Advanced Human Machine Interaction

(Interaction) Data Analysis

Alexandre Pauchet

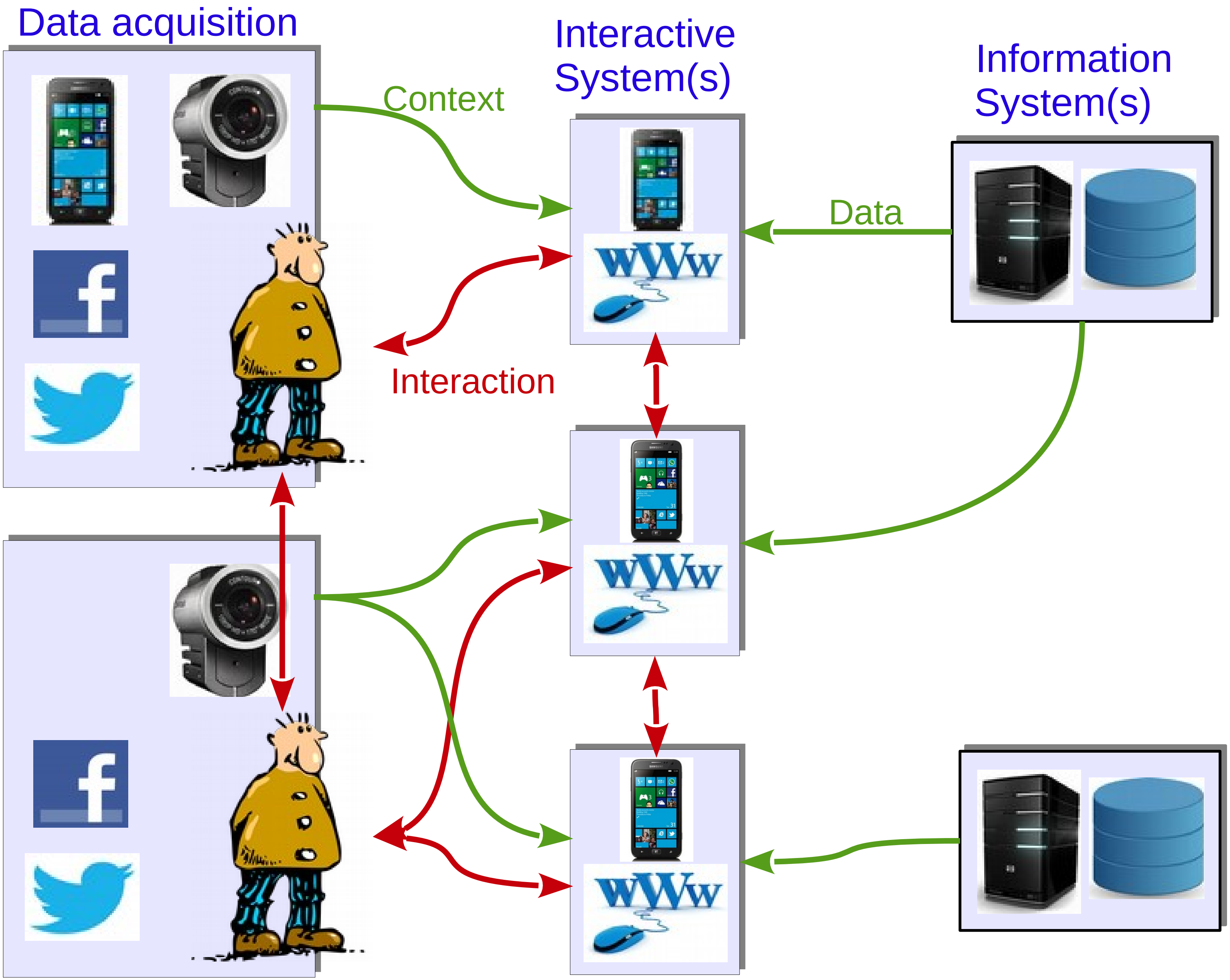
alexandre.pauchet@insa-rouen.fr - BO.B.RC18

Normandie Université

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN NORMANDIE

litis

IHME/IDA: objectives

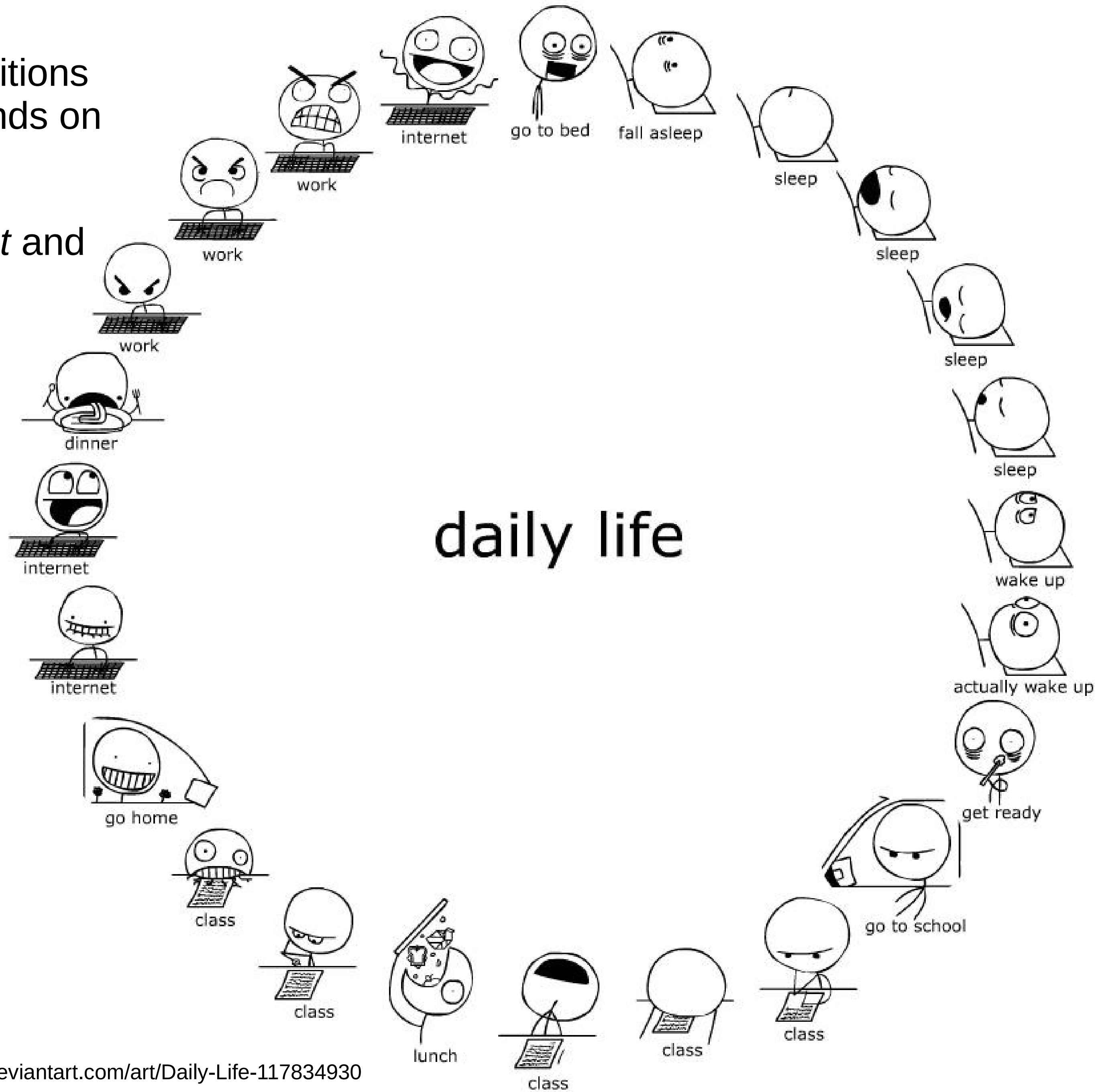


Data analysis: goals

Humans have habits:

- Habits ⇒ regularities, repetitions
- Recurrent behaviors depends on similar contexts

Context: set of *independent* and *sequential* (or *temporal*) characteristics that can be represented in orthogonal spaces



http://ennokni.deviantart.com/art/Daily-Life-117834930

Problem modeling

Formal modeling of a problem

- **Define formally the inputs**

  Ex: "*input* = $\{a_1, \ldots, a_n\}_{n<20}$ *a sequence of actions with each action* a_i *in {up, down, left, right, space}*"

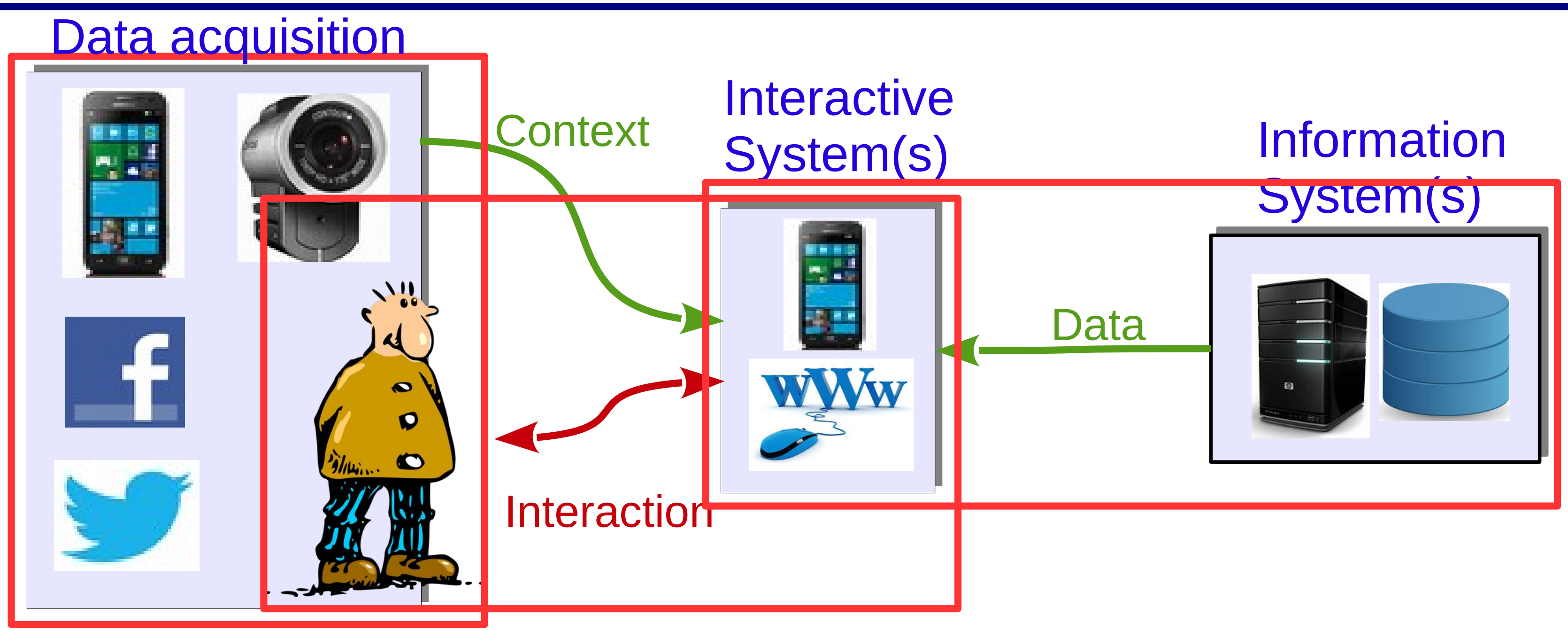
- **Define formally the outputs**

  Ex: "*output = a class among {beginner, advanced}*"

  Ex: "*output = a sequence of words generated as an answer to the user's query* $\{w_1, \ldots, w_n\}_{n>0}$"

- **Define a class of problem / algorithm**
  - Ex (machine learning): classification problem, clustering problem, regression problem, …
  - Ex (logic): induction / deduction problem, …
  - Ex (algorithm): sorting, graph construction, ...

Analysis of interaction data



The data to analyze can be collected from:

- External/internal sensors (context of use)
- Interaction data (log of user actions and/or activity)
- External data requested from the interactive system

Input: discrete data (1/2)

- **Discrete sequences**
  - Ex: sequence of visited web pages, of user actions
  - Representation: ordered set $\{a_1, a_2, \ldots, a_n\}$

- **Discrete sequences of item-sets**
  - Ex: actions from left and right hands in a double-handed game or multiple paddles/keyboards
  - Representations:
    - ordered set of item-sets

$$\{\{a_{1,1}, \ldots, a_{m,1}\}, \{a_{1,2}, \ldots, a_{m,2}\}, \ldots, \{a_{1,n}, \ldots, a_{m,n}\}\}$$

    - matrix

$$\{a_{i,j}\}_{i=1..m;\ j=1..n}$$

Input: discrete data (2/2)

- **Independent sequences**
  - Ex: actions from two different players
  - Representation: set of ordered sets
  - $\{\{a_{1,1}, \ldots, a_{m1,1}\}, \{a_{1,2}, \ldots, a_{m2,2}\}, \ldots, \{a_{1,n}, \ldots, a_{mn,n}\}\}$

Remarks:

- Time is not considered and delays may be different (or not) between two actions
- Different sampling are possible in independent sequences

Input: continuous and mixed data

- **Continuous signal**
  - Ex: user's voice volume
  - Representation: function f(t)

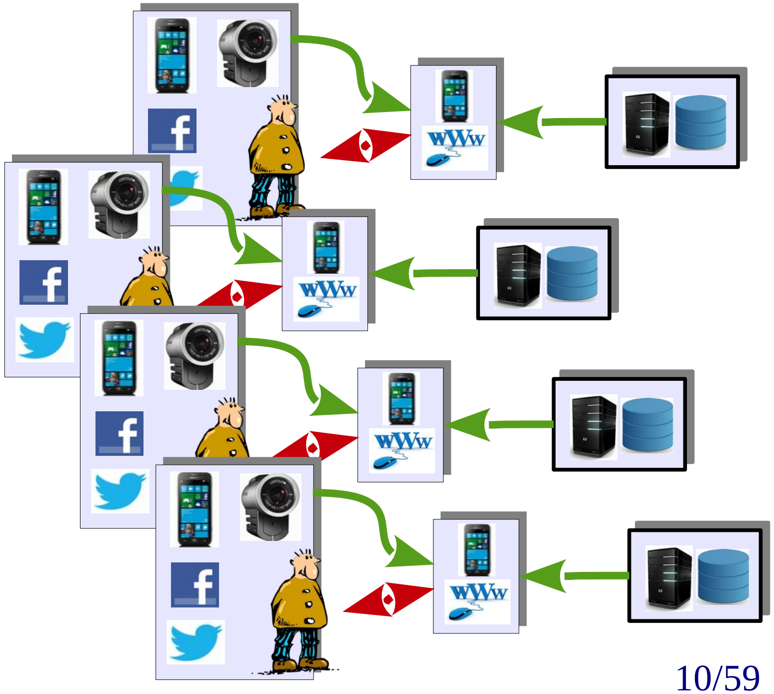
- **Continuous signals**
  - Ex: trajectories of two Wiimotes
  - Representation: set of functions $\{f_1(t), \ldots f_n(t)\}$

Remarks:

- Any continuous signal can be discretized
- Mix of discrete sequences and continuous signals
- There always is sampling

Goal / Objective (class of problem)

- **Behavioral pattern extraction**
  - Intra- / Inter- user behavioral patterns
  - (Multiple) Time scaling
  - Frequent patterns / similar patterns
- **User classification**
  - Clustering of users' activity
  - Segmentation and classification of parts of users' activity
- **Generation**
  - Simulation of user's behavior
  - Generation of interactive behavior
- **Combination of problems**

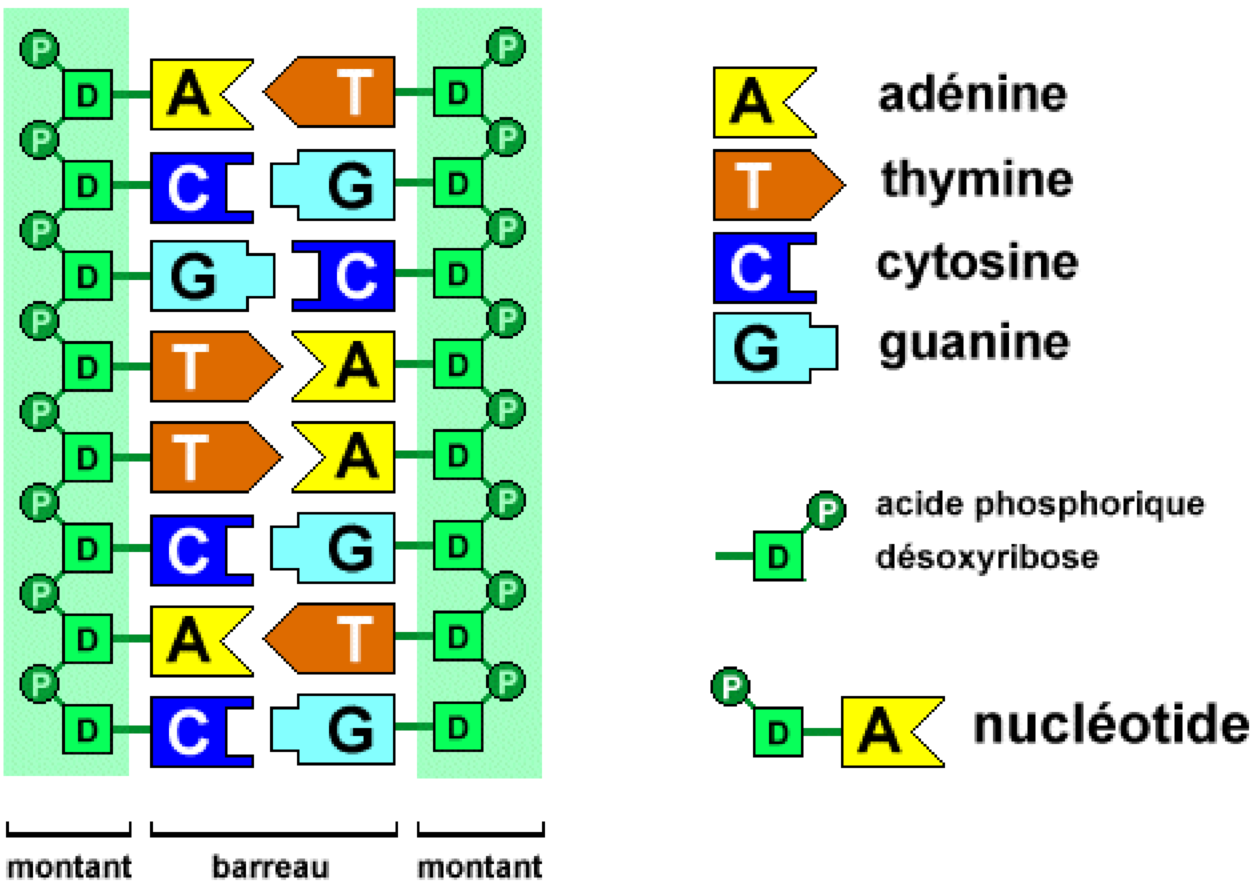


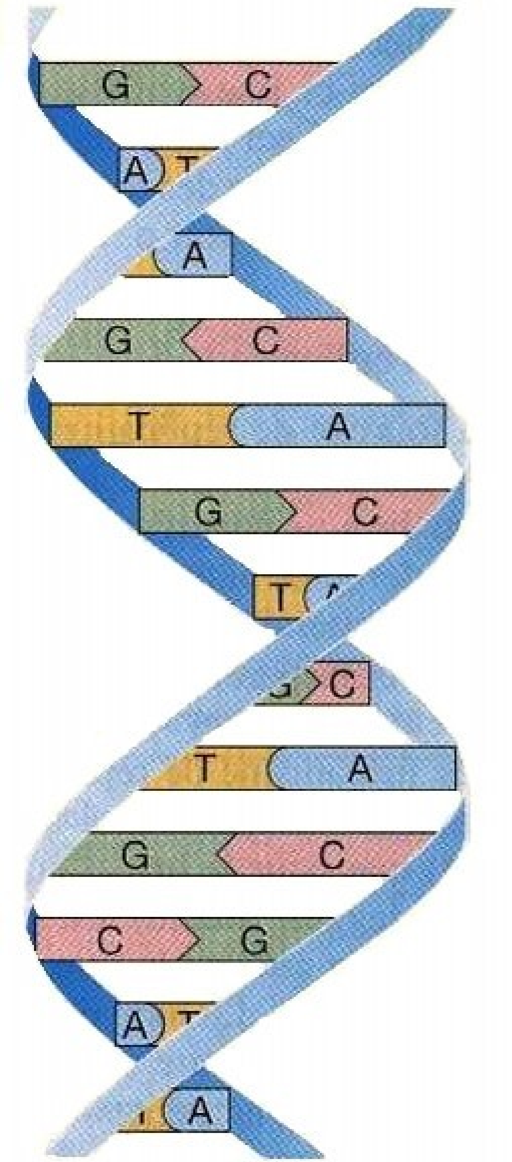
Exercise

- Let $A = \{a_1, ..., a_n\}$ be the set of n actions that users can perform. Let T be a set of r interaction traces from s different users $u_1, ..., u_s$:
  $T = \{\{a_{1,x1}, \ldots, a_{m1,x1}\}_{ux1} \ldots \{a_{1,mr}, \ldots, a_{mrxr}\}_{uxr}\}$
- eg. HMI with 3 possible actions: {{a1 a2 a1 a3}u1 {a2 a2 a2 a1 a3 }u2 {a2 a1 a2 a2 a1 a1 a3}u1 {a1 a1 a3}u2 {a1 a1 a2 a1 a3}u1 }
- Formalize the problem that, from a given sequence of actions, predicts the next action

Discrete sequences
Pattern extraction
Similarity-based approach

Sequence alignment





... A C T G T C A T C A T C T T A C T C A T C A T C T T A C G C T
A T A G C T A T A G T C A T C A T C T T A C G C T A T A G C A C
G C T A C G A A T C T C T G A A T A A C G C T A C G A A T C C ...

... C T G T C C T G C A T C A C T G G A T G T A C C A T C T T A C
T A T G G T A C A T C T A T G T A C T A C T C A C C - T T T T A A C
T C T T A C G C T A T A G C T A T A G T C T A C G A A T C A C ...

String mining and sequence alignment

- Alignment of two sequences of characters
  - Used to compare 2 sequences S_1 (l=m) and S_2 (l=n)
  - How S_1 can be transformed into S_2?
  - Based on a distance or a similarity measure
  - A sequence alignment can be computed using dynamic programming in $O(mn)$
- 2 types of alignments:
  - Global
  - Local (Smith & Waterman, 1981)

Sequence alignment

- $\begin{pmatrix} \mathtt{A} & \mathtt{C} & \mathtt{G} & - & - & \mathtt{A} \\ \mathtt{A} & \mathtt{T} & \mathtt{G} & \mathtt{C} & \mathtt{T} & \mathtt{A} \end{pmatrix}$ is an alignment of the two sequences “**ACGA**” and “**ATGCTA**”.
- Algorithmically, it corresponds to an edition script (i.e. a computer program)

Operation	**Resulting sequence**
Substitution of **A** by **A**	**A**
Substitution of **C** by **T**	**AT**
Substitution of **G** by **G**	**ATG**
Insertion of **C**	**ATGC**
Insertion of **T**	**ATGCT**
Substitution of **A** by **A**	**ATGCTA**

Local alignments

- 3 editing operations
  - Substitution of a symbol from S_1 at a given position by a symbol from S_2
  - Deletion of a symbol from S_1 at a given position
  - Insertion of a symbol in S_2 at a given position
- Scores
  - *Sub(a,b)*: score to substitute symbol *a* by symbol *b*
  - *Del(a)*: score to delete symbol *a*
  - *Ins(a)*: score to insert symbol *a*

Similarity measure

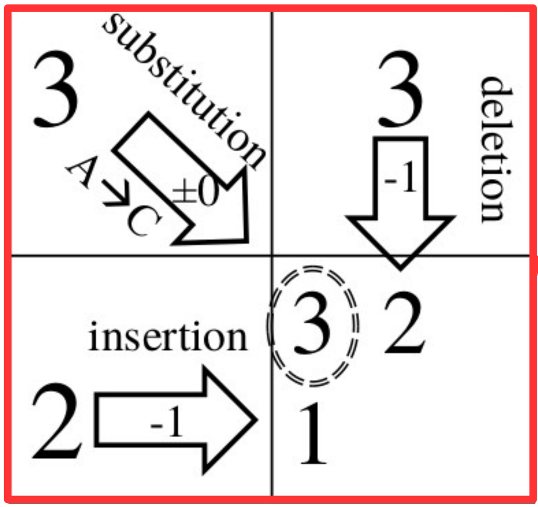
Similarity measure between 2 sub-sequences

$$\boldsymbol{s(x,y) = max \{ score\ of\ e \mid e\ in\ E_{x,y} \}}$$

- $E_{x,y}$: series of editing operations that transform *x* into *y*
- A score of *e* is computed as the sum of all its elementary editing operations

Dynamic programming

Smith & Waterman



$$
\begin{aligned}
t[-1,-1] &= 0,\\
t[i,-1] &= 0,\\
t[-1,j] &= 0,\\
t[i,j] &= max\begin{cases} t[i-1,j-1]+Subs(x[i],y[j]),\\ t[i-1,j]+Del(x[i]),\\ t[i,j-1]+Ins(y[j]),\\ 0 \end{cases}
\end{aligned}
$$

sub(x,x) = 1
sub(x,z) = 0
Ins = del = -1

	A	C	G	T	C	G	A	C	G
A	1	0	0	0	0	0	1	0	0
C	0	2	1	0	1	0	0	2	1
T	0	1	2	2	1	1	0	1	2
C	0	1	1	2	3	2	1	1	1
A	1	0	1	1	2	3	3	2	1
C	0	2	1	1	2	2	3	4	3
G	0	1	3	2	1	3	2	3	5

Remarks:

- Subs/Del/Ins have negative values
- The value at (i,j) position in table t only depends on the 3 adjacent positions
- An optimal alignment (i.e. of maximum score) can be produced by performing a trace back in the values of table t from the (maximal) values up to a position of 0.

Example

	A	C	G	T	C	G	A	C	G
A	1	0	0	0	0	0	1	0	0
C	0	2	0	0	1	0	0	2	0
T	0	0	0	1	0	0	0	0	0
C	0	1	0	0	2	0	0	1	0
A	1	0	0	0	0	0	1	0	0
C	0	2	0	0	1	0	0	2	0
G	0	0	3	1	0	2	0	0	3

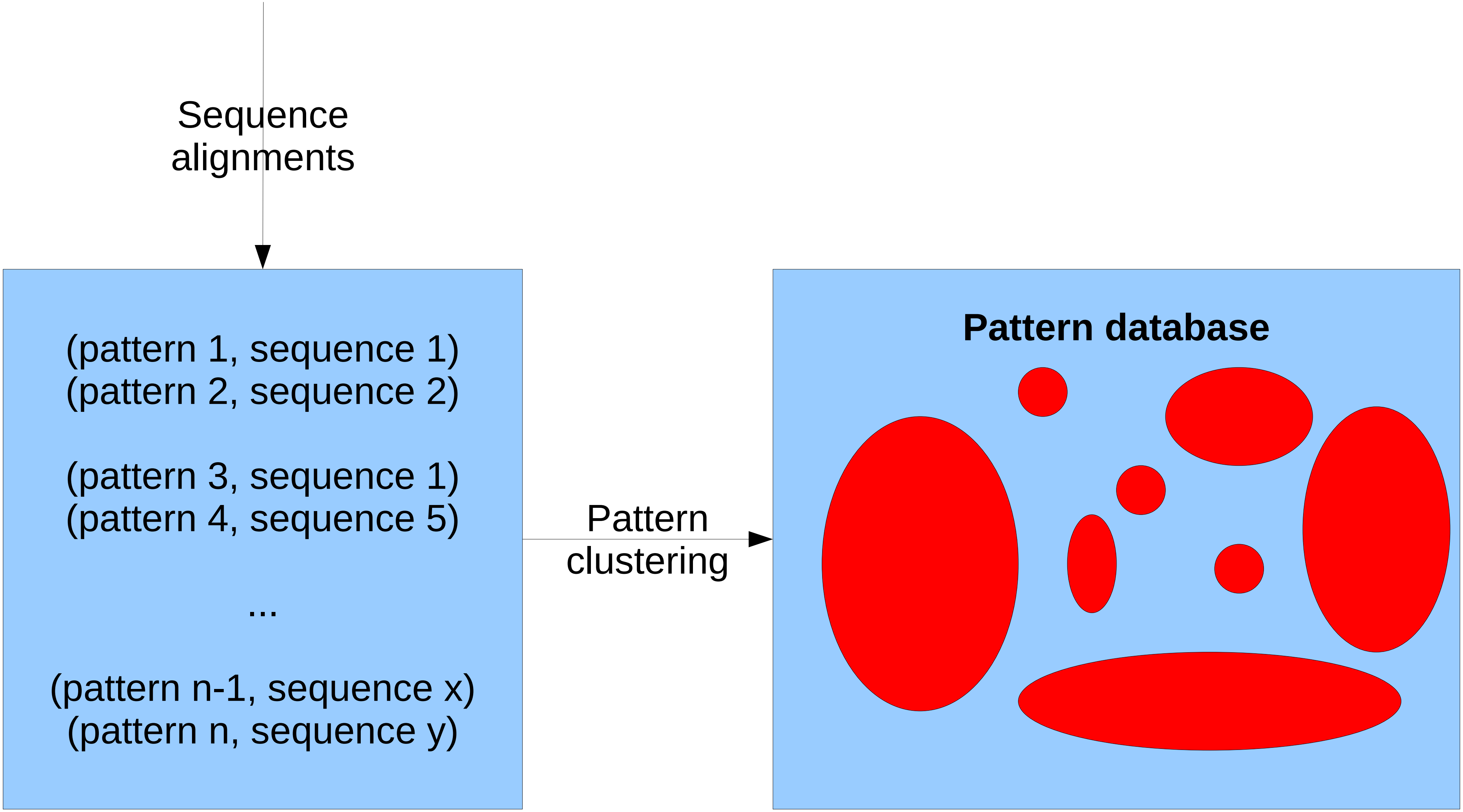
	A	C	G	T	C	G	A	C	G
A	1	0	0	0	0	0	1	0	0
C	0	2	1	0	1	0	0	2	1
T	0	1	2	2	1	1	0	1	2
C	0	1	1	2	3	2	1	1	1
A	1	0	1	1	2	3	3	2	1
C	0	2	1	1	2	2	3	4	3
G	0	1	3	2	1	3	2	3	5

Two accumulated similarity tables obtained using the Smith-Waterman algorithm. The left has been calculated using a similarity score of 1 for matches, and dissimilarity penalties of -2 for non-matching substitutions and indels. The right table has this penalty reduced to -1. In each case, the alignments with a similarity score of at least 3 have been highlighted. Note how the higher penalty leads to smaller, more local alignments.

Exercise

- Formalize the problem corresponding to the pattern extraction from discrete sequences using a similarity-based approach

From patterns to frequent patterns



Discrete sequences
Pattern extraction
Frequency-based approach

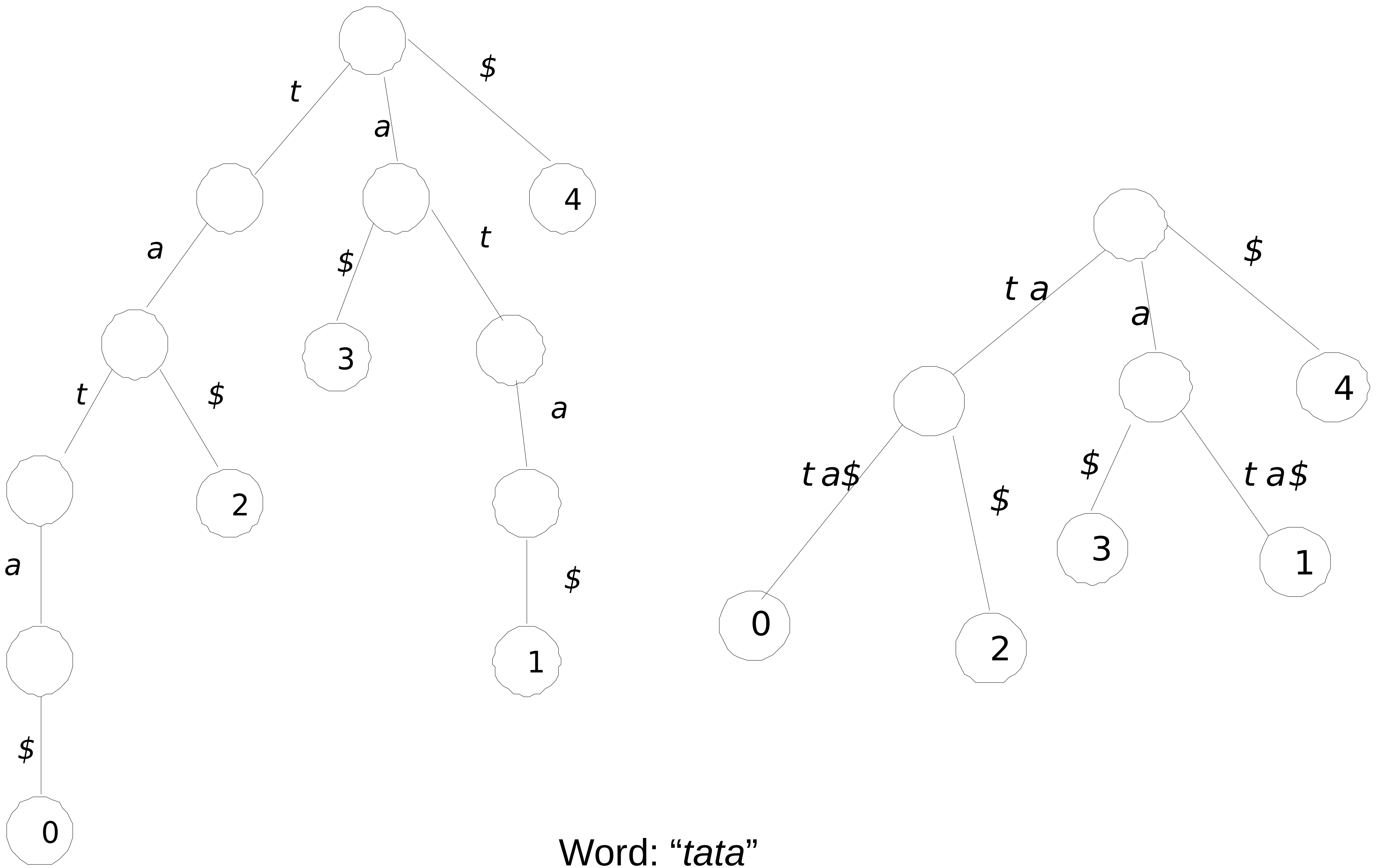
Approach

- Frequent sequential patterns
  - Ex: *users that perform action ‘A’, often perform action ‘B’ shortly after*
  - Association rules (‘A’ => ‘B’ shortly after)
  - Gap: number of elements (actions) between ‘A’ and ‘B’
  - Confidence: how often a rule has been found to be true
- A simple structure: suffix tree

Suffix trees

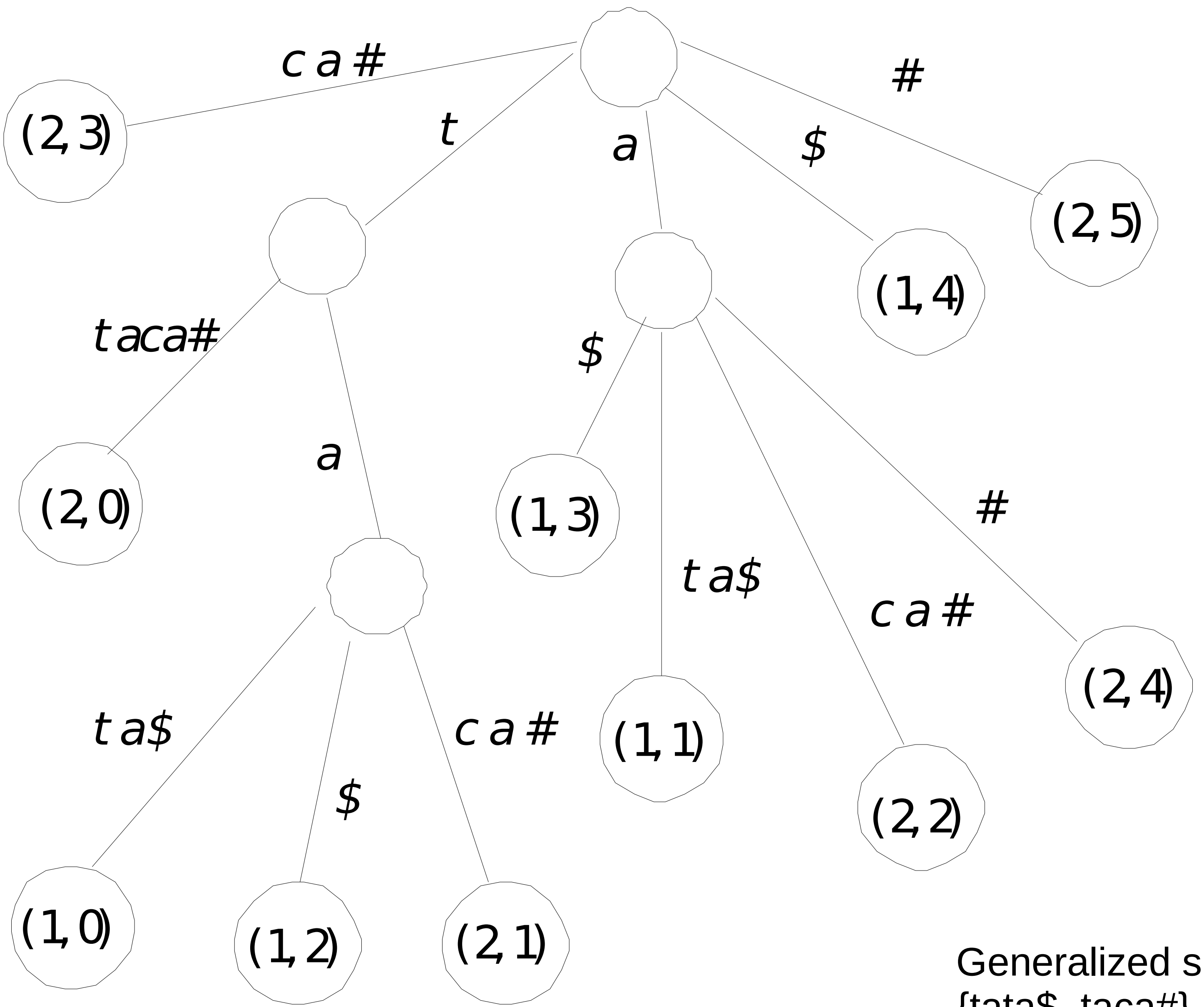
- The **non-compact suffix tree** of a word *y* is the deterministic finite automaton, having a single initial condition called *root* and where the terminal states correspond to the suffix of the word. The language recognized by this automaton is all suffixes of y.
- In practice a terminator is added at the end of the word (usually denoted $).
- Leaves are numbered according to the starting position of the suffix they recognize.
- To **compact the tree**, the internal nodes having only a single outgoing branch are removed and the branches are concatenated.

Example of suffix tree (single word)



Word: “*tata*”

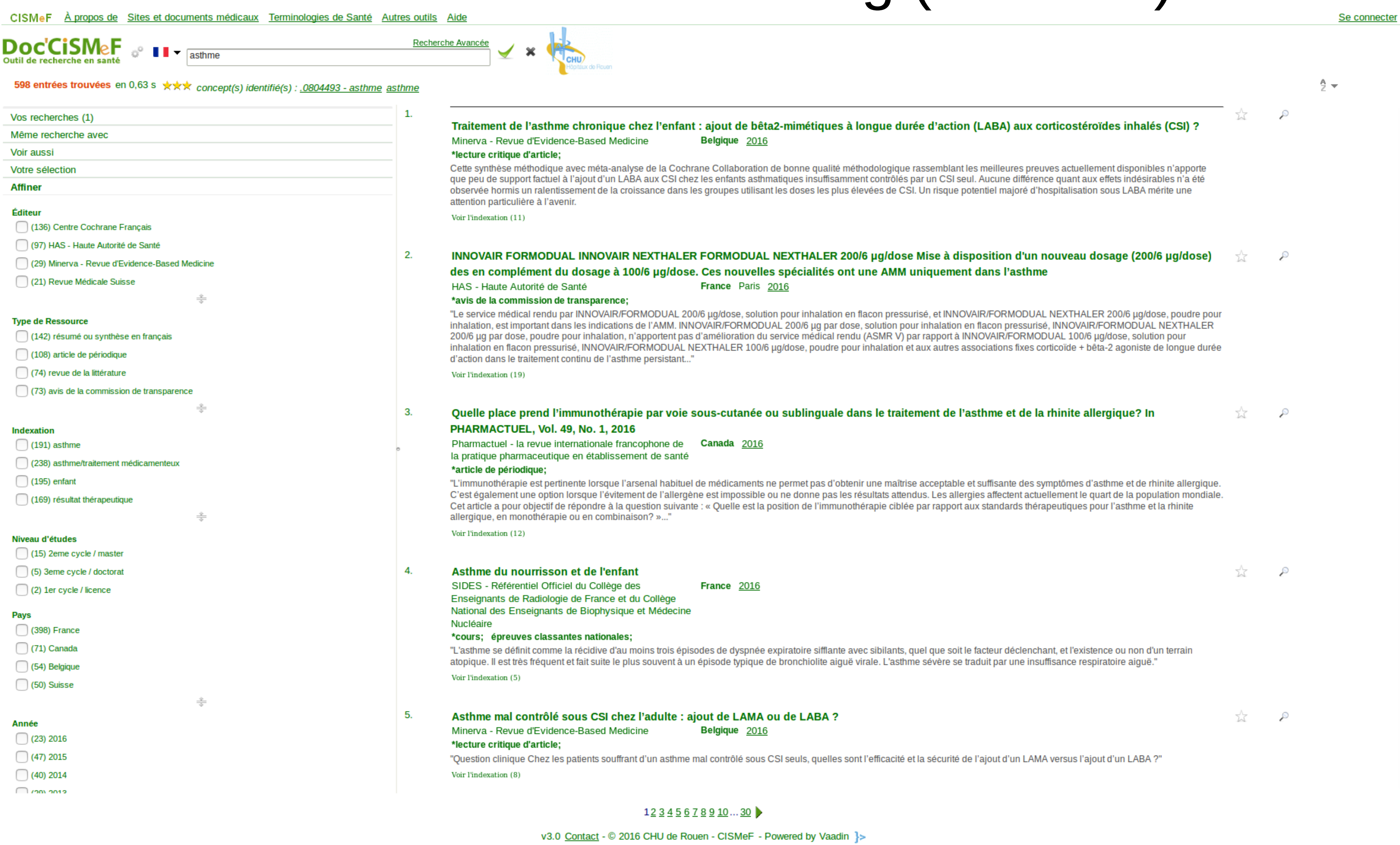
Generalized suffix tree (multiple words)



Generalized suffix tree for
{tata$, taca#}

Application: CISMeF

- Extraction of recurrent behaviors in the navigations within an online health catalog (CISMeF)



Application: CISMeF

- **Data preparation**
  - Episode extraction: IP + semantic distance between documents + time between requests
  - Resource identification: unique ID + delimiter

    example of session: /59451/ /303901/ /170702/

- **Recurrent pattern extraction**
  - Generalized suffix tree
  - Longest repeated substrings

Application: CISMeF

Nombre de liens visités	Nombre d'épisodes	Proportion
1	34 005	70,6 %
2	8 254	17,1 %
3	2 940	6,1 %
4	1 284	2,7 %
5	658	1,4 %
6	346	0,7 %
7	216	0,4 %
8	139	0,3 %
9	91	0,2 %
10	60	0,1 %
>10	175	0,4 %

22 days of log analysis:

- 10mn max for an episode
- 48 168 episodes
- 17mn of data processing (2,39GHz/512Mo)

Longueurs des motifs	2	3	4	5
Nombres de motifs	1557	146	20	4

	Episodes contenant un motif			
	longueur 2	longueur 3	longueur 4	longueur 5
Effectifs	4127	326	42	8
% épisodes	8,568	0,677	0,087	0,017
% épisodes (l>1)	29,139	2,302	0,297	0,056

Discrete sequences

- Pattern extraction:
  - Sequence alignments

    → similarity OK, but frequency is difficult to evaluate (should be paired with pattern clustering)

- Prediction:
  - Generalized suffix trees
  - Seq2seq, CRF, HMM, ...

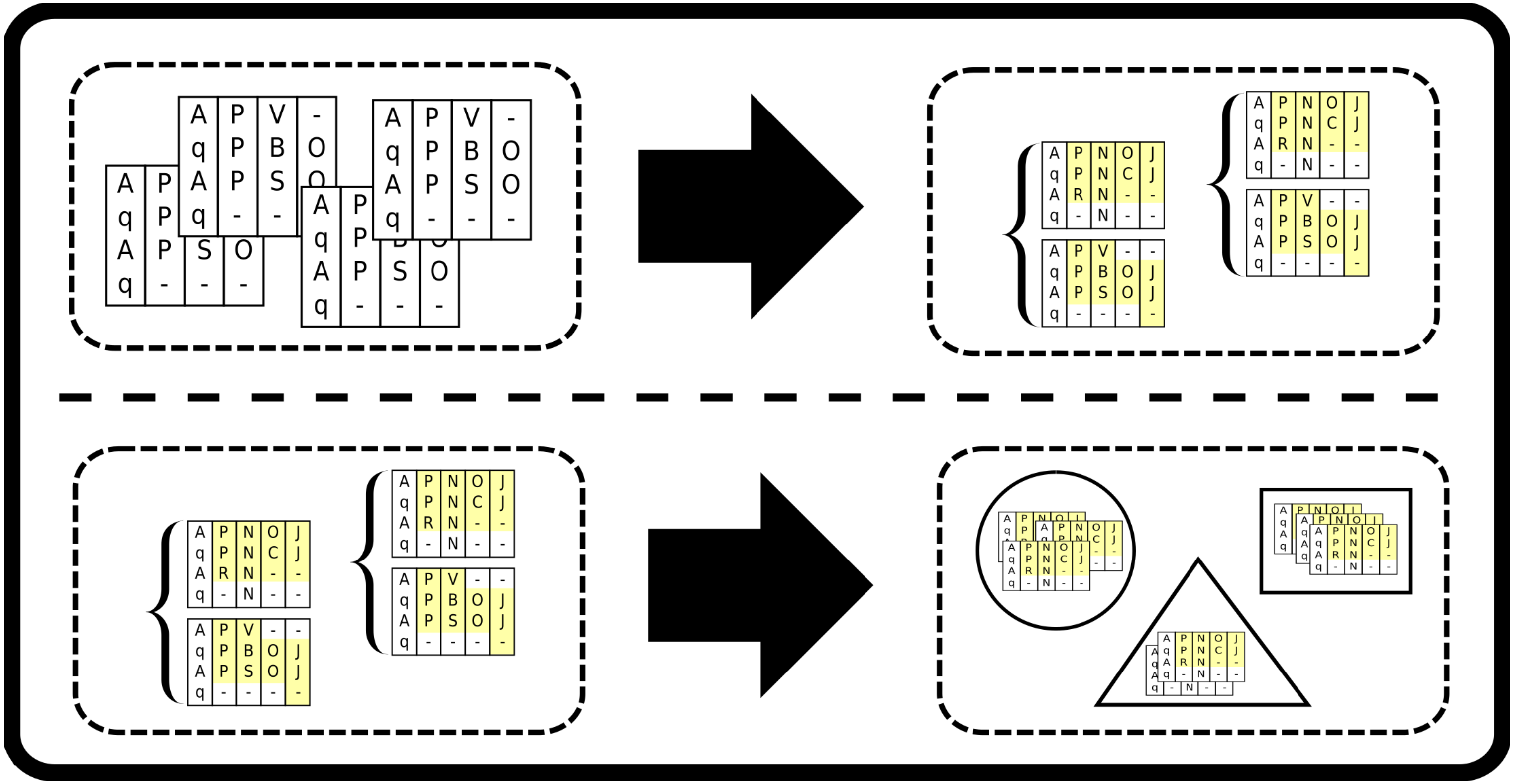
    → frequency OK, but only slight variations (similarity) can be taken into account

Exercise

- Formalize the problem that consists in predicting the most probable action of user given the set of previous actions performed?
- What would be an algorithm to construct a (compact) suffix tree?

Discrete sequences of item-sets
Pattern extraction
Similarity-based approach

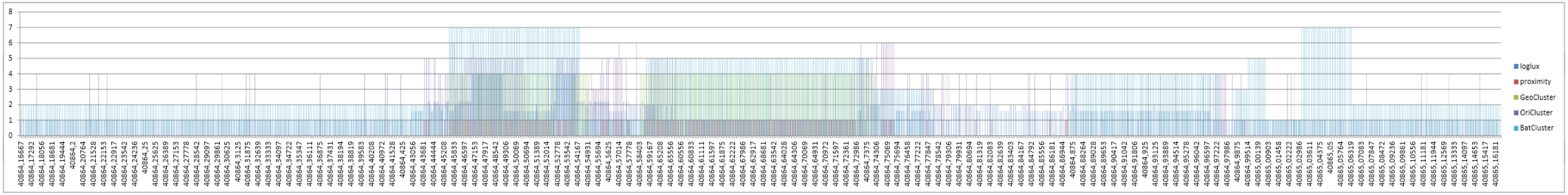
Approach (similar to sequences)

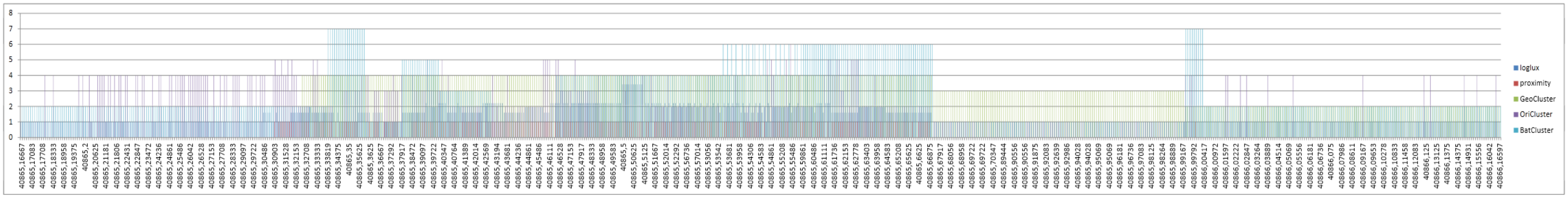


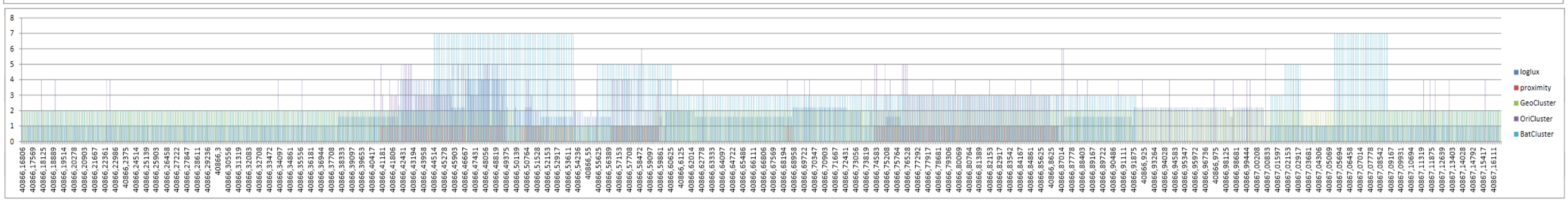
Decomposition in two separated steps

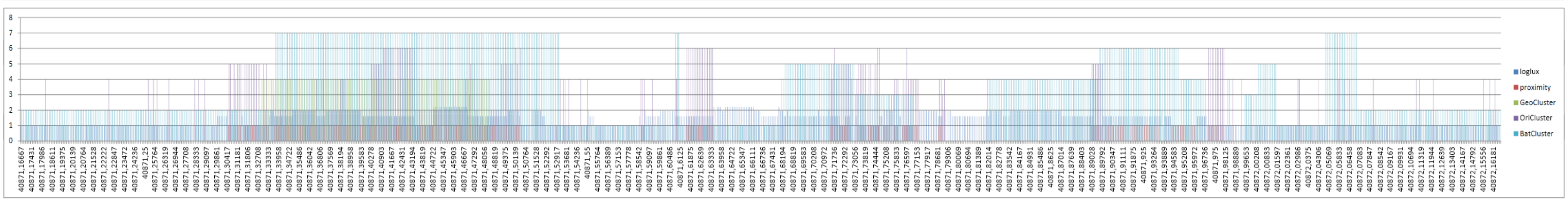
- Extraction of (pair of) interaction patterns
- Clustering of interaction patterns

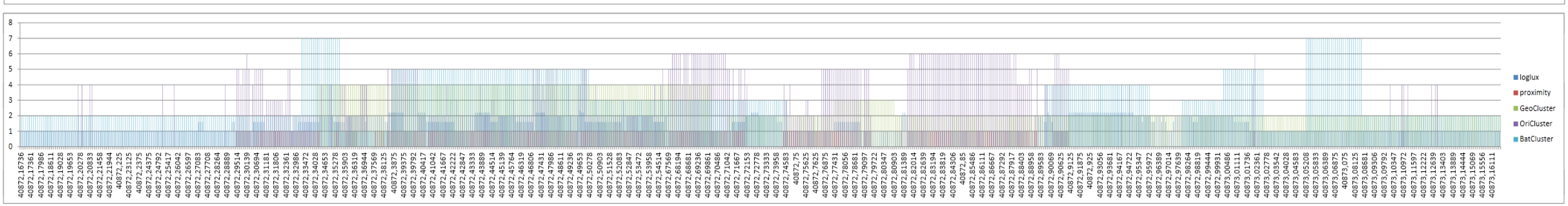
Routine extraction by matrix alignment





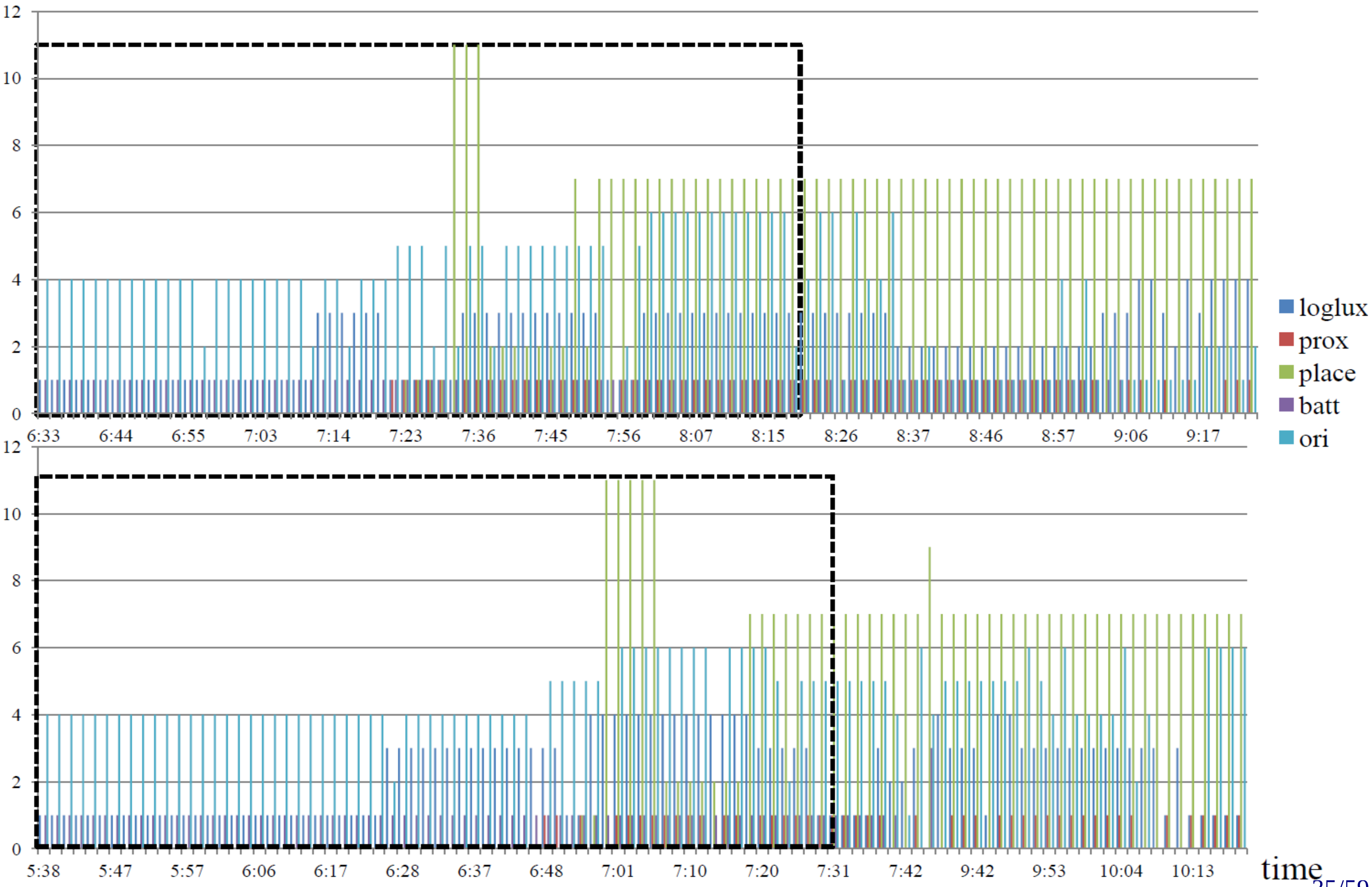




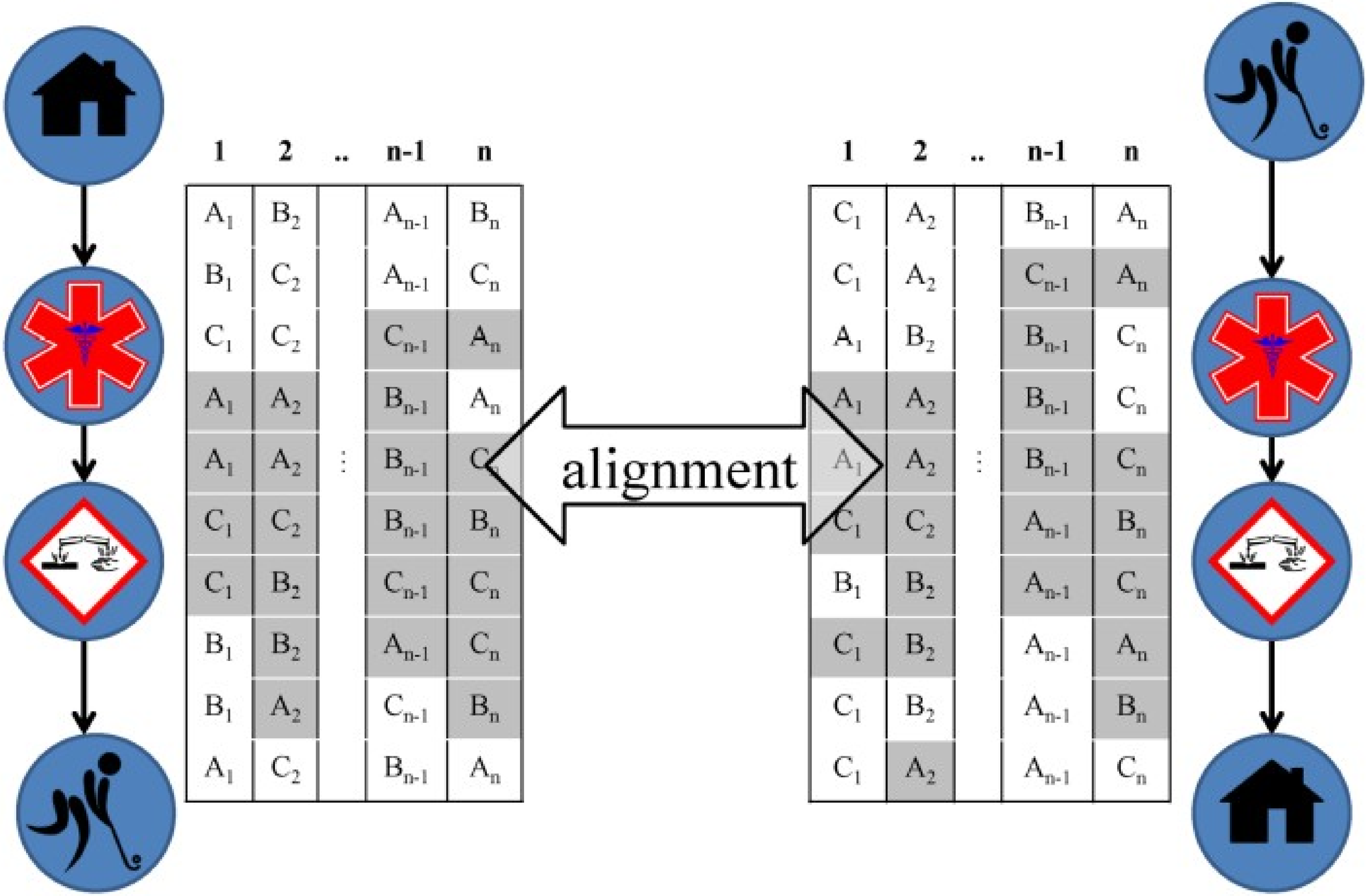


Rick Moritz (PhD thesis), Orange labs - Meylan

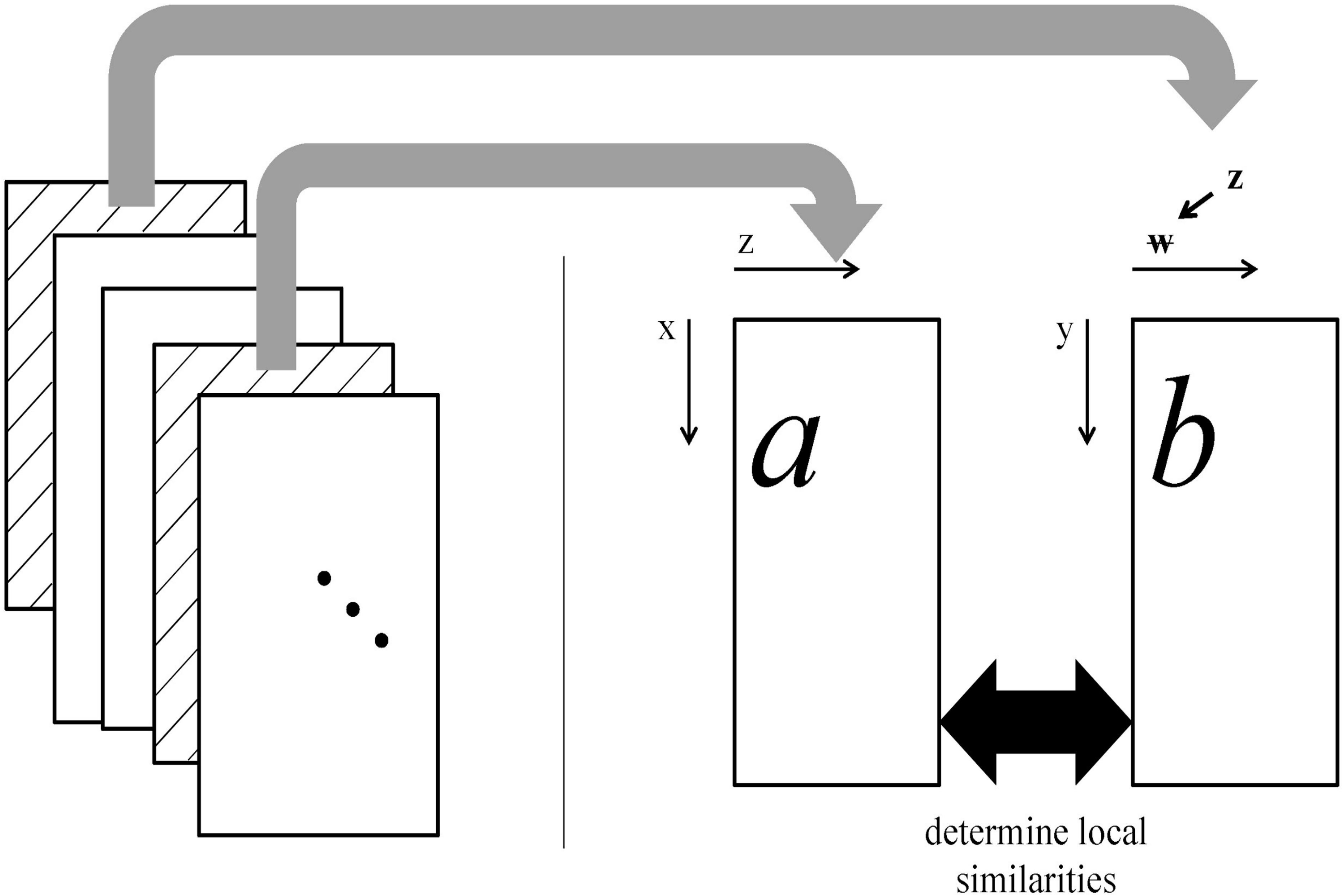
Routine extraction by matrix alignment



Routine extraction by matrix alignment



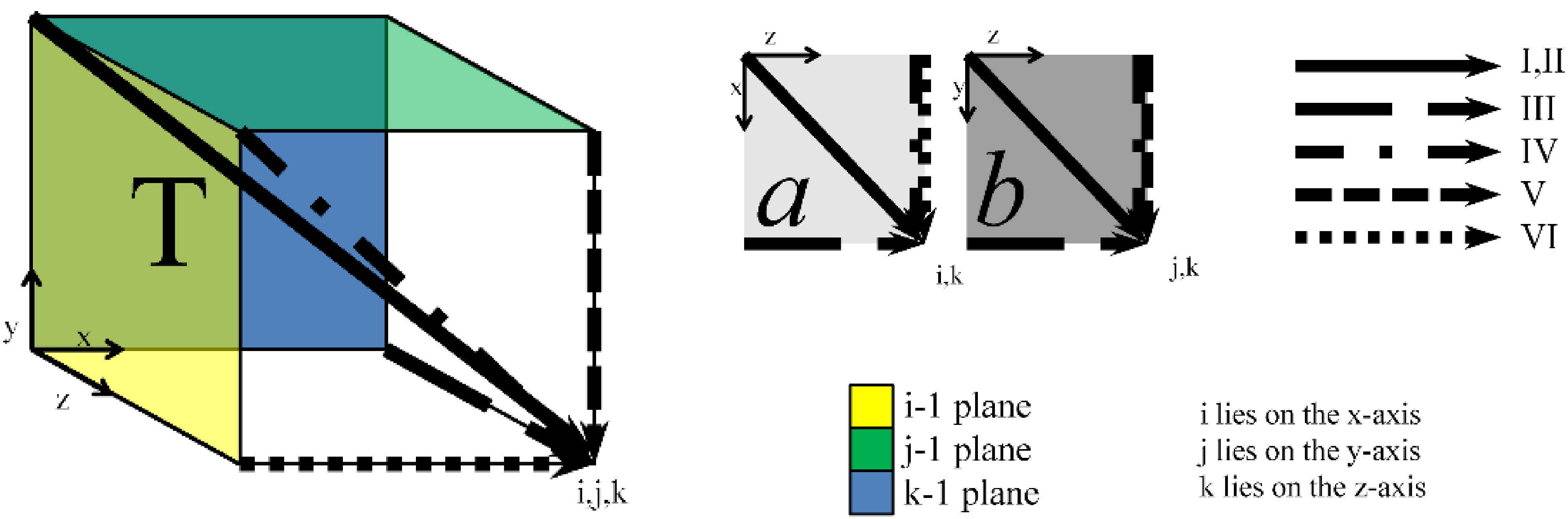
Routine extraction by matrix alignment



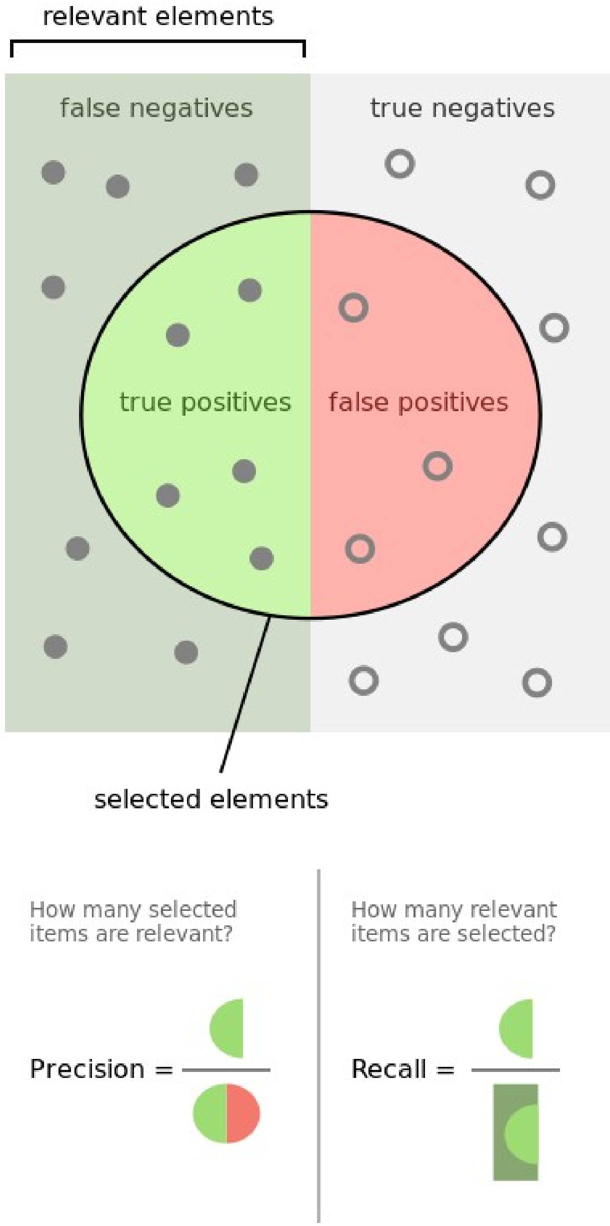
Routine extraction by matrix alignment

$$\mathrm{sim}(a,b) =$$

$$\max \begin{pmatrix} 0 & \\ \mathrm{sim}(a_{1,1}, b_{1,1}) + \mathrm{sim}(a\,[0,0]\uparrow, b\,[0,0]\uparrow) + \mathrm{sim}(a\,[0,1]\leftarrow, b\,[0,1]\leftarrow) & \text{(I)} \\ \mathrm{sim}(a_{1,1}, b_{1,1}) + \mathrm{sim}(a\,[1,0]\uparrow, b\,[1,0]\uparrow) + \mathrm{sim}(a\,[0,0]\leftarrow, b\,[0,0]\leftarrow) & \text{(II)} \\ \mathrm{sim}(a_{0,1}, b_{0,1}) + \mathrm{sim}(a\,[0,0]\uparrow, b\,[0,0]\uparrow) & \text{(III)} \\ \mathrm{sim}(a_{1,0}, b_{1,0}) + \mathrm{sim}(a\,[0,0]\leftarrow, b\,[0,0]\leftarrow) & \text{(IV)} \\ \mathrm{sim}(a_{0,0}, b_{1,0}) + \mathrm{indel}(b[0,0]) + \mathrm{sim}(a[0,0]\leftarrow, b[1,0]\leftarrow) & \text{(V)} \\ \mathrm{sim}(a_{1,0}, b_{0,0}) + \mathrm{indel}(b[0,0]) + \mathrm{sim}(a[1,0]\leftarrow, b[0,0]\leftarrow) & \text{(VI)} \end{pmatrix}$$



Evaluation: usual measures



- **Model evaluation**:
  - Precision

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

  - Recall

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

  - F-measure $F = 2 \cdot \dfrac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$

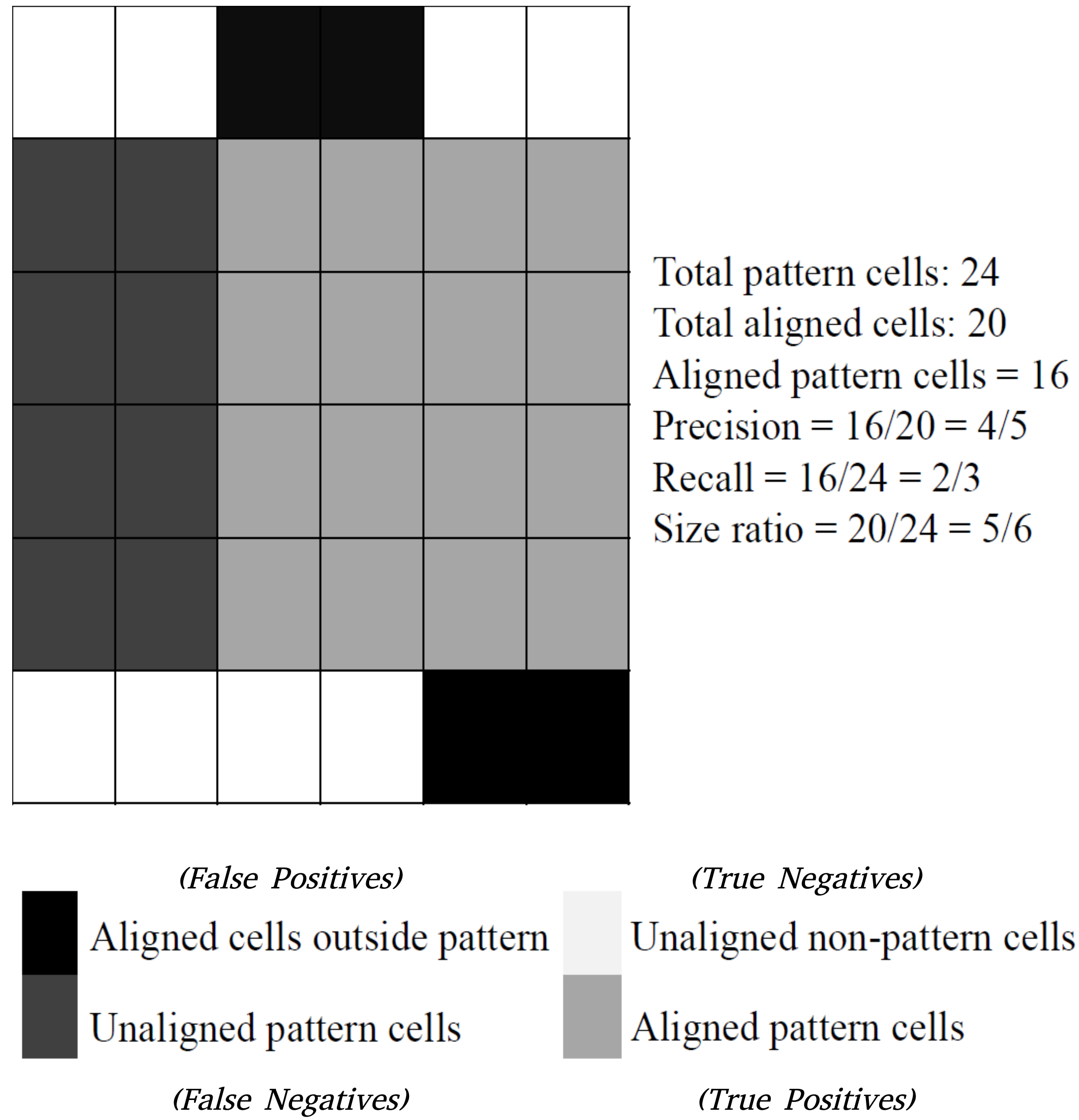
- **Remarks**:
  - Unbalanced classes!
  - Evaluation by class

https://en.wikipedia.org/wiki/Precision_and_recall

Routine extraction by matrix alignment

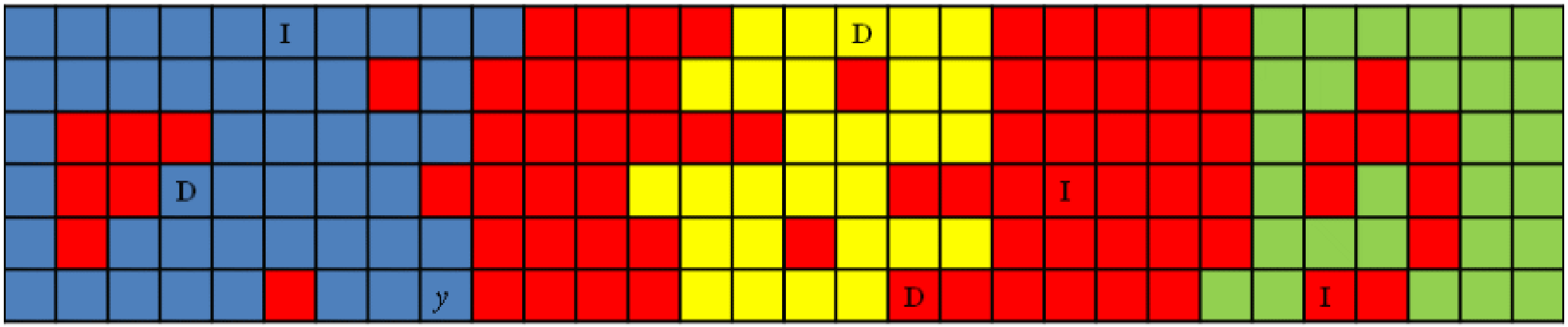
Evaluation

- Data and ground truth
- Computation time
- Precision and recall for each pattern alignment
- Alignment size
- Number of alignments

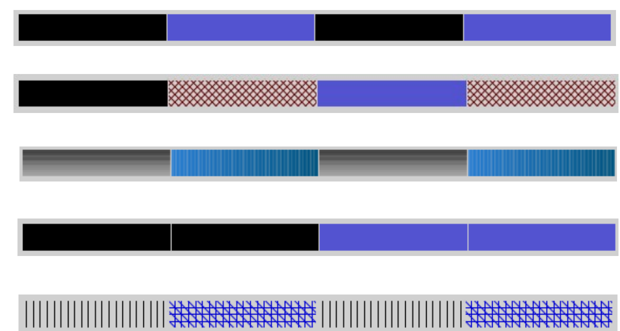


Routine extraction by matrix alignment

- Synthetic data generation



- Scenarios:
  1) Regular activity, no noise
  2) Regular activity, noise between patterns
  3) Noisy patterns
  4) Irregular pattern apparition
  5) Faulty sensors (¾ random data)



Measure	Dataset 1	Dataset 2	Dataset 3	Dataset 4	Dataset 5
$\frac{\#\text{alignments}}{\#\text{pairs of pat.}}$	0.31 ± 0.26	0.69 ± 0.29	0.041 ± 0.054	0.31 ± 0.26	0.22 ± 0.32
precision	0.54 ± 0.22	0.77 ± 0.20	0.13 ± 0.06	0.52 ± 0.24	0.54 ± 0.18
recall	1.00 ± 0.00	1.00 ± 0.00	0.56 ± 0.19	1.00 ± 0.01	0.20 ± 0.06
$\frac{\text{alignment size}}{\text{pattern size}}$	5.66 ± 4.48	2.45 ± 2.51	6.72 ± 3.96	6.65 ± 6.08	1.60 ± 0.24

Exercise

- Formalize the problem of behavior pattern extraction from discrete sequences of set of actions, based on a similarity approach (looking for longest and more similar behaviors)

Discrete sequences of item-sets
Pattern extraction
Frequency-based approach

Data-mining sequences of item-sets

- Frequent item-sets
  - Ex: *users usually select A & B keys*
  - Association rules (‘A’ => ‘B’ in the same item-set)
  - Support: how frequently an item-set appears in the dataset
  - Confidence: how often a rule has been found to be true
- Frequent sequential patterns
  - Ex: *users that perform action ‘A’, often perform action ‘B’ shortly after*
  - Association rules (‘A’ => ‘B’ in two following item-sets)
  - Gap: item-sets appearing inside the association rules

Definitions

- **Item**: minimal element
- **Item-set**: ordered (preference, …) set of items
- **Sequence of item-sets**: ordered set of item-sets
- **Transaction**: tuple in the database; a set of items or a sequence of item-sets.
  - training set = set of transactions
  - $\{\{a_{1,1}, \ldots, a_{1,m1}\}, \{a_{2,1}, \ldots, a_{2,m2}\}, \ldots, \{a_{n,1}, \ldots, a_{n,mn}\}\}$
  - Or a sparse matrix…
- **Association rule**: application $X \rightarrow Y$ where X and Y are disjoint set of items or set of item-sets

Evaluation of association rules ($X \rightarrow Y$)

- **Support**: absolute probability $P(X \cup Y)$

  $||X \cup Y|| / ||BD||$ = % of transactions satisfying the rule

- **Confidence**: conditional probability P(Y/X)

  $||X \cup Y|| / ||X||$ = % of transactions

  verifying the implication

  = support(XY)/support(X)

- An interesting association rule is a rule whose Confidence > Minconf and Support > Minsup

Apriori algorithm (Agrawal & Srikant, 1994)

- **Idea**: if an item-set is not frequent, then all its supersets are not frequent:
  - If {A} is not frequent then {AB} cannot be frequent
  - if {AB} is frequent then {A} and {B} are frequent

- **Process**:

  1) Generate iteratively candidate item-sets:
     - First pass: search for frequent 1-sets
     - Generate a candidate of size k from two candidates of size k-1 differentiated by the last element
     - Filter the sets of items with minimum support (keeping frequent item-sets)
  2) Use frequent item-sets to generate association rules

Apriori

Apriori

Input: L_1={frequent 1-itemsets};

Output: L_k={frequent k-itemsets};

for (k=2; $L_{k-1} \neq \emptyset$; k++) do {

  C_k=apriori-gen(L_{k-1});

    // Generate new candidates

  }

  L_k={ $c \in C_k$ | numberOf(c,DB) >= minsup } ;

    // Filter candidates

}

return L_k;

Apriori-gen

Input: L_{k-1}={frequent (k-1)-itemsets};

Items of L_{k-1} are ordered lexicographically

Output: C_k={candidates frequent (k)-itemsets};

- Step 1: Self-join on L_{k-1}

For each (p_{k-1},q_{k-1}) so that p_{k-1}<q_{k-1} do {

  C_k={ c_k | lexicographically ordered combinaison of p_{k-1} and q_{k-1}}

}

- Step 2: Pruning

foreach c_k in C_k do {

  foreach (k-1)-subsets t_{k-1} of C_k do {

    if (t_{k-1} is not in L_{k-1}) then

      delete t_{k-1} from C_k

  }

}

Apriori (exemple)

min_support=2

base D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

Scan D

C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

Scan D

C_2

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

L_2

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C_3

itemset
{2 3 5}

Scan D

L_3

itemset	sup
{2 3 5}	2

Apriori (generating association rules)

//Input: MinConf, L_k (frequent item-sets)
//Output: R, set of association rules

```
R = ∅ ;
foreach subsets S≠∅,  S≠Lk of Lk do {
    Confidence = Sup(S(Lk-S))/Sup(S)
    If Confidence >= MinConf then {
      R = R ∪ {“ S → Lk-S ”} ;
    }
  }
}
return R ;
```

Example :

$\{2\ 3\} \rightarrow \{5\}$ confidence=2/2
$\{2\ 5\} \rightarrow \{3\}$ confidence=2/3
…
$\{2\} \rightarrow \{3\ 5\}$ confidence=2/3
...

Exercise

- Formalize the problem of behavior pattern extraction from discrete sequences of set of actions, based on a frequency approach (looking for most frequent behaviors)

Continuous signal(s)

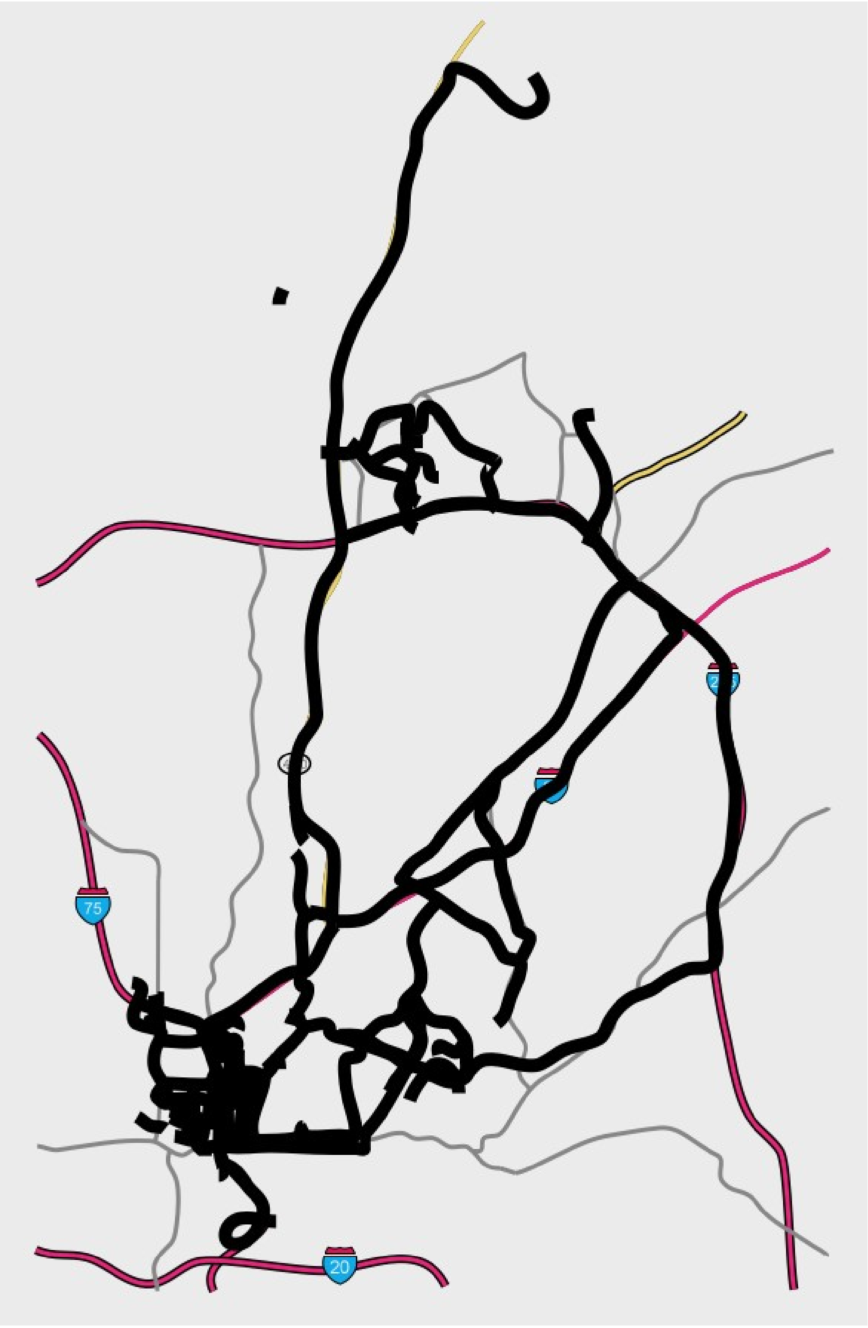
(Single) continuous signal

- **Continuous numeric acquisition is impossible**
  - Discretize an analogical signal to numerical values along 2 dimensions:
    - Continuous / discrete (alphabet)
    - Time (sampling)

      NB: a re-sampling can be necessary according to the goal

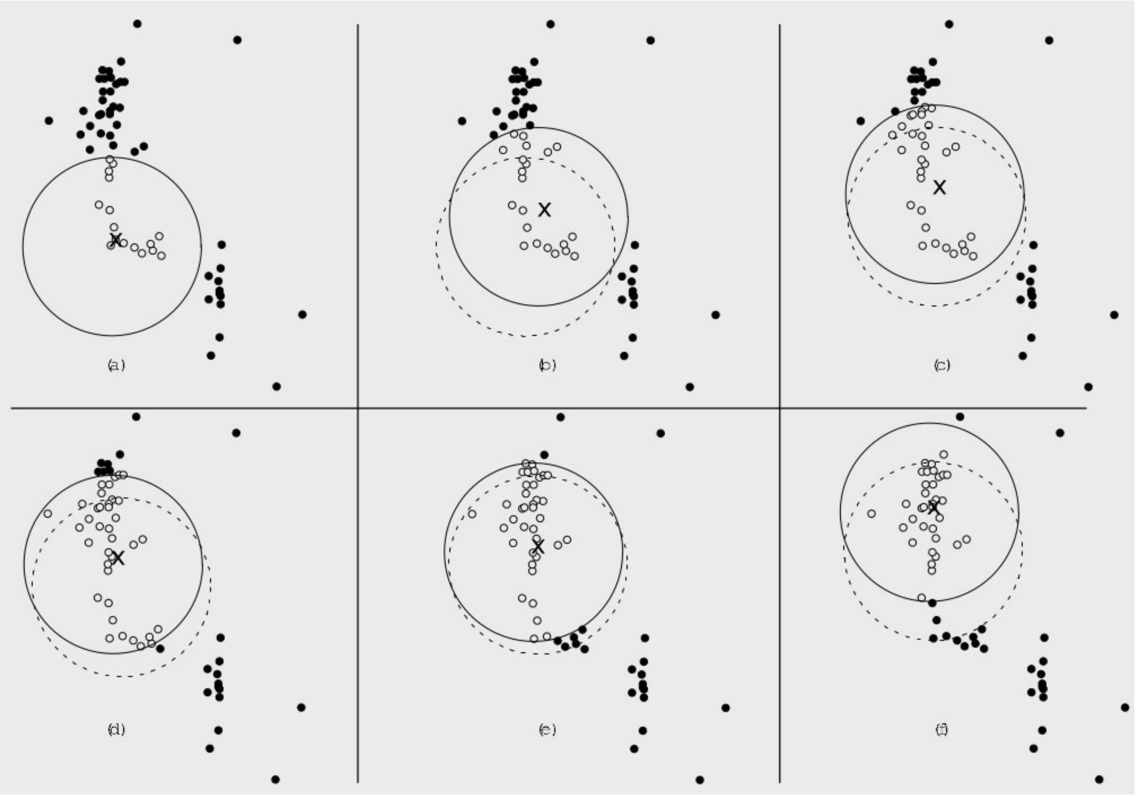
- **Discretization process**
  - Heterogeneous sensors (unnormalized)
  - Semantic information (analyze & user’s feedback)

Example: GPS signal

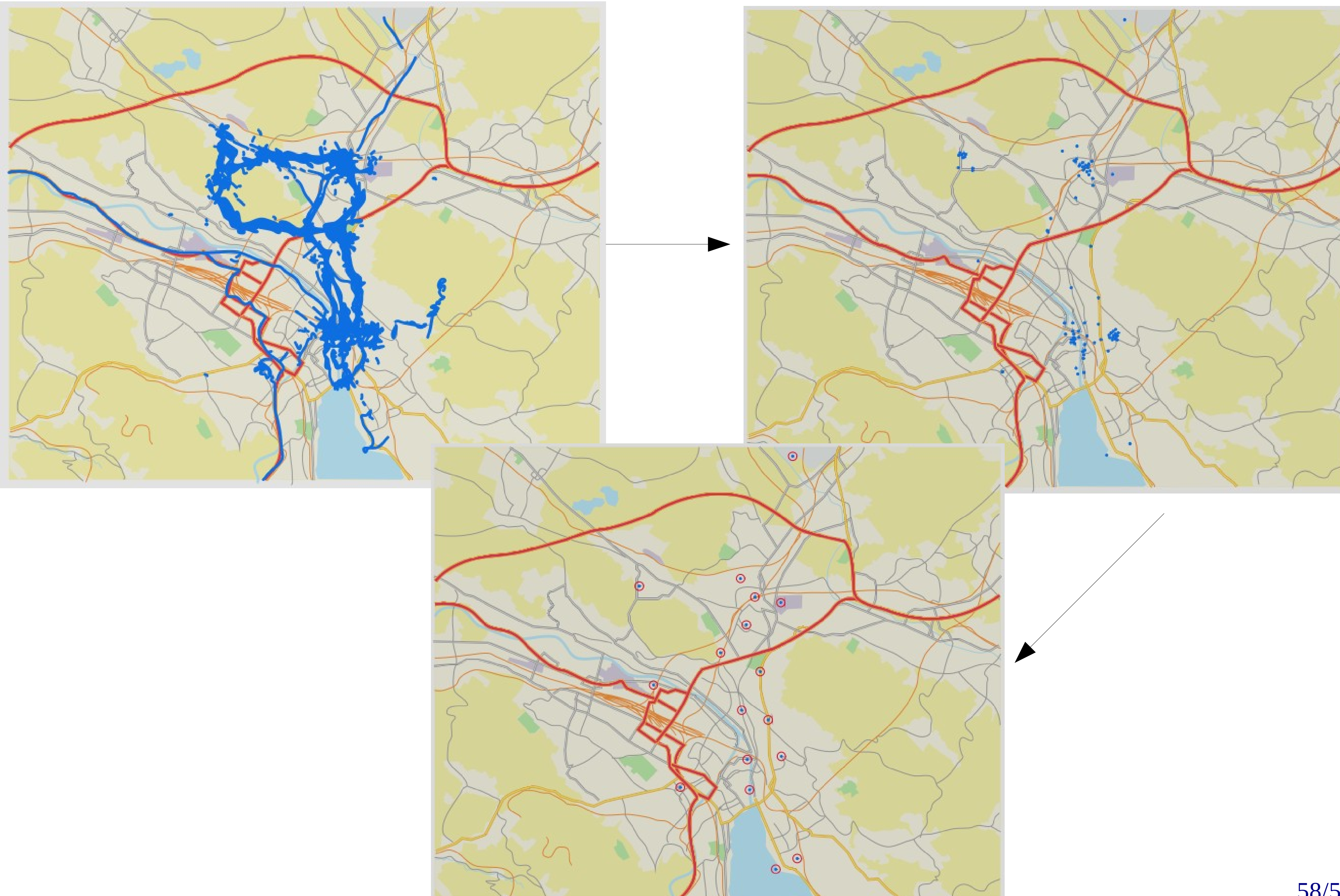


Daniel Ashbrook & Thad Starner : « Using GPS to learn significant locations and predict movement across multiple users », Personal and Ubiquitous Computing, volume 7, number 5, pp. 275-286, Springer, 2003.

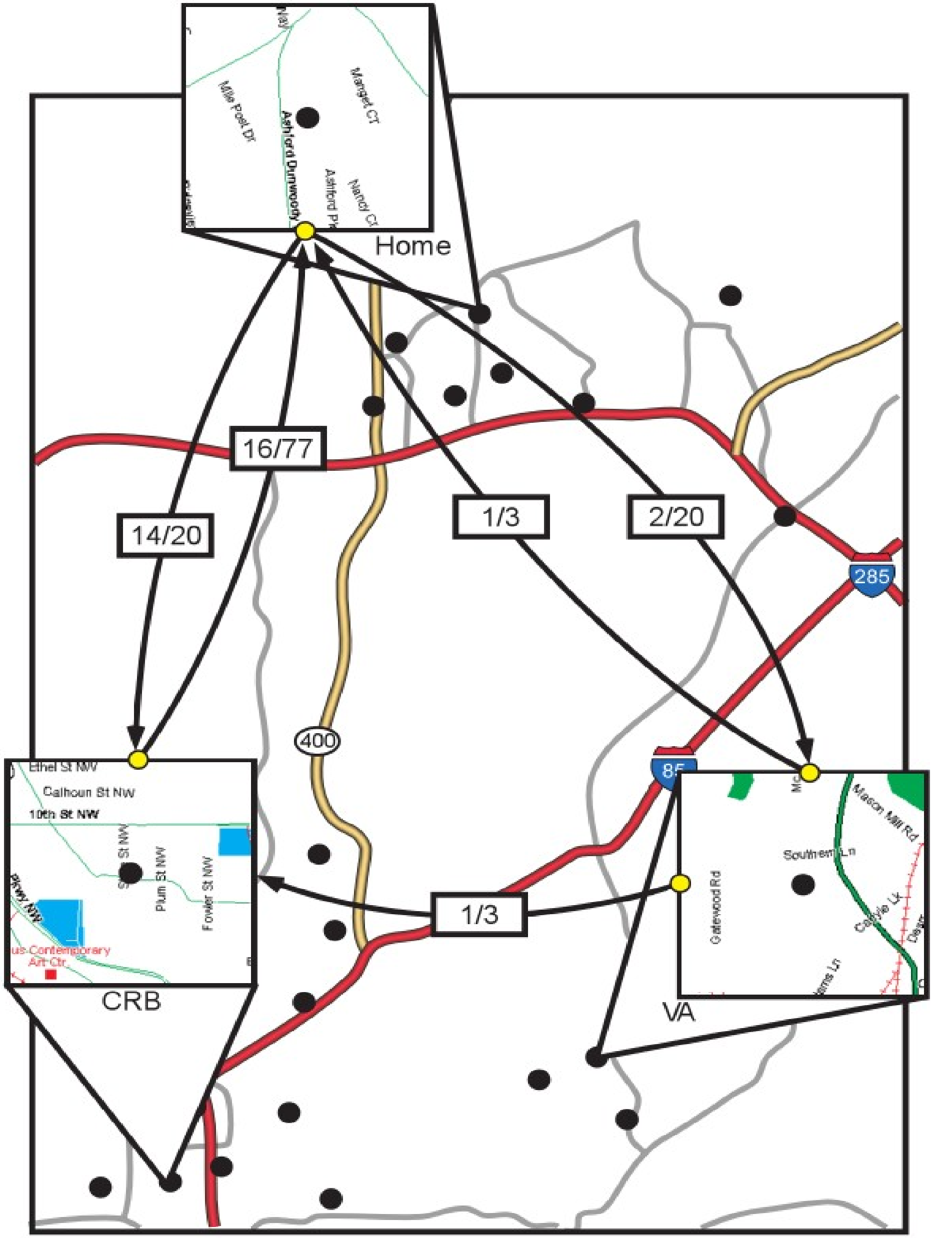
- Finding **significant places / positions** (time dependence: time threshold)
- Clustering places into **locations / keypoints** (spacial dependence: cluster radius)



Example: GPS signal



Example: GPS signal



Transition	Relative Frequency	Probability
$A \to B$	$14/20$	0.7
$A \to B \to A$	$3/14$	0.2142
$A \to B \to C$	$2/14$	0.1428
$A \to B \to D$	$3/14$	0.2142
$A \to B \to E$	$1/14$	0.0714
$A \to B \to F$	$1/14$	0.0714
$A \to B \to G$	$1/14$	0.0714
$A \to B \to H$	$1/14$	0.0714
$A \to B \to I$	$1/14$	0.0714
$B \to A$	$16/77$	0.2077
$B \to A \to B$	$13/16$	0.8125
$B \to A \to J$	$3/16$	0.1875
$B \to C$	$10/77$	0.1298
$B \to C \to A$	$6/10$	0.6
$B \to C \to K$	$4/10$	0.4
$D \to B$	$5/7$	0.7142
$D \to B \to A$	$2/5$	0.4
$D \to B \to L$	$2/5$	0.4
$D \to B \to M$	$1/5$	0.2

Probabilities for transitions in Markov models
Key: A = “Home”

Ashbrook & Starner, 2002

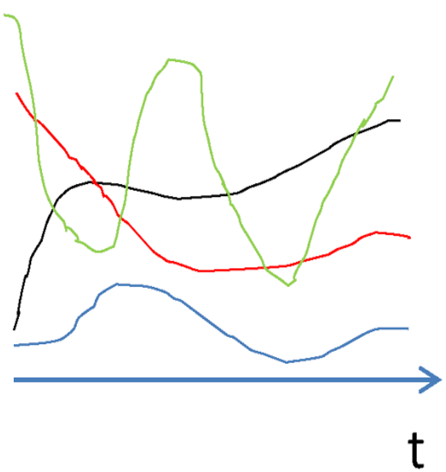
Exercise

- Formalize the problem of finding keypoints from GPS data?
- Formalize the problem of behavior pattern extraction from a sequence of visited places? From GPS data?

(Multiple) Continuous signals

$C = C(t)$

$\mathbb{R} \rightarrow \mathbb{R}^n$



discretisation

$C = (c_1(t_1), c_2(t_2), \ldots, c_n(t_n))$

$\mathbb{N}^n \rightarrow \mathbb{N}^n$

1.1	0.3	0.2	0.8
0.7		0.2	
0.4	0.5	0.3	
0.4	0.6	0.4	
0.6		0.5	0.5
0.8	0.3	0.2	
0.5	0.6	0.2	

t

synchronisation and classification

$C = (c_1, c_2, \ldots, c_n)(t)$

$\mathbb{N} \rightarrow \mathbb{N}^n$

A_1	A_2	A_3	A_4
B_1	A_2	A_3	A_4
C_1	B_2	A_3	A_4
C_1	B_2	B_3	A_4
B_1	B_2	B_3	B_4
B_1	A_2	A_3	B_4
C_1	B_2	A_3	B_4

t

Multiple sensors: (discrete) approaches