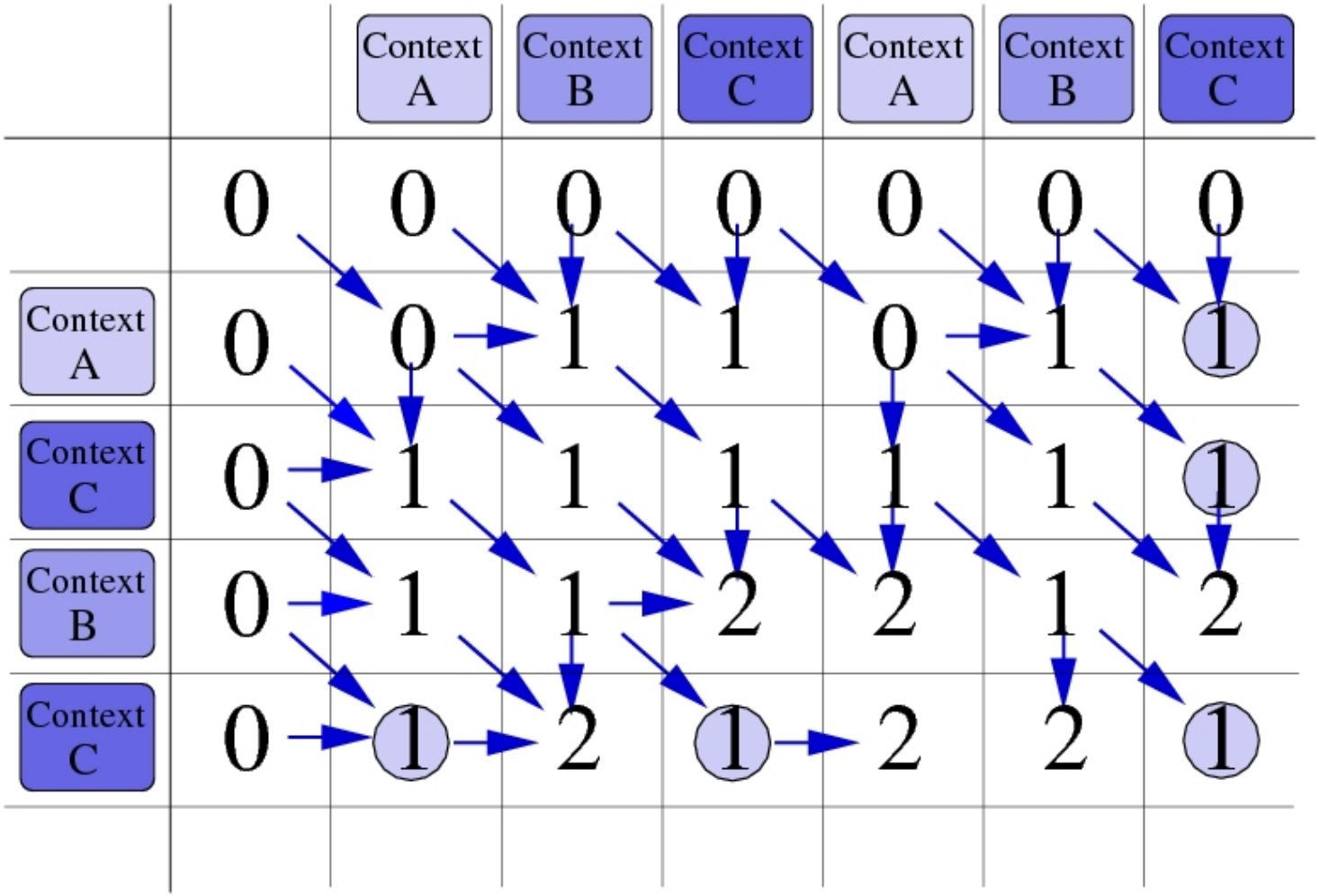
Prediction with alignment methods

Example



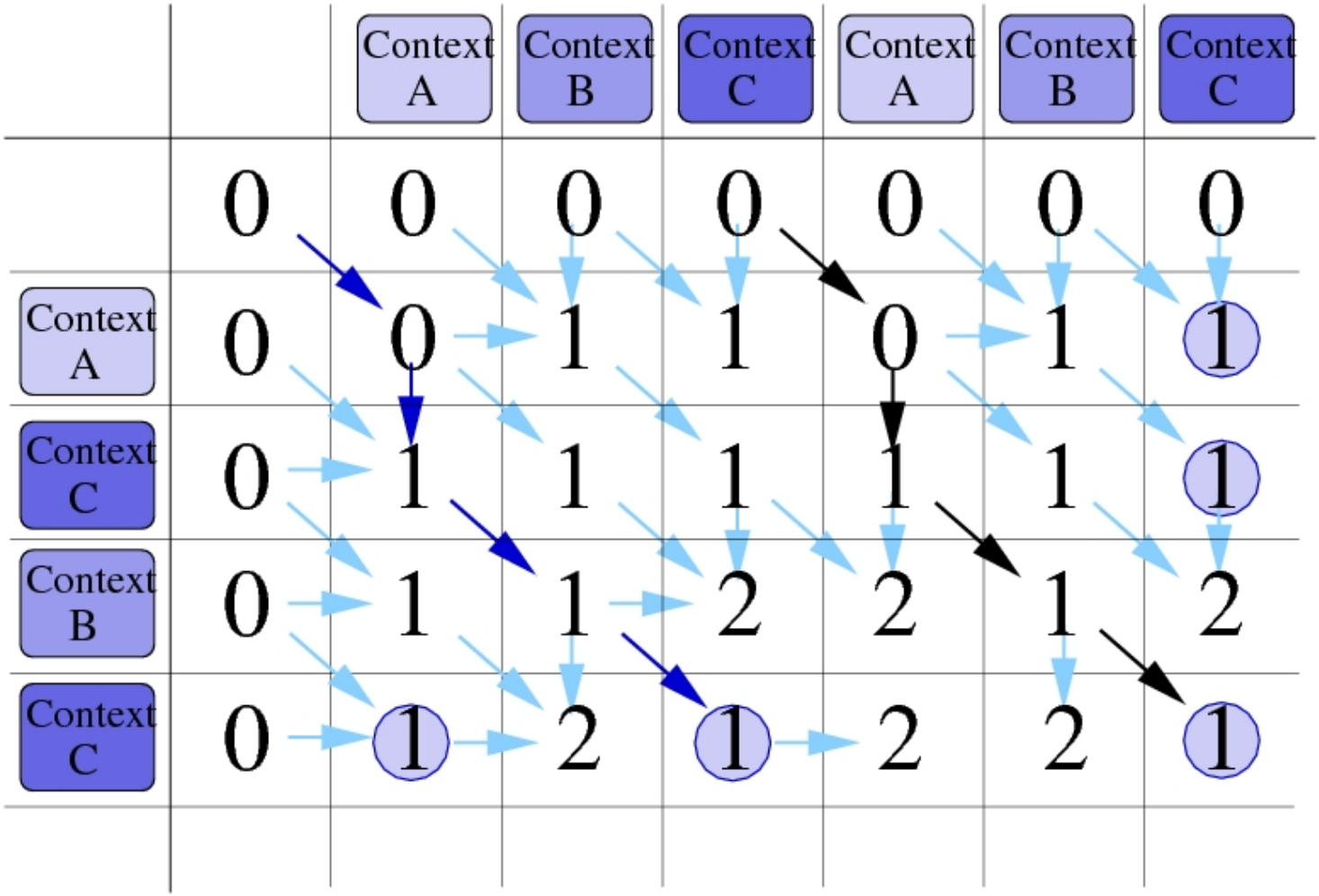
Prediction with alignment methods

Example



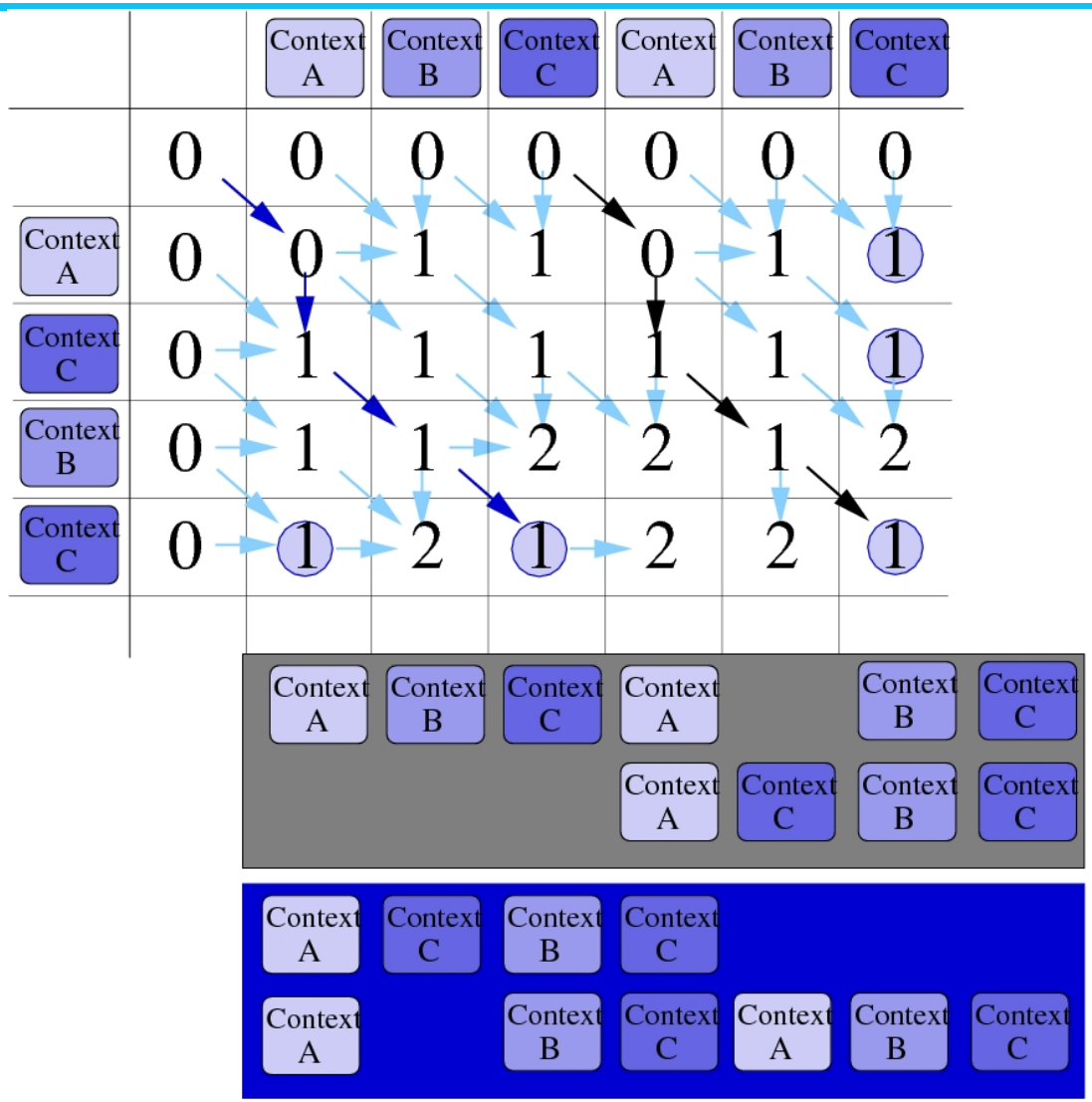
Prediction with alignment methods

Example



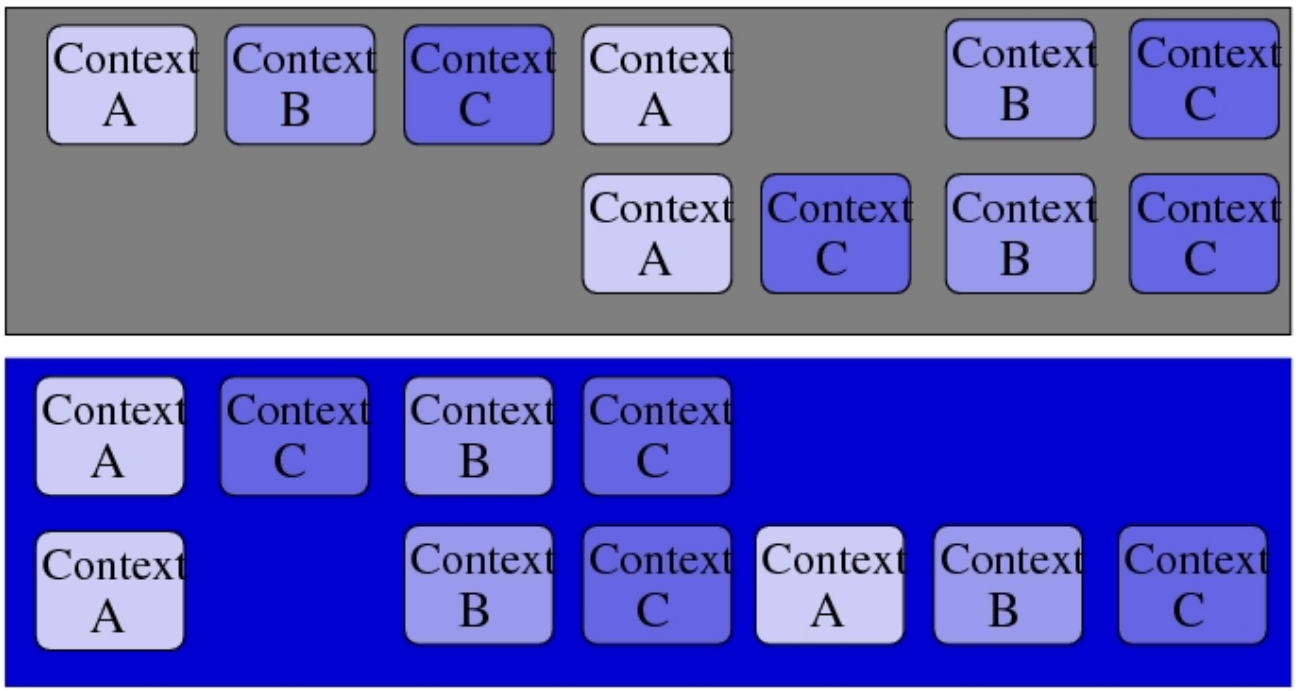
Prediction with alignment methods

Example



Prediction with alignment methods

Example



Prediction with alignment methods

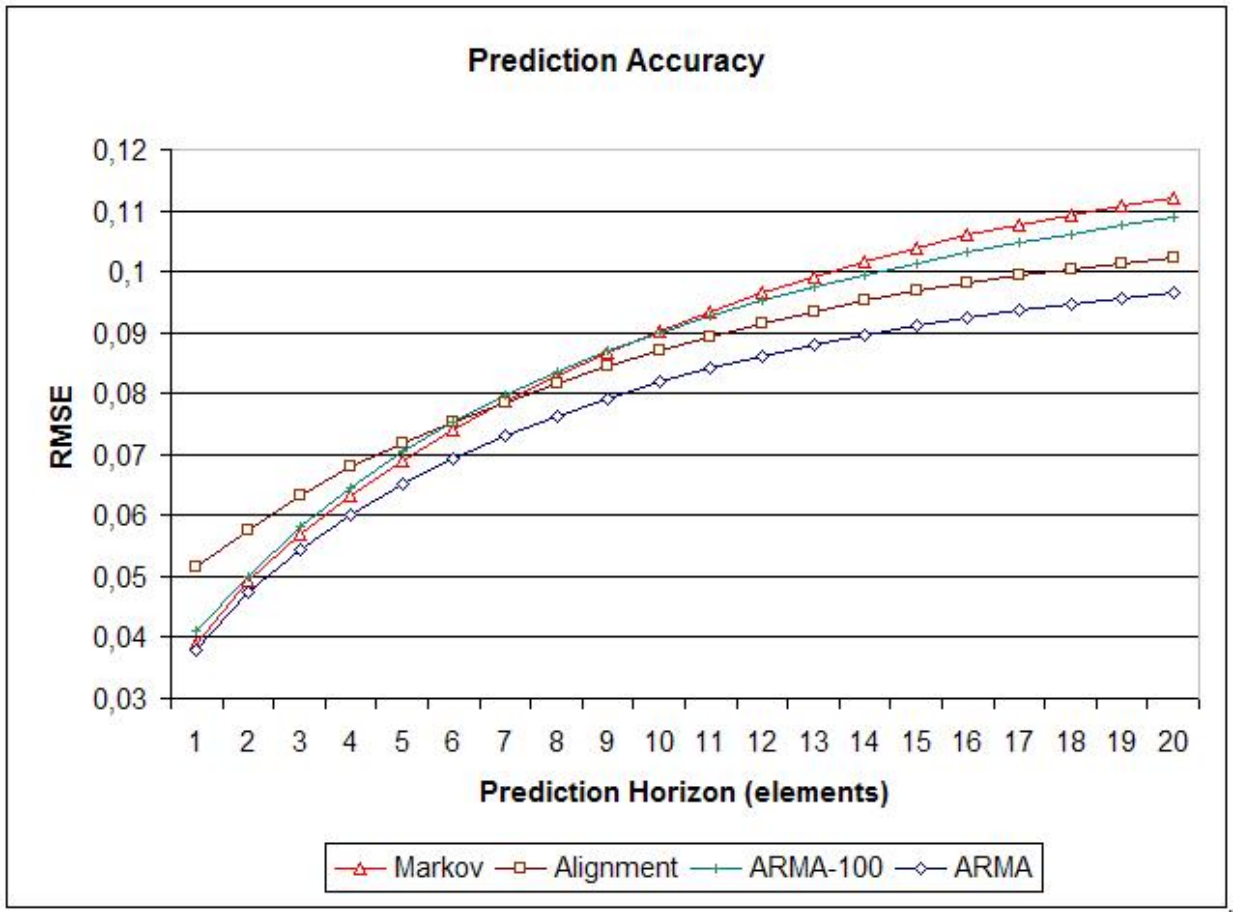
Application scenarios

- Wind power prediction
  - Wind power samples from wind farms in Germany
  - February 2004 to April 2005
  - taken in an hourly fashion

Prediction with alignment methods

Application scenarios

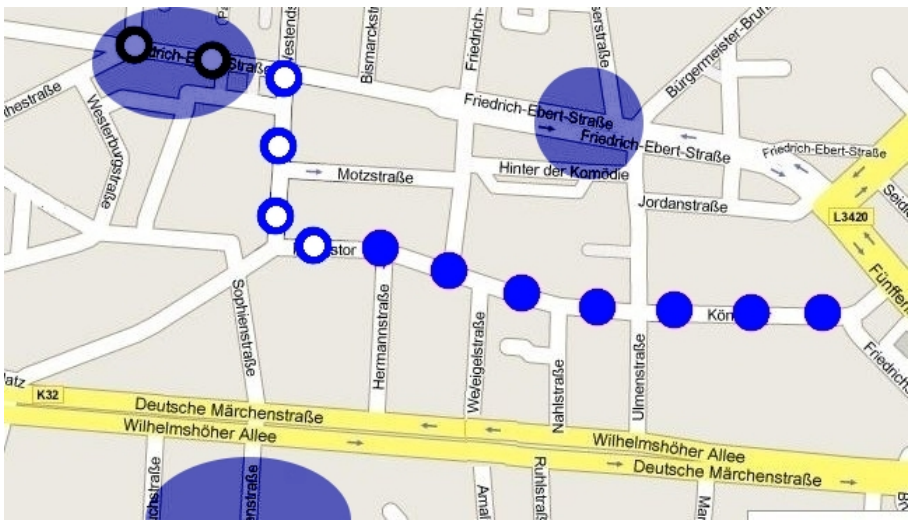
- Wind power prediction



Prediction with alignment methods

Application scenarios

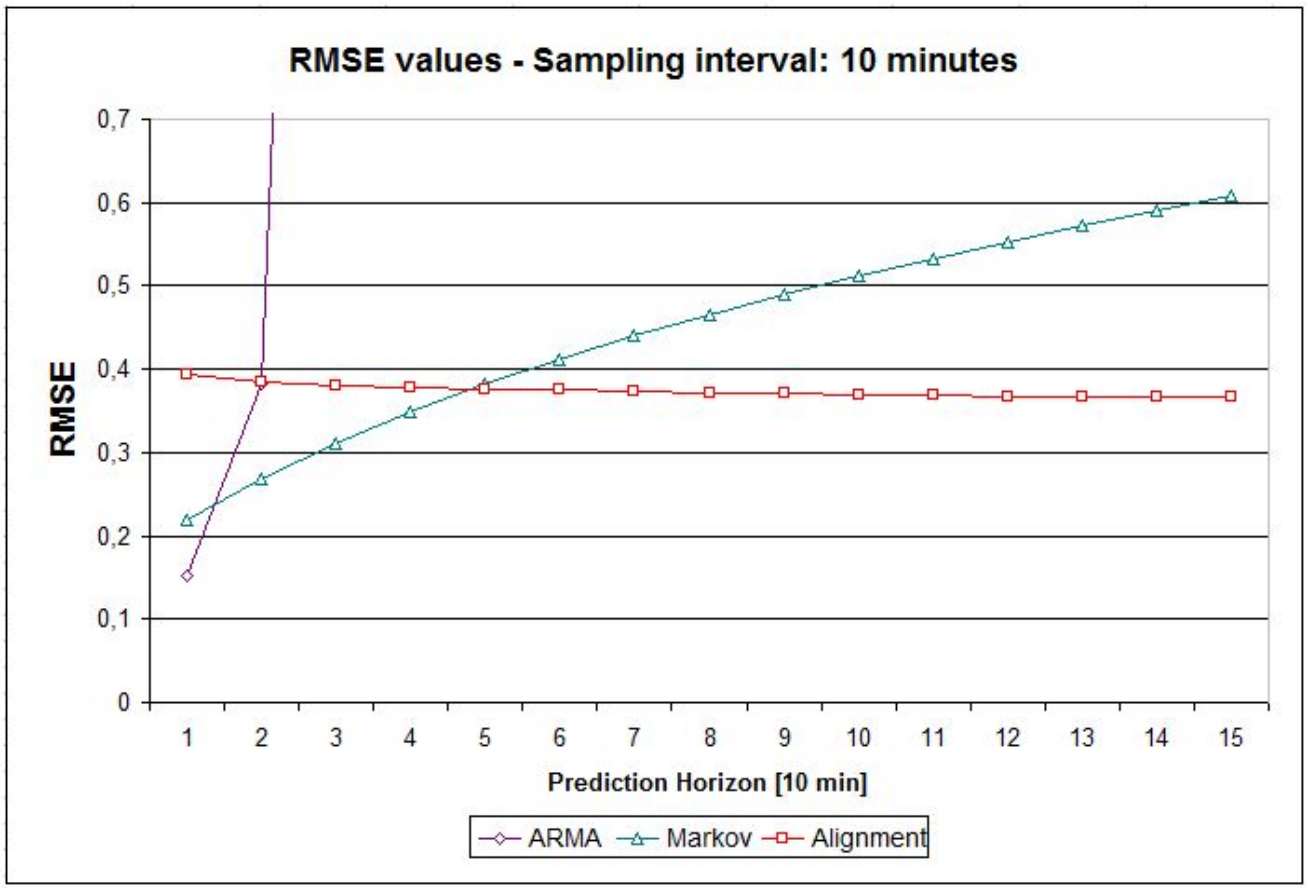
- GPS location prediction
  - GPS locations of a mobile user
  - 21 days
  - Sampling interval 2 minutes
  - Contexts: GPS measurements



Prediction with alignment methods

Application scenarios

- GPS location prediction



Outline

Prediction with alignment approaches

1. Introduction to alignment methods
   - Alignment of two strings
   - Heuristical approaches for database search
   - Multiple alignments
2. Prediction with alignment methods
   - Application scenarios
3. Properties of the alignment prediction approach

Properties of the alignment prediction approach

Processing load

- Runtime for computing a prediction: $(O(I \cdot k^2))$
  - Exact alignment calculation: $O(k^2)$
  - For each of I typical sequences

Properties of the alignment prediction approach

Memory requirements

- Memory requirements
  - $O(k \cdot l)$
  - Length of typical sequences: k
  - Number of typical sequences stored: l

Properties of the alignment prediction approach

Prediction horizon

- Prediction horizon dependent on typical sequence length

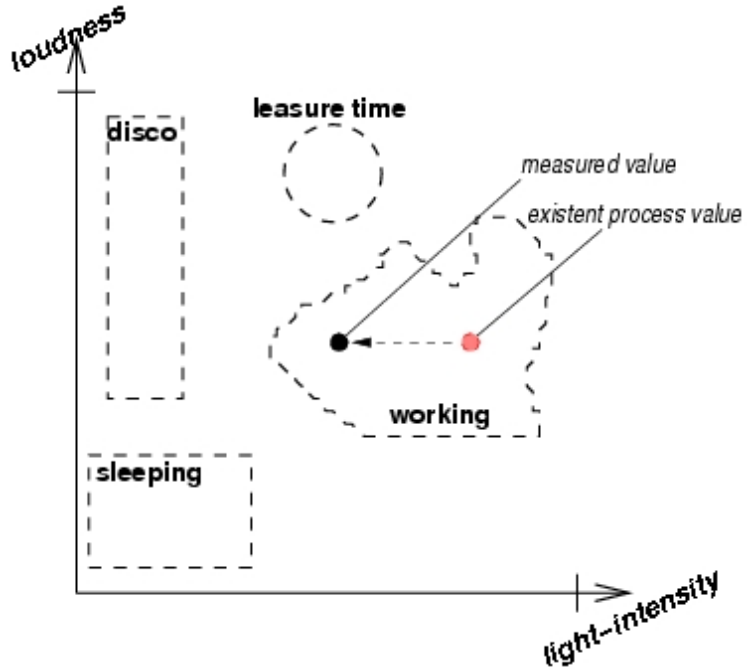
Properties of the alignment prediction approach

Adaptability

- The alignment prediction approach is able to adapt to changing environments
  - When typical behaviour patterns change, new typical patterns are observed and stored
  - Possible: Weighting of typical patterns according to relevance

Properties of the alignment prediction approach

Multi-dimensional time series



- The alignment prediction algorithm is well suited for multi-dimensional time series
  - Distance between observed contexts at a time dependent on alignment metric (scoring function)
  - Example: Eucledian distance in vector space

Properties of the alignment prediction approach

Iterative prediction

- Iterative Prediction (technically) possible
  - Utilise end of typical context sequence as input of the prediction approach
  - However, an extension of a typical context sequence is no longer a typical sequence
  - Otherwise the typical sequence had been longer
- Not feasible for context prediction

Properties of the alignment prediction approach

Prediction of context durations

- Prediction of context duration possible
  - Sampling interval of context sequences defines time interval

Properties of the alignment prediction approach

Approximate matching of patterns

- Approximate matching
  - Dependent on scoring function
  - Cost of gap symbols and mismatches guides the approximate matching

Properties of the alignment prediction approach

Context data types

- All context data types supported
  - Context sequences interpreted as points in coordinate systems
  - Vector spaces defined by context types
  - Multi-type context sequences feasible

Properties of the alignment prediction approach

Pre-processing

- Pre-processing required to identify typical context sequences
- On-line approach feasible
- Runtime: $O(k^2)$
  - Find local alignments in the input sequence
  - Align input sequence to itself
  - Occurrence or the alignment determines its relevance
  - Also possible: Multiple alignments

Aspects of prediction algorithms

Summary

	IPAM	ONISI	Markov	CRF
Numeric Contexts	yes	no	no	no
Non-numeric Contexts	yes	yes	yes	yes
Complexity	$O(k)$	$O(k^2)$	$O(C^2)$	$O(C^2)$
Learning ability	(no)	yes	yes	yes
Approximate matching	no	no	no	no
Multi-dim. TS	(no)	(no)	(no)	(no)
Discrete data	yes	yes	yes	yes
Variable length patterns	no	yes	no	(yes)
Multi-type TS	yes	no	(no)	(no)
Continuous data	no	no	no	no
Pre-processing	$O(k)$	–	$O(k)$	$O(k)$
Context durations	no	no	no	no
Continuous time	no	no	yes	yes

Aspects of prediction algorithms

Summary

	SPM	Align	SOM	PCA
Numeric Contexts	yes	yes		
Non-numeric Contexts	yes	yes		
Complexity	$O(1)$	$O(I \cdot k^2)$		
Learning ability	(yes)	yes		
Approximate matching	no	yes		
Multi-dim. TS	(no)	yes		
Discrete data	yes	yes		
Variable length patterns	yes	yes		
Multi-type TS	no	yes		
Continuous data	no	no		
Pre-processing	$O(k)$	$O(k^2)$		
Context durations	no	yes		
Continuous time	no	no		

Properties of the alignment prediction approach

Conclusion

- The alignment prediction approach is a flexible prediction method
- All context types supported
- Multi-dimensional and multi-type time series feasible
- High computational complexity