Algorithms for context prediction in Ubiquitous Systems

Exact sequence Matching

Stephan Sigg

Institute of Distributed and Ubiquitous Systems
Technische Universität Braunschweig

November 18, 2008

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 approahces
    - 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 approaohces
    - Dempster shafer
    - Evolutionary algorithms
    - Neural networks
    - Simulated annealing

Outline

Simple prediction approaches: ONISI and IPAM

1. Important aspects of context prediction algorithms
2. Exact sequence matching
3. Algorithm: IPAM
4. Algorithm: ONISI

Aspects of prediction algorithms

In Ubiquitous Computing

- Prediction accuracy
- High prediction horizon
- Adaptability
- Memory and processing load
- Multi-dimensional time series
- Iterative prediction
- Prediction of context durations
- Relaxation of typical behaviour patterns
- Context data types
- Pre-processing of time series data

Aspects of prediction algorithms

Prediction accuracy

- Context prediction is an optimisation problem
  - Prediction errors have to be minimised
  - Low error probability desired

Aspects of prediction algorithms

High prediction horizon

- A prediction algorithm shall provide a high prediction horizon
- At the same time: low error probability
- Prediction accuracy decreases with increasing prediction horizon
    - Low degradation speed desired

Aspects of prediction algorithms

Adaptability

- Learning is essential in Ubiquitous Environments
  - Environment is subject to changes
    - typically slow changes
  - Behaviour patterns of persons might change due to external influences
    - Relocation
    - New Job
    - Vacancy
    - New semester and time schedule
- Without learning, prediction accuracy will decrease over time

Aspects of prediction algorithms

Memory and processing load

- Devices for ubiquitous computing typically small scale and mobile
  - Low processing power
  - Restricted memory and storage size

Aspects of prediction algorithms

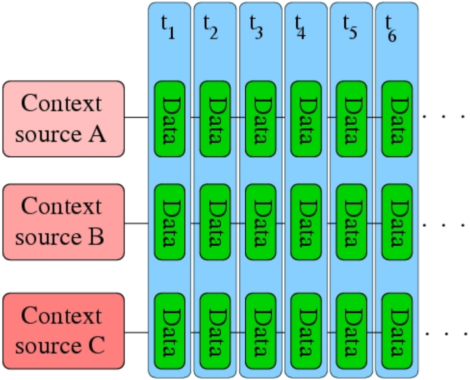
Multi-dimensional time series

- Typically several context sources attached to a device
- The time series observed is therefore multi-dimensional
- Algorithms that are only applicable to one-dimensional input unsuited in many scenarios
  - Solution: Model multi-dimensional TS by several one-dimensional TS
  - Problem: Inter-relation between time series not modelled

Aspects of prediction algorithms

Multi-dimensional time series

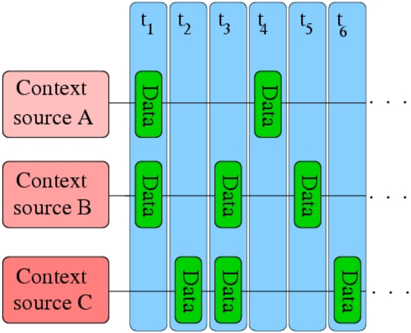
- Ideallised: Context data sources synchonised
  - Very unlikely



Aspects of prediction algorithms

Multi-dimensional time series

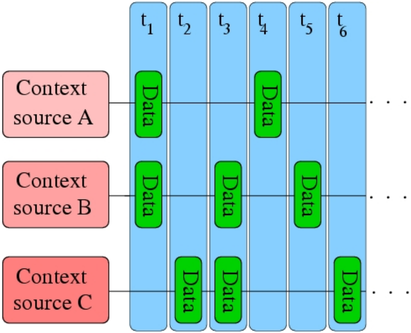
- Realistic scneario: No synchonisation between context sources
  - Context sources push information when specific events occur
  - Duty cycling (time differs between context sources)



Aspects of prediction algorithms

Multi-dimensional time series

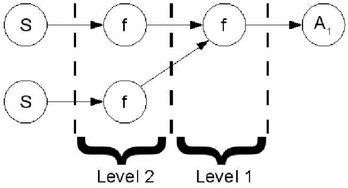
- Question: Which context values for a given time interval?
  - Interpolation of context values?
  - Last value measured?

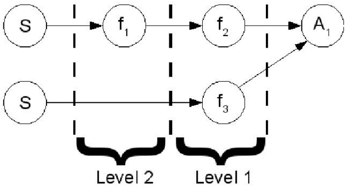


Aspects of prediction algorithms

Multi-dimensional time series

- Proposal/Idea: Context reasoning by DAG[1]
  - Design processing sequence that the algorithm shall follow/respect?
  - Designed for context reasoning – Also applicable for context prediction
  - Problems: How to design this processing graph on-line and autonomously?





[1] Bernd Niklas Klein, Sian Lun Lau, Andreas Pirali, Tino Löffler, Klaus David. *DAGR-DAG based context reasoning: An architecture for context aware applications*. In Proceedings of the eighth international workshop on applications and services in wireless networks, Kassel, Germany, pp. 20-25, 2008.

Aspects of prediction algorithms

Iterative prediction

- Prediction horizon can be extended by iterative prediction
  - Utilise predicted contexts as input
- Problem: Less accurate
  - Predicted contexts more error prone than measured values

Aspects of prediction algorithms

Prediction of context durations

- Context durations make a difference
  - Different duration of contexts might also indicate other situations/Contexts
  - It is more difficult to predict a context together with its occurence time instead of simply a context sequence
  - Duration can be modelled by repeatedly occurring contexts in a context sequence

Aspects of prediction algorithms

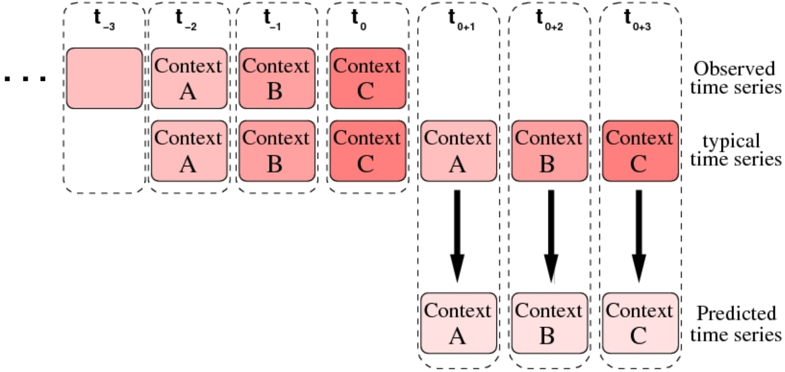
Relaxation of typical behaviour patterns

- 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

Relaxation of typical behaviour patterns

- Exact sequence matching



Aspects of prediction algorithms

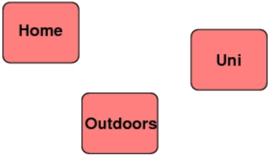
Context data types

- 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

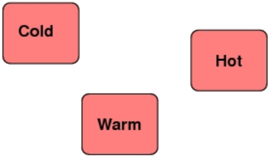
- Nominal contexts
  - $=$
  - $\neq$



Aspects of prediction algorithms

Context data types

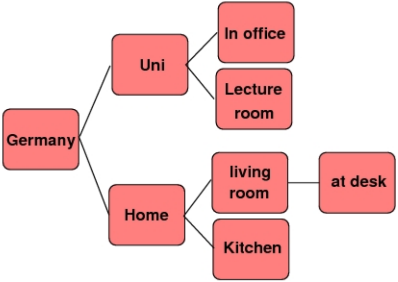
- Ordinal contexts
  - $<$
  - $>$
  - $=$



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	-	-	-	+

Aspects of prediction algorithms

Pre-processing of time series data

- For context prediction,preprocessing of context data is often applied
  - Identify typical context patterns
  - Derive occurrence probability of contexts
  - Derive context transition probabilities
- Distinguish between on-line and off-line processing
- Problem: Increased processing load

Aspects of prediction algorithms

Summary

	IPAM	ONISI	Markov	CRF
Numeric Contexts				
Non-numeric Contexts				
Complexity				
Learning ability				
Approximate matching				
Multi-dim. TS				
Discrete data				
Variable length patterns				
Multi-type TS				
Continuous data				
Pre-processing				
Context durations				
Continuous time				

Aspects of prediction algorithms

Summary

	PCA	SPM	Align	SOM
Numeric Contexts				
Non-numeric Contexts				
Complexity				
Learning ability				
Approximate matching				
Multi-dim. TS				
Discrete data				
Variable length patterns				
Multi-type TS				
Continuous data				
Pre-processing				
Context durations				
Continuous time				

Aspects of prediction algorithms

Summary

	ARMA	Kalman	SVM	DS
Numeric Contexts				
Non-numeric Contexts				
Complexity				
Learning ability				
Approximate matching				
Multi-dim. TS				
Discrete data				
Variable length patterns				
Multi-type TS				
Continuous data				
Pre-processing				
Context durations				
Continuous time				

Aspects of prediction algorithms

Summary

	EAs	NN	Sim. Anneal
Numeric Contexts			
Non-numeric Contexts			
Complexity			
Learning ability			
Approximate matching			
Multi-dim. TS			
Discrete data			
Variable length patterns			
Multi-type TS			
Continuous data			
Pre-processing			
Context durations			
Continuous time			

Outline

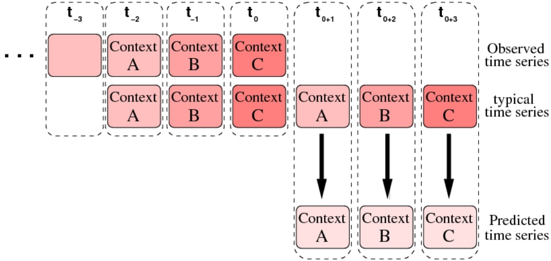
Simple prediction approaches: ONISI and IPAM

1. Important aspects of context prediction algorithms
2. Exact sequence matching
3. Algorithm: IPAM
4. Algorithm: ONISI

Exact sequence matching

Introduction

- File a given sequence for the exact occurence of a sub-sequence
- 'Pattern Matching' or 'String Matching'[2]
- Easily extended to context prediction:
  - Prediction $\equiv$ continuation of matched sequence



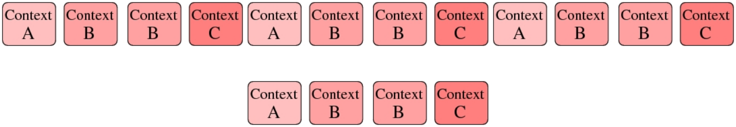
[2] Richard O. Duda, Peter E. Hard and David G. Stork, *Pattern Classification*, Wiley-Interscience, 2nd edition, 2001.

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

Example

Suppose we have a large text such as Herman Melville's Moby Dick and want to classify it as relevant to the topic of fish or to the topic of hunting.

- Keywords for the fish topic
  - might include 'salman', 'whale', fishing', 'ocean'
- Keywords for hunting
  - might include 'gun', 'bullet', 'shoot'.
- String matching would determine the number of occurrences of such keywords in the text.
- A simple count of keyword occurrences could then be used to classify the text according to topic

Exact sequence matching

String matching

Task

Determine whether a candidate string $\mathbf{x}$ is a substring of $\mathbf{y}$.

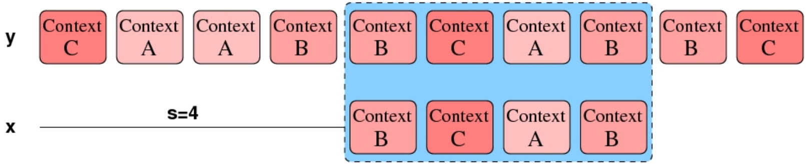
- Typically: $\mathbf{x} \ll \mathbf{y}$
- Each character in $\mathbf{x}$ and $\mathbf{y}$ is taken from an alphabet Σ
  - DNA bases,
  - Binary sequences ($\Sigma = \{0, 1\}$)
  - Alphanumeric sequences ($\Sigma = \{0, ..., 9, a, ..., z, A, ..., Z\}$)
  - Context sequences – Each character represents a context

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
```

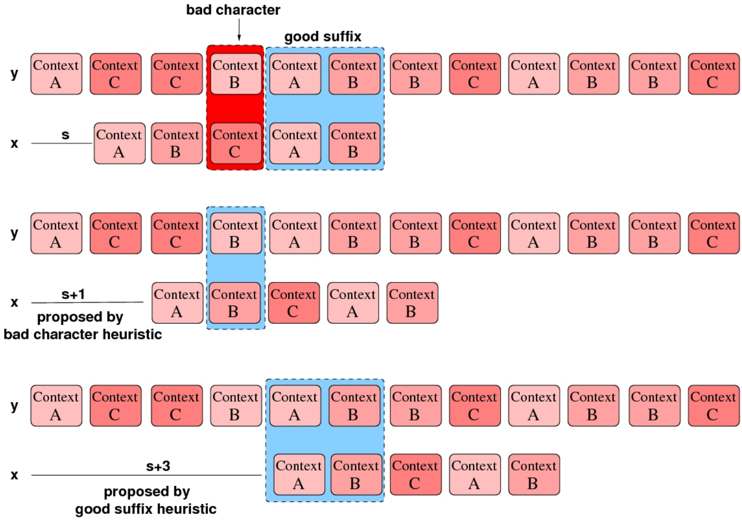
Exact sequence matching

Boyer-Moore string matching algorithm

- The Boyer-Moore algorithm utilises information known from recent candidate shifts
- Character compositions are done in reverse order
- increment to a new shift need not be 1
- Benefits from two heuristics:
  - Good suffix heuristic
  - Bad character heuristic

Exact sequence matching

Bad character heuristic and good suffix heuristic



Exact sequence matching

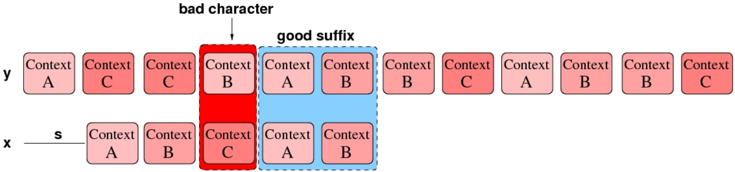
Bad character heuristic and good suffix heuristic

- Bad character heuristic
  - Since character comparisons proceed from right to left, bad character is found as efficiently as possible
  - Since current shift s is invalid, no additional character comparisons are required
  - Proposes incrementing the shift by an amount to align the rightmost occurrence of the bad character in $\mathbf{x}$ with the bad character identified in $\mathbf{y}$.
  - No valid shifts have been dropped
- Good suffix heuristic
  - A suffix of $\mathbf{x}$ is a substring of $\mathbf{x}$ that contains the final character in $\mathbf{x}$
  - At shift s the rightmost contiguous characters in $\mathbf{y}$ that match those in $\mathbf{x}$ are called the good suffix
  - Character comparisons are made from right to left and are therefore optimal

Exact sequence matching

Last occurrence function

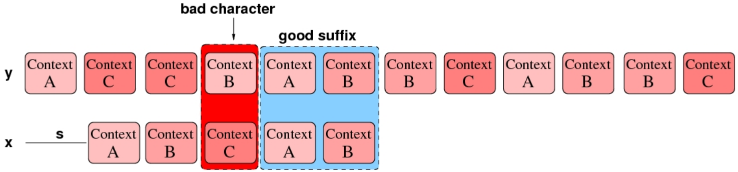
- Table containing every letter in the alphabet
- Plus position of its rightmost occurrence in **x**
- Example:
  - A, 4
  - B, 5
  - C, 3
- Computation only once
  - Does not significantly impact the runtime



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

Boyer-Moore algorithm

- Computational complexity
  - Homework / Exercise ;-)

Asymptotic computation time

$O(k) \longmapsto$ computation time not larger than $c \cdot k$ for a suitable constant c and k large.

Exact sequence matching

Problems

Problems with exact string matching approaches in Ubiquitous Computing

- Data might be distorted by noise
    - In Ubicomp scenarios we even expect noisy input data
    - Exact pattern matching not feasible then

Exact sequence matching

Approximate matching approaches

- Edit distance
- Nearest neighbour approaches

Exact sequence matching

Approximate matching approaches

- Nearest neighbour approaches
  - Strings are identified as vectors in a coordinate system
  - Since these vectors are numerical, distance/similarity between vectors can be computed by common metrics
- We will consider these approaches later in the lecture

Exact sequence matching

Approximate matching approaches

- Edit distance
  - Task: Compute the difference
  - Problem: What is the similarity/distance between string sequences?
  - Example: is 'abbccc' closer to 'aabbcc' or to 'abbcccb'?
- We will consider these approaches later in the lecture

Outline

Simple prediction approaches: ONISI and IPAM

1. Important aspects of context prediction algorithms
2. Exact sequence matching
3. Algorithm: IPAM
4. Algorithm: ONISI

Algorithm: 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[a]

[a] B.D. Davison and H. Hirsh, *Predicting sequences of user actions*. In: AAAI/ICML Workshop on predicting the future: AI approaches to time-series analysis. 1998

```
...
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
96102514:25:46 emacs
...
```

Algorithm: IPAM

Introduction and scenario

- It was observed that recently issued commands had the greatest impact on the follow-up command
- Idea: Standard learning algorithms ignore rare but possibly important events in a time series
    - IPAM was designed to improve this

Example

Predict hardware failures in routing networks. Typically every packet is routed as expected and no error occurs. In some, very rare cases, a hardware router might collapse. Likely result: packet losses, congestion, re-calculation of routing tables or a disconnected part of the network.
The event is rare but serious.

Algorithm: IPAM

Introduction and scenario

- Requirements of an optimal on-line learning algorithm
  - Have predictive accuracy at least as good as the bset known resource-unlimited methods
  - Operate incrementally (Modifying existing model rather than building a new one as new data is optained)
  - Be affected by all events (remembering uncommmon, but useful, events regardless of how much time has passed)
  - Do not necessarily retain a copy of all events observed
  - Output a list of predictions sorted by confidence
  - Adapt to changes to the target concept
  - Be fast enough for interactive use
  - Learn by passive observation
  - Apply even by absence of domain knowledge

Algorithm: IPAM

Algorithmic approach

- IPAM (Incremental Probabilistic Action Modeling)
  - on-line learning algorithm
  - Utilises last few events issued in order to predict the next event in a sequence of events
  - Prediction horizon: 1
  - Interative prediction possible
  - Impact of empiric factor: α

Algorithm: IPAM

Algorithmic approach

- Algorithmic operation
  - While observing sequence: matrix of prediction probabilities is maintained.
  - Colums: all possible events,
  - Rows: added and modified as events occur.
  - First event c_i: New row is added
    - Each column in this row holds probability that event observed after c_i was observed.
  - Row initialised with uniform probabilities $\frac{1}{n}$
  - Next event c_{i+1}: New row added and initialised with uniform probabilities.
  - Preceding row, every column multiplied with $0 \leq \alpha \leq 1$ and column c_i increased by $(1 - \alpha)$.
  - Probability to predict event sequences that have not been observed for some time diminishes

Algorithm: 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}		$\frac{1}{n}$	$\frac{1}{n}$	$\frac{1}{n}$	

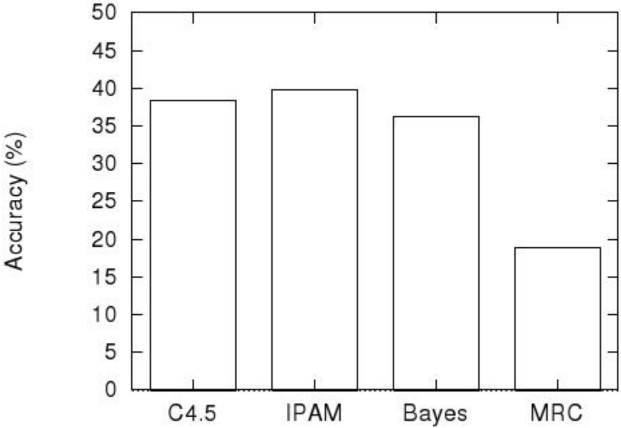
Algorithm: IPAM

Results and figures

- Collected command histories of 77 users (mostly undergraduate students)
- Over 168,000 commands executed
- During a period of 2-6 months
- Average user over 2000 command instances
- 77 distinct commands per user on average
- 8.4% of the commands were new and had not been logged previously
- Users repeated the last command 20% of the time

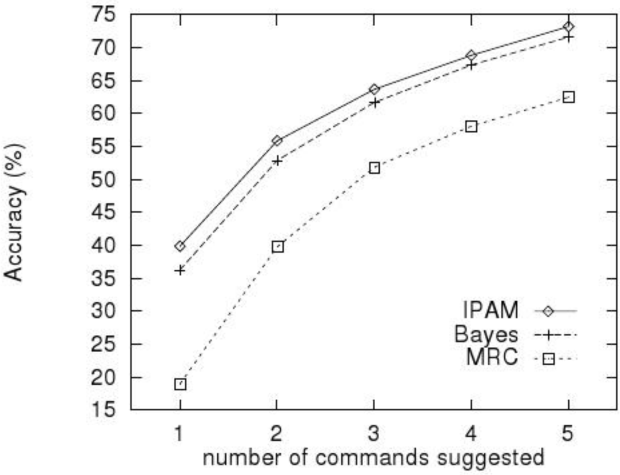
Algorithm: IPAM

Results and figures



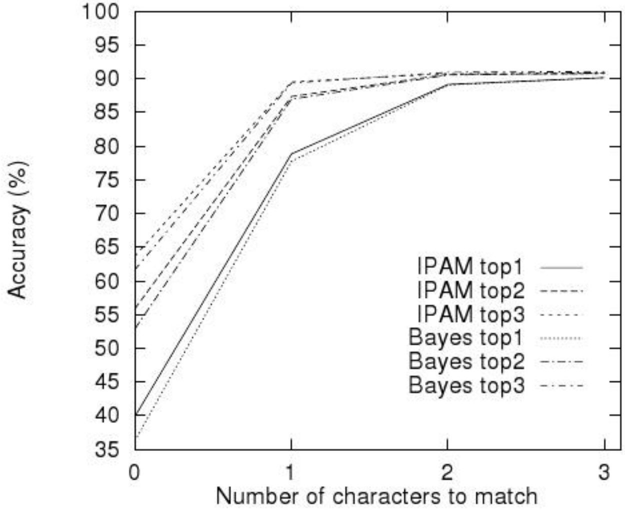
Algorithm: IPAM

Results and figures



Algorithm: IPAM

Prediction accuracy



Algorithm: IPAM

Adaptability

- The IPAM algorithm has only restricted ability to adapt to changing environments
  - It can learn new context sequences
  - The importance of the occurrence of context sequences can change
  - No support for newly observed contexts
  - When new contexts occur in an environment, the algorithm can not make use of this

Algorithm: IPAM

Memory and processing load

- Memory load
  - IPAM keeps a table in memory of size $O(k^2)$
  - $k =$ number of distinct commands
- Processing load
  - Predictions performed in constant time
  - Updates require time $O(k)$

Algorithm: IPAM

Multi-dimensional time series

- The IPAM algorithm is not well suited for multi-dimensional time series
  - It was designed for one-dimensional command-line input
  - In scenarios with more than one context source the approach is not feasible
  - Possible: Aggregation of multi-dimensional time series to one-dimensional time series.

Algorithm: IPAM

Iterative prediction

- Iterative Prediction possible
  - Steep decrease in prediction accuracy expected since prediction horizon is only 1

Algorithm: IPAM

Prediction of context durations

- Prediction of context duration not possible
  - Algorithm was designed to predict the occurrence of the next event.
  - Event durations are not considered by the algorithm
  - Only simple sequence of occurring events possible

Algorithm: IPAM

Approximate matching of patterns

- Exact pattern matching
  - The IPAM algorithm utilises exact pattern matching
  - Approximate matching was not implemented
  - Theoretically it is possible to implement approximate matching for the IPAM algorithm.

Algorithm: IPAM

Context data types

- All data types supported by IPAM
  - Cathegorical data utilised for prediction
  - IPAM considers typical context patterns for prediction
  - Numerical, fluctuating data will, however blow up the memory requirement and decrease the prediction accuracy as exact pattern matching is applied.
  - Trends are not considered

Algorithm: IPAM

Pre-processing

- Pre-processing required but low computational complexity
- Algorithm designer has to specify all possible events/contexts
- Computational complexity to initialise the prediction matrix: $O(k)$

Algorithm: IPAM

Conclusion

- Low computational complexity
- Low memory requirements
- Cathegorical time series prediction
- Prediction of typical patterns – no trends considered
- Prediction history only of length 2 – Complex command sequences can thus not be distinguished
- All events known in advance: No adaptive operation when also occuring contexts might change.
- Not well suited for flexible, changing ubiquitous environments

Aspects of prediction algorithms

Summary

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

Outline

Simple prediction approaches: ONISI and IPAM

1. Important aspects of context prediction algorithms
2. Exact sequence matching
3. Algorithm: IPAM
4. Algorithm: ONISI

Algorithm: ONISI

Introduction and scenario

Scenario

The use of an unmodified application by a user shall be observed in order to build application and usage-models

- Observe user-interactions with the application interface[3]
- From these observations a state-space is build which the user navigates
- Stochastic properties of state transitions are also modelled
- Task: Observe a user and model its decision process

[3] Peter Gorniak and David Poole, *Predicting future user actions by observing unmodified applications*. In Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence, pp. 217-222, 2000.

Algorithm: ONISI

Introduction and scenario

- Challenges when recent results are to be applied to new application
  - Results often do not transfer easily
  - Implementation used in research uses modified application
    - Non-trivial to repeat modification
    - Time-consuming to repeat modification
    - increases application complexity.
  - Hand-crafted application models required
    - time consuming

Algorithm: ONISI

Introduction and scenario

Approach

Extract knowledge from a user's interaction with the application

- No prior knowledge of the application
  - Purpose
  - Structure
- No modification to the application

- Predict future actions building on the usage-model extracted from an application

Algorithm: ONISI

Algorithmic approach

- ONISI (ON-line Implicit State Identification)
  - Assign probabilities to all possible actions in the currently observed interface state
  - Employs k nearest neighbours scheme
    - Metric: sequence match length
  - Java implementation
    - Wrapper to existing java applications
    - Able to record interfaces of java applications
    - No modification of application required

Algorithm: ONISI

Algorithmic approach

State of a user

A state of a user consists of a combination of the user's internal state and the application's interface state.

- Attempt: Try to determine the policy the user is employing from our observation of the user's interaction history.

Algorithm: ONISI

Algorithmic approach

- Prediction: Search interaction history for behavioural patterns similar to current pattern
- Required:
  - Observed pattern to extract from the interaction history
  - Method to determine occurrence of pattern in history
  - Function that ranks possible actions

Algorithm: 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 prediction 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 $l_t(s, a)$
- $\frac{l_t(s,a)}{\sum_i l_t(s,a_i)}$

Algorithm: 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 $f_t(s, a)$
- $\frac{l_t(s,a)}{\sum_i l_t(s,a_i)}$

Algorithm: ONISI

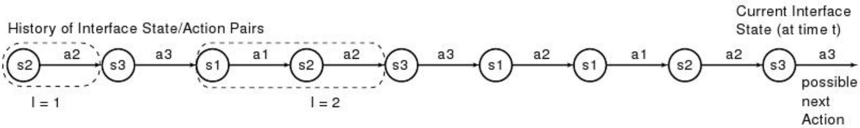
Algorithmic operation

- 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 has occurred: $f_t(s, a)$
- Return ranking

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

Algorithm: ONISI

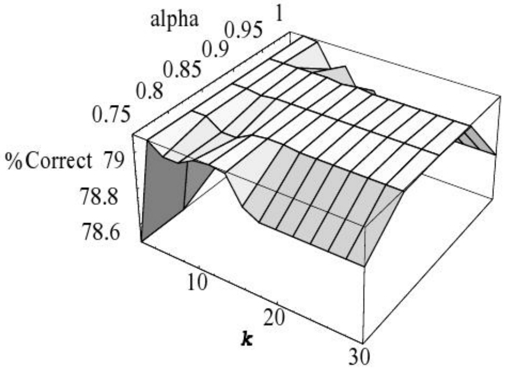
Algorithmic operation



- 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: $\{2,1,0\}$
- $$l_t(s_3, a_3) = \frac{0+1+2}{3} = 1 \quad (2)$$
- $$R_t(s_3, a_3) = 0.9\frac{1}{5} + 0.1\frac{50}{100} = 0.18 + 0.05 = 0.23 \quad (3)$$

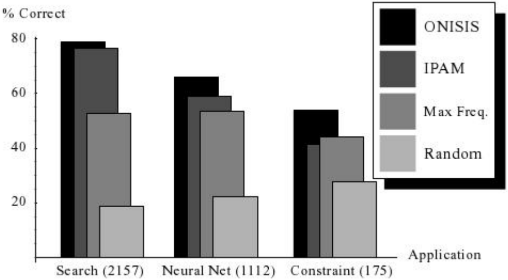
Algorithm: ONISI

Results and figures – Performance at various parameter settings



Algorithm: ONISI

Prediction accuracy – Performance



Algorithm: ONISI

Prediction horizon

- Prediction horizon can be extended by iterative prediction
  - Utilise predicted contexts as input
- Problem: Less accurate
  - Predicted contexts more error prone than measured values

Algorithm: ONISI

Adaptability

- The ONISI is well adaptable to arbitrary java applications
  - It can learn new context sequences
  - Also, new events can be observed

Algorithm: ONISI

Memory and processing load

- Exercise ;-)

Algorithm: ONISI

Multi-dimensional time series

- The ONISI algorithm is not suited for multi-dimensional time series
  - It was designed for one-dimensional nput
  - In scenarios with more than one context source the approach is not feasible
  - Possible: Aggregation of multi-dimensional time series to one-dimensional time series.

Algorithm: ONISI

Iterative prediction

- Iterative Prediction possible
  - Steep decrease in prediction accuracy expected since prediction horizon is only 1

Algorithm: ONISI

Prediction of context durations

- Prediction of context duration not possible
  - Algorithm was designed to predict the occurrence of the next event.
  - Event durations are not considered by the algorithm
  - Only simple sequence of occurring events possible

Algorithm: ONISI

Approximate matching of patterns

- Exact pattern matching
  - The ONISI algorithm utilises exact pattern matching
  - Approximate matching was not implemented
  - Theoretically it is possible to implement approximate matching for the algorithm.

Algorithm: ONISI

Context data types

- Only cathegorical time series data supported by ONISI
  - ONISI considers typical context patterns for prediction

Algorithm: ONISI

Pre-processing

- No Pre-processing required
- On-line approach

Algorithm: ONISI

Conclusion

- Low computational complexity (?)
- Cathegorical time series prediction
- Prediction of typical patterns
- Prediction history or arbitrary length
- Frequency and length of patterns considered
- Not well suited for flexible, changing ubiquitous environments

Aspects of prediction algorithms

Summary

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