Analysis of Patterns
–
Introduction to Pattern Discovery

Florent Nicart

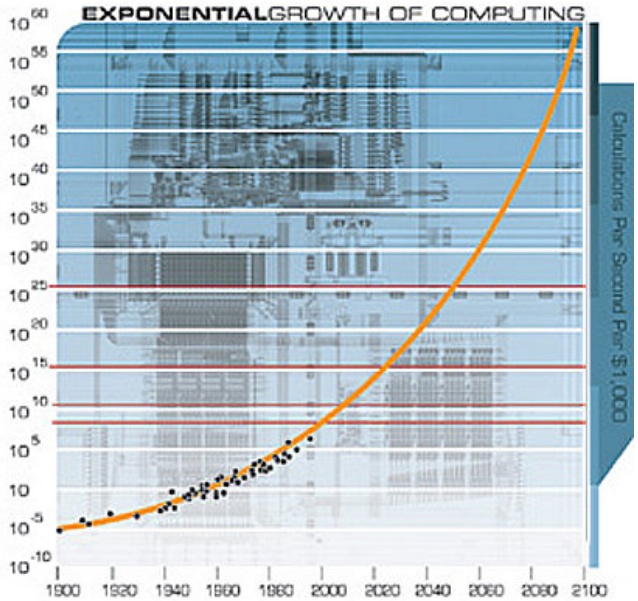
University of Rouen

September 29th, 2009

Outline

1. Introduction
   - Why Pattern Discovery ?
   - Data sets
   - Pattern Matching vs Pattern Discovery
2. Association Patterns
   - Frequent Set Mining
   - Association Rules Mining
3. Sequential Patterns
   - Sequences
   - Languages

Why Pattern Discovery ?



- Increase of data storage capacity,
- increase of computational power,
- inter-disciplinary techniques,
- objective : to understand the underlying process generating the dataset.

Availability of huge datasets

Databases are everywhere

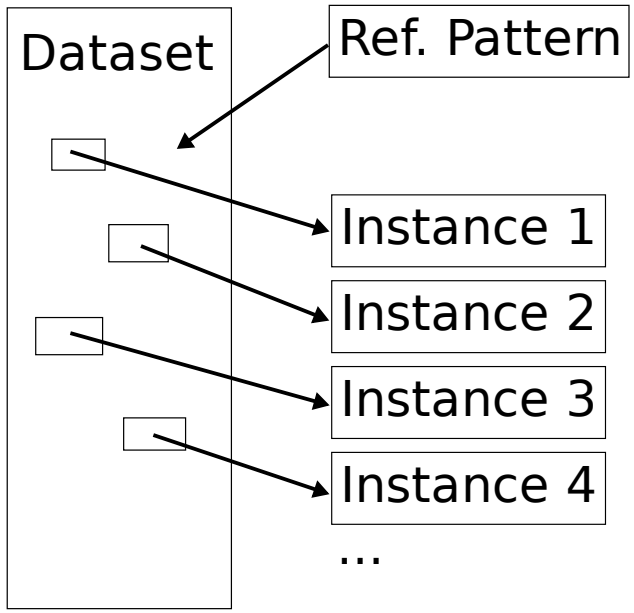
- Supermarket transactions,
- credit card records,
- telephone call details,
- weblogs, ISP logs,
- genetic databases.

What is a pattern ?

Pattern matching : a collection of similar values

Pattern discovery : a particuliar combination of values

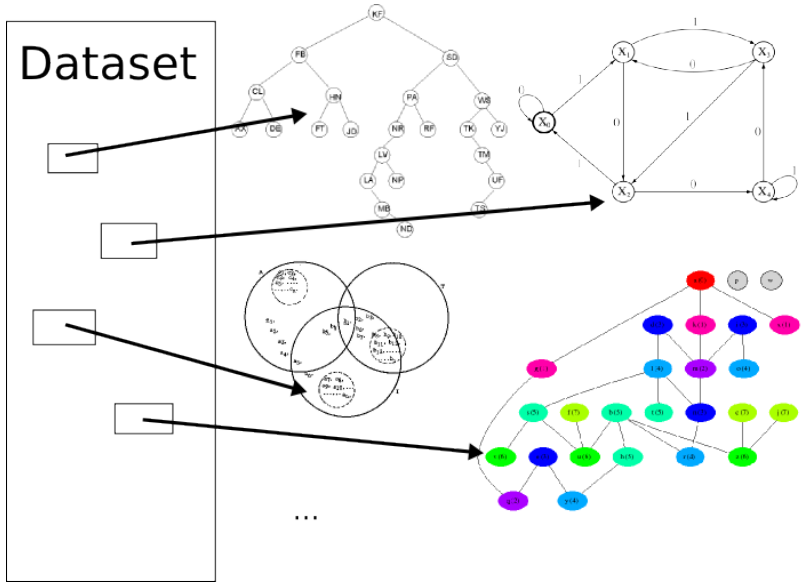
Pattern Matching vs Pattern Discovery



Pattern Matching

- a dataset,
- a pattern,
- a metric and a distance.

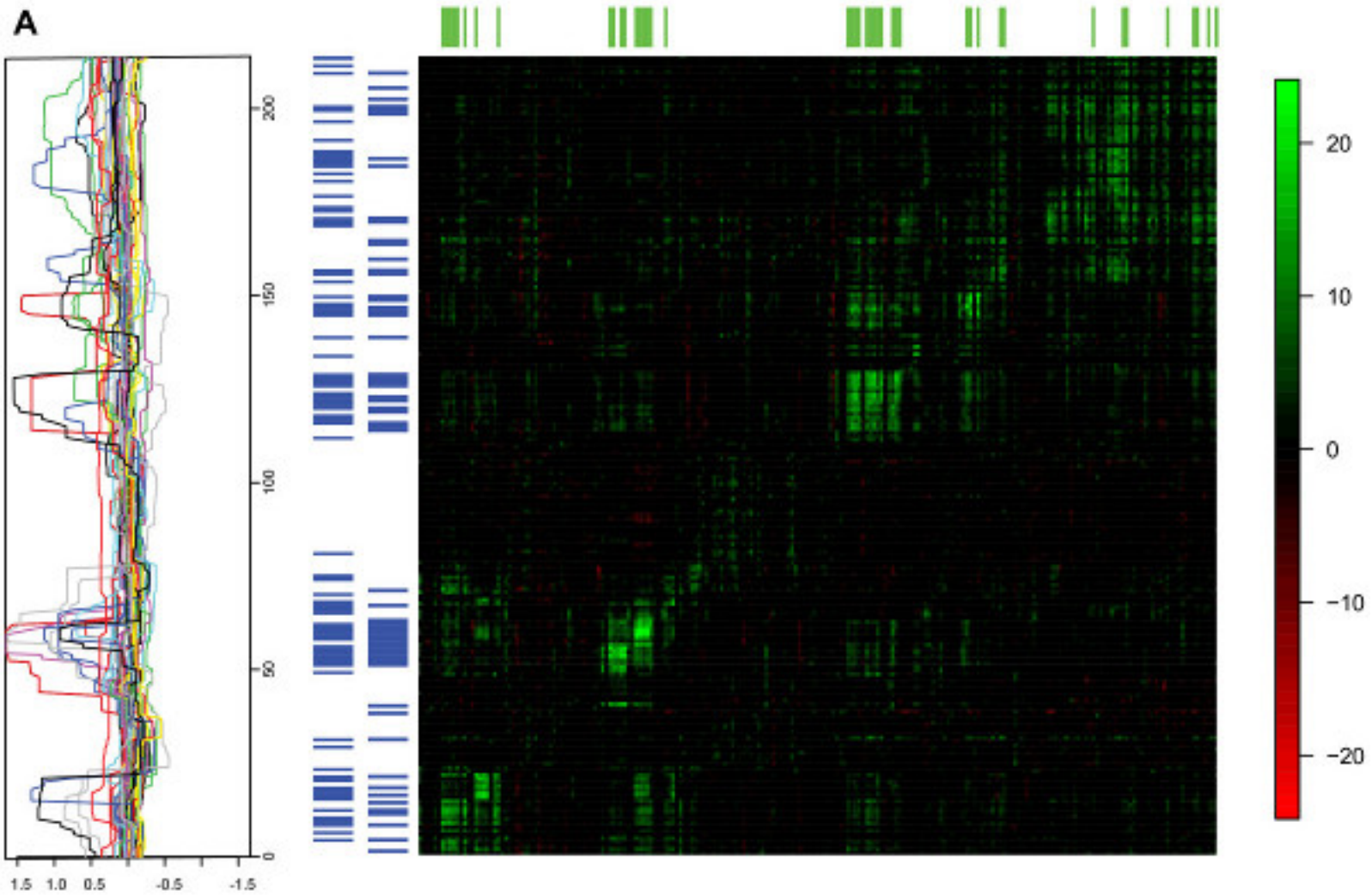
Pattern Matching vs Pattern Discovery



Pattern Discovery

- a dataset,
- a metric and a distance
- "a pattern space"

Association Patterns



The pattern :

Finding (subsets) events that occur together [Agrawal93].

Rakesh Agrawal and Ramakrishnan Srikant.
Fast Algorithms for Mining Association Rules in Large Databases,
Proceedings of 20th International Conference on Very Large Data Bases, 1994.

Typical application (origin) :

Market basket analysis :

- "10% of customers are buying wine and cheese",
- "15% are buying crisps and beer",

and the legendary[a] :

- "People buying beer on saturday are very likely to buy nappies" (Wal-Mart).

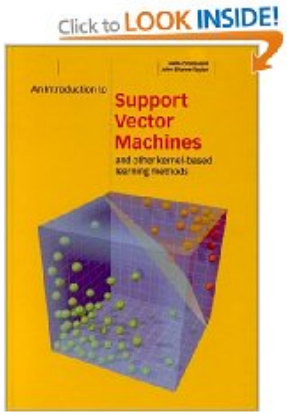
---

[a] Beer and Nappies – A Data Mining Urban Legend
http://web.onetel.net.uk/~hibou/Beer and Nappies.html

Example (Amazon associations)

amazon.co.uk

Join Amazon Prime and get unlimited Free One-Day Delivery. Already a member? Sign in.



Share your own customer images
Search inside this book

An Introduction to Support Vector Machines and Other Kernel-based Learning Methods (Hardcover)
by Nello Cristianini (Author), John Shawe-Taylor (Author)
(1 customer review)

RRP: ~~£41.00~~
Price: £38.95 & this item **Delivered FREE in the UK** with Super Saver Delivery. See details and conditions
You Save: £2.05 (5%)

In stock.
Dispatched from and sold by **Amazon.co.uk**. Gift-wrap available.

Want guaranteed delivery by Tuesday, August 25? Order it in the next 3 hours and 55 minutes, and choose **Express** delivery at checkout. See Details

25 new from £32.80 **6 used** from £29.52

Quantity: 30
Add to Shopping Basket
or
Sign in to turn on 1-Click ordering.

More Buying Choices
31 used & new from £29.52
Have one to sell? Sell yours here

Add to Wish List
Add to Wedding List

Frequently Bought Together

Price For All Three: £141.54
Add all three to Basket
Show availability and shipping details

- **This item:** An Introduction to Support Vector Machines and Other Kernel-based Learning Methods by Nello Cristianini
- Kernel Methods for Pattern Analysis by John Shawe-Taylor
- Pattern Recognition and Machine Learning (Information Science and Statistics) (Information Science and Statistics) by Christopher M. Bishop

Association Patterns
Introduction

Other typical applications :

- Financial-services/telecommunications companies : to which combination of services customers most often subscribe,
- "Quality-assurance : which combinations of components are most likely to fail at the same time,
- Web log analysis,
- Finding Association Rules,
- Classification, ...

Frequent Set Mining
Defining the problem

The input data

- $\mathcal{I} = \{i_1, i_2, \ldots, i_n\}$, set of items,
- itemset S : any subset[a] of $\mathcal{I}$,
- transaction : $t = \langle tid, S_{tid} \rangle$, $tid \in \mathbb{N}$ and $S_{tid} \subseteq \mathcal{I}$
- $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$, set of transactions,

[a] Usually excluding the empty set

Common itemsets

$t \cap t' = S \cap S'$ with $t = \langle tid, S \rangle$ and $t' = \langle tid', S' \rangle$

$common(t, t') = 2^{t \cap t'} \setminus \emptyset$

Example (Transactions and itemsets)

basket	P_1	P_2	P_3	P_4	P_5	I_{tid}
t_1	1	0	0	0	0	$\{P_1\}$
t_2	1	1	1	1	0	$\{P_1, P_2, P_3, P_4\}$
t_3	1	0	1	0	1	$\{P_1, P_3, P_5\}$
t_4	0	0	1	0	0	$\{P_3\}$
t_5	0	1	1	1	0	$\{P_2, P_3, P_4\}$
t_6	1	1	1	0	0	$\{P_1, P_2, P_3\}$
t_7	1	0	1	1	0	$\{P_1, P_3, P_4\}$
t_8	0	1	1	0	1	$\{P_2, P_3, P_5\}$
t_9	1	0	0	1	0	$\{P_1, P_4\}$
t_{10}	0	1	1	0	1	$\{P_2, P_3, P_5\}$

$common(t_2, t_3) = \{\{P_1\}, \{P_3\}, \{P_1, P_3\}, \}$

Frequent Set Mining
Defining the problem

Frequent Itemset Mining (FIM)

- Find the most frequent itemsets that occur amongst all the transactions of $\mathcal{T}$.
- The pattern space is $2^{\mathcal{I}}$

Complexity

- Size of the space of patterns : $2^{|\mathcal{I}|}$

example	#items sold	$2^{\lvert\mathcal{I}\rvert}$
Small shops	5000	$1,4 * 10^{1505}$
ebay.com	30.147.410	$5 * 10^{9075274}$
amazon(+third)	+5.000.000	$9,5 * 10^{1505149}$
ebay.com(cat)	20.000	$4 * 10^{6020}$

Frequent Set Mining
Definitions

Definition (Support Count of an itemset)

The *Support count* (or simply count) of an itemset is the number of transactions where it occurs.
$Count(S) = |\{< tid, S_{tid} > |\ S \subseteq S_{tid}\}|$

Definition (Support of an itemset)

The *Support* of an itemset is the proportion of transactions where it occurs. $Support(S) = Count(S)/n$

Definition (Minimal support)

minsup is the minimal support of itemset to be considered.

Definition (Supported itemsets)

S is a supported itemset iff $Support(S) >= minsup$.

Frequent Set Mining
Definitions

Example (Supported itemsets)

basket	P_1	P_2	P_3	P_4	P_5	I_{tid}
t_1	1	0	0	0	0	$\{P_1\}$
t_2	1	1	1	1	0	$\{P_1, P_2, P_3, P_4\}$
t_3	1	0	1	0	1	$\{P_1, P_3, P_5\}$
t_4	0	0	1	0	0	$\{P_3\}$
t_5	0	1	1	1	0	$\{P_2, P_3, P_4\}$
t_6	1	1	1	0	0	$\{P_1, P_2, P_3\}$
t_7	1	0	1	1	0	$\{P_1, P_3, P_4\}$
t_8	0	1	1	0	1	$\{P_2, P_3, P_5\}$
t_9	1	0	0	1	0	$\{P_1, P_4\}$
t_{10}	0	1	1	0	1	$\{P_2, P_3, P_5\}$

$Support(\{P_1, P_3\}) = |\{t_2, t_3, t_6, t_7\}|/10 = 0.4$
$Support(\{P_2, P_4\}) = |\{t_2, t_5\}|/10 = 0.2$
with $minsup = 0.25$, $\{P_1, P_3\}$ is supported, $\{P_2, P_4\}$ is not.

Frequent Set Mining
Definitions

Theorem (Downward closure property/Support monotonicity.)

$S \subseteq S' \equiv Support(S) >= Support(S')$

By corollary, if L_k is the set of supported sets with cardinality k :

Theorem

$L_k = \emptyset \Rightarrow \forall i > k, L_i = \emptyset$

Frequent Set Mining
The Apriori algorithm

The Apriori algorithm

$L_1 \leftarrow$ supported itemsets of cardinality one
$k \leftarrow 2$
while $(L_{k-1} = \emptyset)$ {
    $C_k \leftarrow create\_candidates(L_{k-1})$
    $L_k \leftarrow prune(C_k)$
    $k \leftarrow k + 1$
}
return $L_1 \cup L_2 \cup \ldots L_k$

Generating C_k from L_{k-1}

Join Step

Compare each member of L_{k-1} , say A, with every other member, say B, in turn. If the first $k-2$ items in A and B (i.e. all but the rightmost elements of the two itemsets) are identical, place set $A \cup B$ into C_k.

Prune Step

For each member c of C_k in turn $\{$

Examine all subsets of c with $k-1$ elements

Delete c from C_k if any of the subsets is not a member of L_{k-1}

$\}$

Frequent Set Mining

The Apriori algorithm : an example

Example (Joint step for C_5)

$L_4 = \{\{p,q,r,s\},\{p,q,r,t\},\{p,q,r,z\},\{p,q,s,z\},$
$\{p,r,s,z\},\{q,r,s,z\},\{r,s,w,x\},\{r,s,w,z\},\{r,t,v,x\},$
$\{r,t,v,z\},\{r,t,x,z\},\{r,v,x,y\},\{r,v,x,z\},\{r,v,y,z\},$
$\{r,x,y,z\},\{t,v,x,z\},\{v,x,y,z\}\}$

l_1	l_2	Contribution to C'_5
$\{p,q,r,s\}$	$\{p,q,r,t\}$	$\{p,q,r,s,t\}$
$\{p,q,r,s\}$	$\{p,q,r,z\}$	$\{p,q,r,s,z\}$
$\{p,q,r,t\}$	$\{p,q,r,z\}$	$\{p,q,r,t,z\}$
$\{r,s,w,x\}$	$\{r,s,w,z\}$	$\{r,s,w,x,z\}$
$\{r,t,v,x\}$	$\{r,t,v,z\}$	$\{r,t,v,x,z\}$
$\{r,v,x,y\}$	$\{r,v,x,z\}$	$\{r,v,x,y,z\}$

$\rightarrow C'_5 = \{\{p,q,r,s,t\},\{p,q,r,s,z\},\{p,q,r,t,z\},\{r,s,w,x,z\},$
$\{r,t,v,x,z\},\{r,v,x,y,z\}\}$

Frequent Set Mining

The Apriori algorithm : an example

Example (Pruning step for C_5)

$L_4 = \{\{p,q,r,s\},\{p,q,r,t\},\{p,q,r,z\},\{p,q,s,z\},$
$\{p,r,s,z\},\{q,r,s,z\},\{r,s,w,x\},\{r,s,w,z\},\{r,t,v,x\},$
$\{r,t,v,z\},\{r,t,x,z\},\{r,v,x,y\},\{r,v,x,z\},\{r,v,y,z\},$
$\{r,x,y,z\},\{t,v,x,z\},\{v,x,y,z\}\}$

$C'_5 = \{\{p,q,r,s,t\},\{p,q,r,s,z\},\{p,q,r,t,z\},\{r,s,w,x,z\},$
$\{r,t,v,x,z\},\{r,v,x,y,z\}\}$

Itemset in C'_5	Subsets all in L_4 ?
$\{p,q,r,s,t\}$	No : $\{p,q,s,t\} \notin L_4$
$\{p,q,r,s,z\}$	Yes
$\{p,q,r,t,z\}$	No : $\{p,q,t,z\} \notin L_4$
$\{r,s,w,x,z\}$	No : $\{r,s,x,z\} \notin L_4$
$\{r,t,v,x,z\}$	Yes
$\{r,v,x,y,z\}$	Yes

$C_5 = \{\{p,q,r,s,z\},\{r,t,v,x,z\},\{r,v,x,y,z\}\}$

Association Rules Mining
Introduction

- We would like to use the itemset found to express association rules...

Association Rules Mining
Introduction

Definition (Rule)

$L \rightarrow R$ with L (antecedant/condition) and R (consequence) being predicates.

- Reads L implies R,
- ex : "If it rains then the ground will be wet."
- $\{ab\} \rightarrow \{cd\}$ means $\{a, b\} \subseteq S \rightarrow \{c, d\} \subseteq S, \forall S \in \mathcal{T}$

Definition (Association Rule)

$L \rightarrow R$ with $L \neq \emptyset$ and $R \neq \emptyset$ being disjoint itemsets.

Association Rules Mining
Introduction

- Probabilistic *AR* : *"When beer and crisps are bought together, cheese is bought in 45% of cases"*,
- $L \rightarrow R$: $L \cup R$ is an itemset :

Definition (Support of an Association Rule)

$Support(L \rightarrow R) = Support(L \cup R) = Count(L \cup R)/n$

Definition (Confidence of an Association Rule)

$Confidence(L \rightarrow R) = Support(L \cup R)/Support(L) = Count(L \cup R)/Count(L)$

Association Rules Mining
Generating rules

- Only supported rules : $Support(L \rightarrow R) \geq minsup$,
- given $\mid L \cup R \mid = k$, then number of rules that can be generated :

$$\Sigma_{i=1}^{k-1} C_i^k = 2^k - 2$$

- Filtering rules : $Confidence(L \rightarrow R) \geq minconf$

Definition (Confident Right-hand side)

The right-hand side of a rule is said *confident* iff $Confidence(L \rightarrow R) \geq minconf$ and *unconfident* otherwise.

Association Rules Mining
Generating rules

theorem

$$Confidence(L \cup \{x\} \rightarrow R) \geq Confidence(L \rightarrow R \cup \{x\})$$

Corrolary

Any superset of an unconfident right-hand itemset is unconfident.
Any (non-empty) subset of a confident right-hand itemset is confident.

- Rules can be generated with an Apriori-like algorithm.

Sequential Patterns

- Temporal sequence data sets :
  - Supermaket customer transactions,
  - Weblogs : user navigation,
  - Alarm logs : fault analysis,
- Text sequence data sets :
  - Genomics,
  - documents,
  - etc.

Sequences

Finding sequential patterns in sequences of events :

- Data set : supermaket customer transactions,
- Pattern : customers renting "The Fellowship of the Ring" rent "The Two Towers" and then "The Return of the King"

Rakesh Agrawal and Ramakrishnan Srikant.
Mining Sequential Patterns,
Proceedings of 5th International Conference on Extending Database Technology, 1995.

Sequences
Definitions

Definition (Sequence)

$s = \langle s_1 \dots s_n \rangle$ where s_k is an itemset

Definition (Contained sequence)

A sequence $\langle a_1 \dots a_n \rangle$ is contained in a sequence $\langle b_1 \dots b_m \rangle$ if there exists integers $i_1 < i_2 < \dots < i_n$ such that $a_1 \subseteq b_{i_1}$, $a_2 \subseteq b_{i_2}$, ... $a_n \subseteq b_{i_n}$.

Examples

- $\langle \{3\}\{4,5\}\{8\} \rangle$ is contained in $\langle \{7\}\{3,8\}\{9\}\{4,5,6\}\{8\} \rangle$
- $\langle \{3\}\{5\} \rangle$ is not contained in $\langle \{3,5\} \rangle$

Sequences
Examples

Customer Id	Transaction Time	Items Bought
1	June 25 '93	30
1	June 30 '93	90
2	June 10 '93	10, 20
2	June 15 '93	30
2	June 20 '93	40, 60, 70
3	June 25 '93	30, 50, 70
4	June 25 '93	30
4	June 30 '93	40, 70
4	July 25 '93	90
5	June 12 '93	90

Customer Id	Customer Sequence
1	$\langle\{30\}\{90\}\rangle$
2	$\langle\{10,20\}\{30\}\{40,60,70\}\rangle$
3	$\langle\{30,50,70\}\{30\}\{40,70\}\{90\}\rangle$
5	$\langle\{90\}\rangle$

Sequences
Sequence Support

Definition (Sequence Support)

Let $\mathcal{T}_i$ be the set of transactions of customer i,
$Support(s) = |\{i \mid s \subseteq \mathcal{T}_i\}|$

Definition (Maximal sequence)

In a set of sequences, a sequence is *maximal* if it is not contained in any other sequence.

Pattern discovery task :

Find all maximal sequences (*Sequential Patterns*) in a set of customer transactions having a support greater than *minseqsup*.

Sequences
Sequence Support

Customer Id	Customer Sequence
1	$\langle\{30\}\{90\}\rangle$
2	$\langle\{10,20\}\{30\}\{40,60,70\}\rangle$
3	$\langle\{30,50,70\}\{30\}\{40,70\}\{90\}\rangle$
5	$\langle\{90\}\rangle$

Sequential Patterns ($minseqsup = 0.25$)
$\langle\{30\}\{90\}\rangle$
$\langle\{30\}\{40,70\}\rangle$

Sequences $\langle\{30\}\rangle$, $\langle\{40\}\rangle$, $\langle\{70\}\rangle$, $\langle\{90\}\rangle$, $\langle\{30\}\{40\}\rangle$, $\langle\{30\}\{70\}\rangle$, $\langle\{40,70\}\rangle$ are not maximal.

Algorithm *Agrawal et al*

- Sorting phase : (customer id, transaction time)
- Customer-supported itemset phase : customer count based Apriori
- Transformation phase : transforms the customer transaction to accelerate the containing tests
- Sequence phase : find supported sequences from (customer)supported itemsets (AprioriAll, AprioriSome/DynamicSome)
- Maximal phase : keep only the maximal supported sequences.

Sequences
Episode Mining

Finding *episodes* in sequences of events :

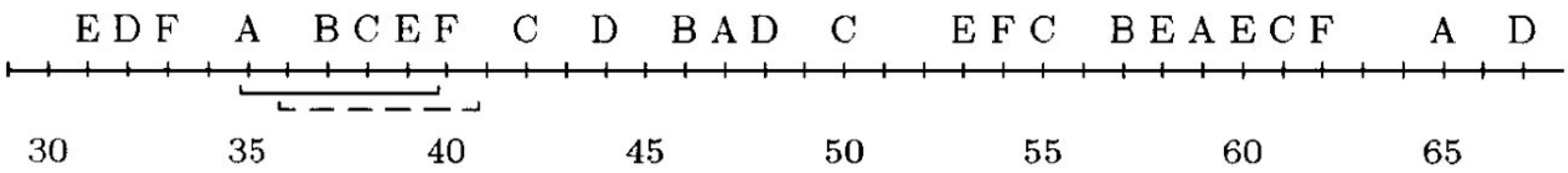
- Data set : set of timed events,
- Pattern : groups of events occuring frequently *close* to each other

Heikki Mannila, Hannu Toivonen and A. Inkeri Verkamo, Discovery of Frequent Episodes in Event Sequences, *Data Mining and Knowledge Discovery*, 1997.

Sequences
Event sequences

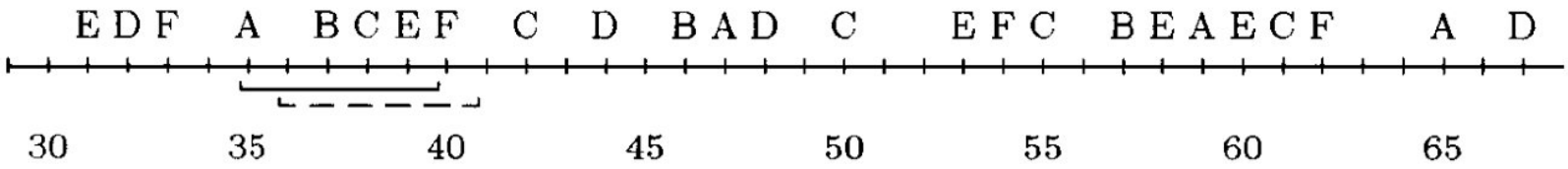
Definition (event)

Let E be a set of event types, an event is a pair (A, t) where $A \in E$ and t is an integer (timestamp).



$$s = \langle (E, 31), (D, 32), (F, 33), (A, 35), (B, 37), \ldots, (D, 67) \rangle$$

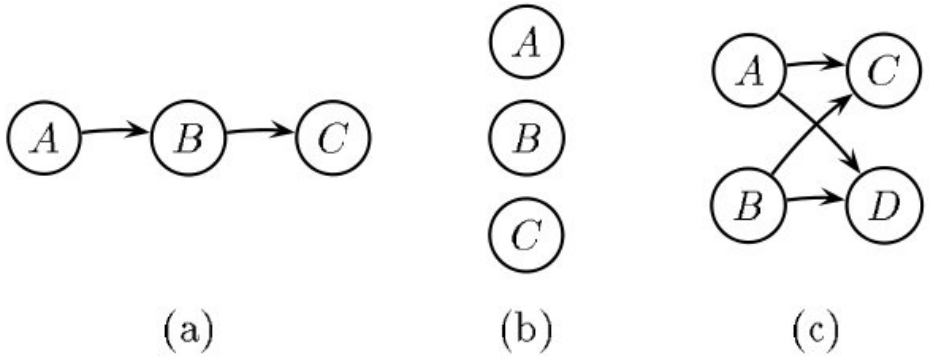
Sequences
Episodes



$$s = \langle (E, 31), (D, 32), (F, 33), (A, 35), (B, 37), \ldots, (D, 67) \rangle$$

Definition (episode)

An *episode* is a partially ordered subset of events.

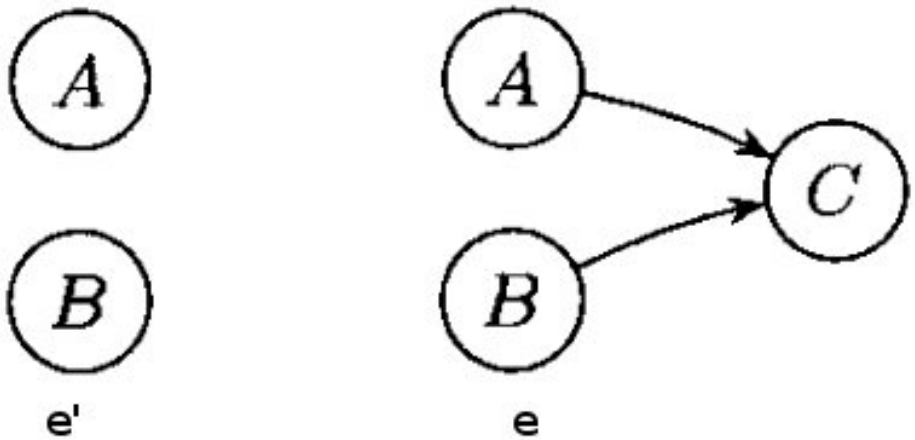


$$e = (V, \leq, g) \ , \ g : V \rightarrow E$$

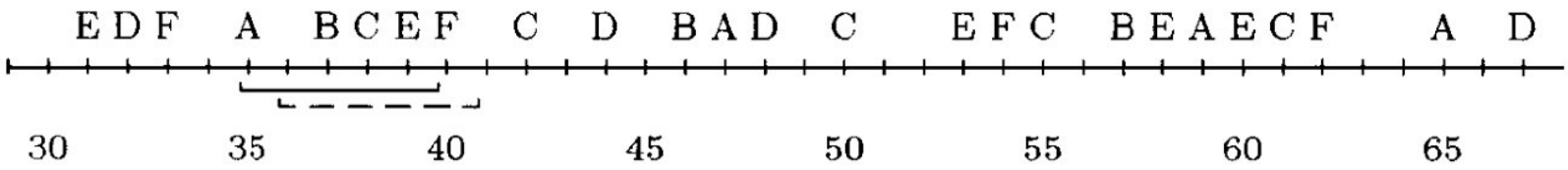
Sequences
Subepisodes

Definition (Subepisode)

An episode $e = (V', \leq', g')$ is a *subepisode* of $e = (V, \leq, g)$ if $V' \subset V$, $\forall v \in V', g'(v) = g(v)$, and $\forall v, w \in V', v \leq' w \rightarrow v \leq w$



Sequences
Mining episodes



Pattern discovery task

Given

- a window size,
- a minimal frequency,
- a class of episodes,

find all the frequent episodes in the class.

Note : frequency of an episode = amount of windows in which it occurs.

Sequences
Algorithm

Algorithm (*Mannila et al*)

$C_1 \leftarrow \{\{e\} \mid e \in E\}$
$k \leftarrow 1$
while $(C_k = \emptyset)$ {
    $L_k \leftarrow frequent(C_k, s, minfr, winsize)$
    $C_{k+1} \leftarrow generate(L_k, \mathcal{C})$
    $k \leftarrow k + 1$
}
return $L_1 \cup L_2 \cup \ldots L_k$

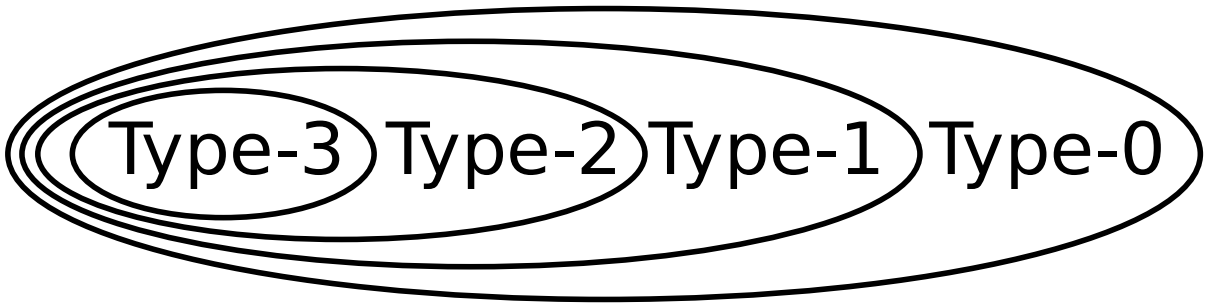
Languages

Discovering (rational) languages

Languages
What is Gramatical Inference ?

- A grammar describes a language/set of words (possibly infinite).
- Is it possible, from a finite sample (or a positive and negative sample) to guess the grammar ?
- Grammatical inference : find a description of the generating process.

Chomsky hierarchy :



Type	Grammar	Language	Machine
type-0	Unrestricted	Recurs. enumer.	Turing machine
type-1	Context-sensitive	Context-sensitive	Linear bounded
type-2	Context-free	Context-free	Pushdown aut.
type-3	***Regular***	***Regular***	***Finite aut.***

Languages
Definitions : regular language

Definition (Regular language)

Let Σ be a finite set of symbols (ex : $\Sigma = \{a, b, c\}$), a regular language $L \subseteq \Sigma^*$ recursively defined by mean of *concatenations, unions* and *stars*

Example (Regular language)

$L = a \cdot b^* \cdot c = \{ac, abc, abbc, abbbc, \ldots\}$

Theorem (Kleene)

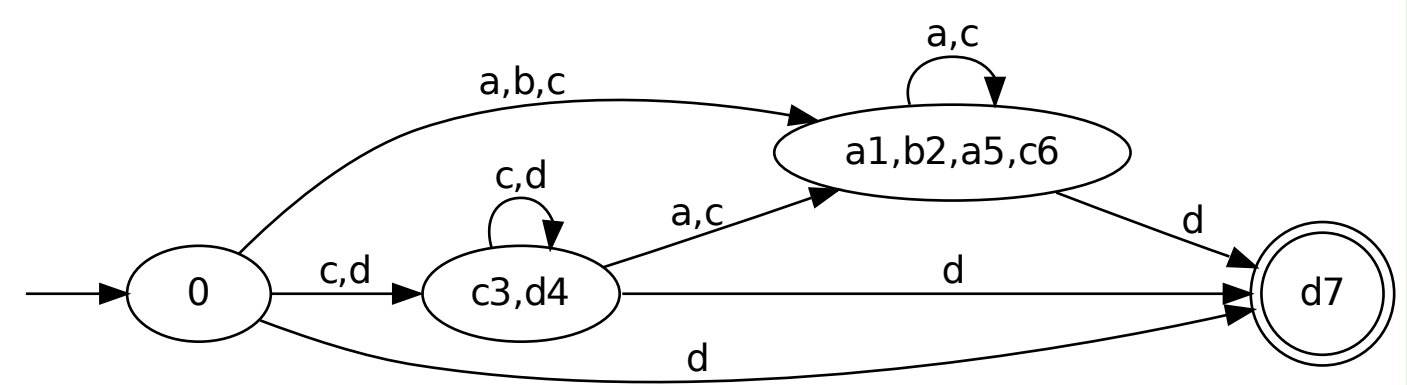
$$Rat = Rec$$

Definition (Finite state automaton)

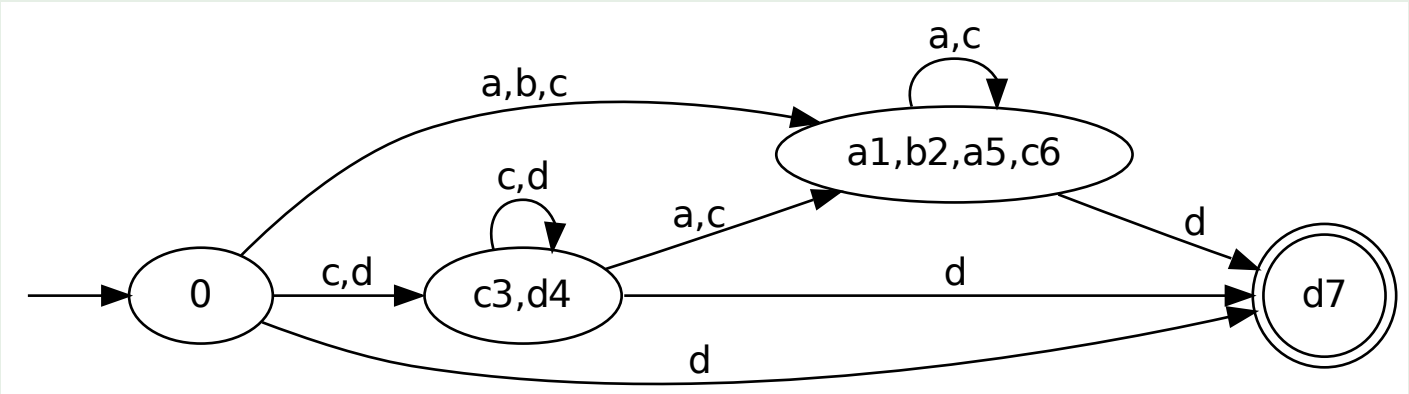
Let Σ be an alphabet, a finite automaton is a 5-tuple $\mathcal{A} = \langle \Sigma, Q, I, F, E \rangle$ where

Q	is a finite set of states,
$I \subseteq Q$	is the set of initial states,
$F \subseteq Q$	is the set of final states,
$E \subseteq Q \times \Sigma \times Q$	is the set of transitions.

Example (Finite state automaton)



Example (Finite state automaton)



- Path, accepted path
- accepted word
- Language $L(\mathcal{A})$ recognized by $\mathcal{A}$

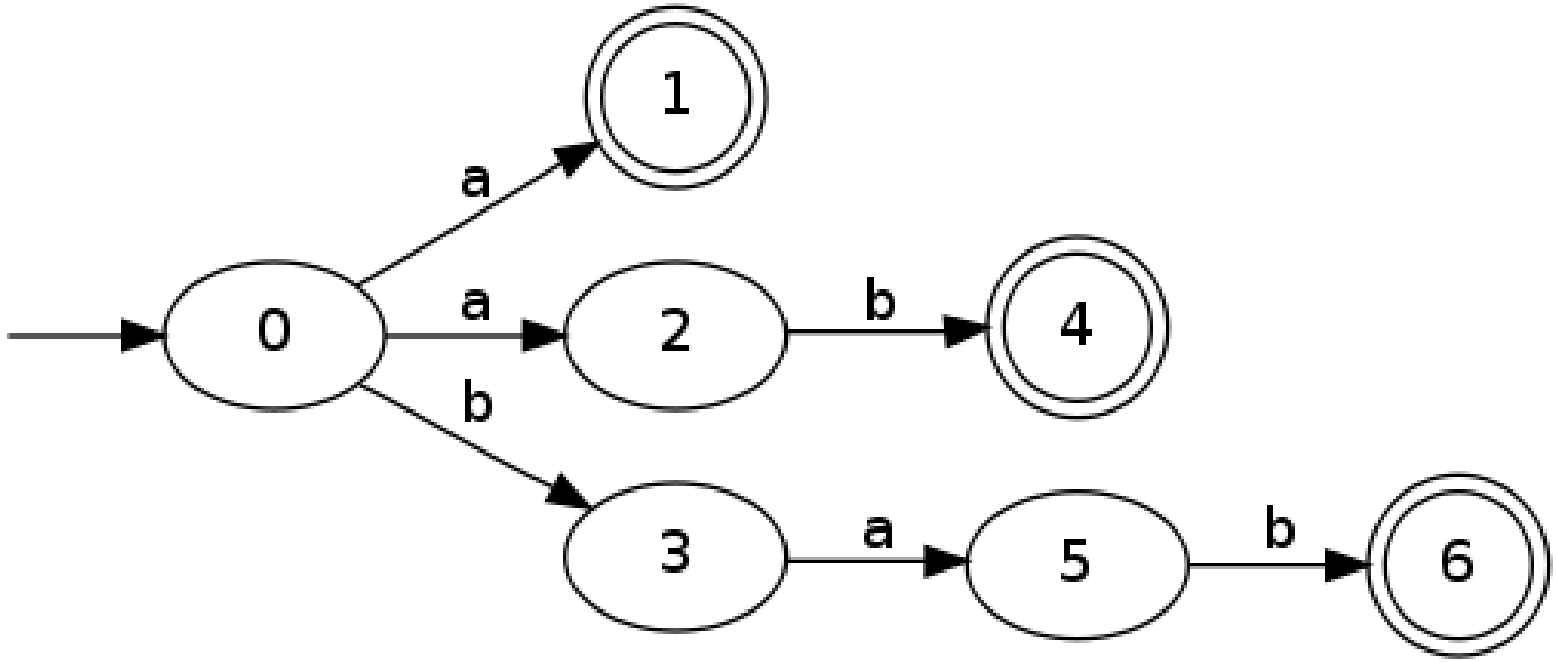
Languages
Maximal automaton of a sample

Definition (Maximal automaton)

Given a sample I_+, the maximal automaton recognizing I_+ is the biggest automaton recognizing I_+.

Example (Maximal automaton)

$I_+ = \{a, ab, bab\}$:



Languages
Quotient of an automaton

Definition (Partition of a set)

For any set S, a partition π is defined by $\pi = \{s \mid s \subseteq S\}$

$$\forall s \in \pi, s \neq \emptyset,$$

such that $\forall s, s' \in \pi, s \cap s' = \emptyset,$

$$\cup_{s \in \pi} = S$$

Definition (Quotient of an automaton)

$\mathcal{A}/\pi = \langle \Sigma, Q', I', F', E' \rangle$ is a quotient of $\mathcal{A} = \langle \Sigma, Q, I, F, E \rangle$ iff

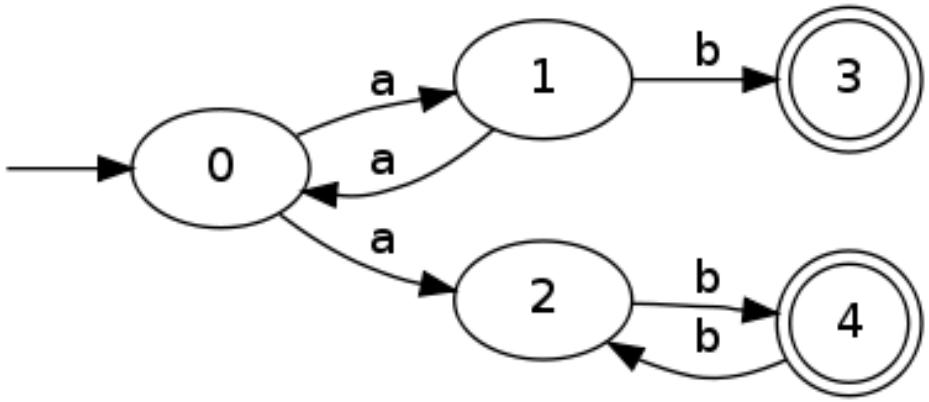
$$\begin{aligned} Q' &= Q/\pi \\ I &= \{B \in Q' \mid \exists q \in B s.t. q \in I\} \\ F &= \{B \in Q' \mid \exists q \in B s.t. q \in F\} \\ E &= \{\langle B, a, B' \rangle \in Q' \times \Sigma \times Q' \mid \exists q \in B, q' \in B', a \in \Sigma \\ &\quad s.t. \langle q, a, q' \rangle \in E\} \end{aligned}$$

Property

if $\mathcal{A}/\pi$ is the quotient of $\mathcal{A}$ w.r.t. relation π, then $L(\mathcal{A}) \subseteq L(\mathcal{A}/\pi)$

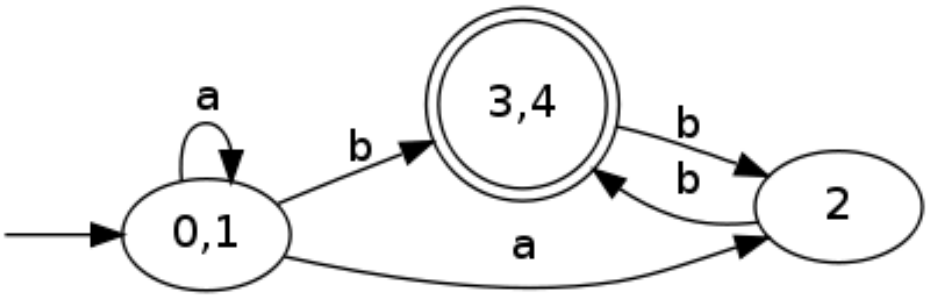
- any derivation produces a generalisation,
- the set of automata obtained by subsequent derivation forms a lattice.

Languages
Example : partition π_1



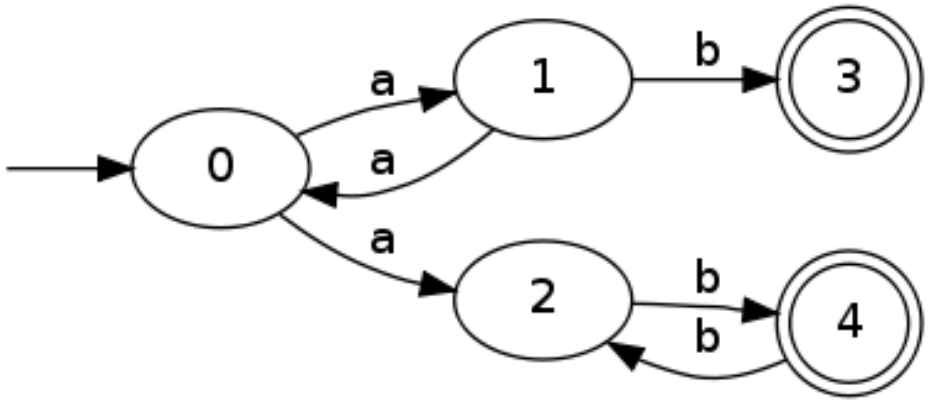
$$L(\mathcal{A}) = a(aa)^*b + ab(bb)^*$$

$$\pi_1 = \{\{0,1\},\{2\},\{3,4\}\}$$



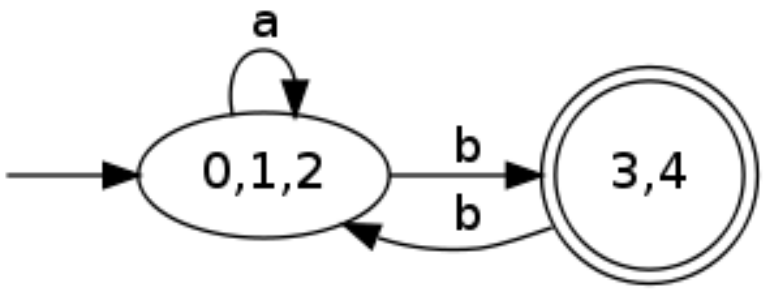
$$L(\mathcal{A}/\pi_1) = a^*(b + ab(bb)^*)$$

Languages
Example : partition π_2



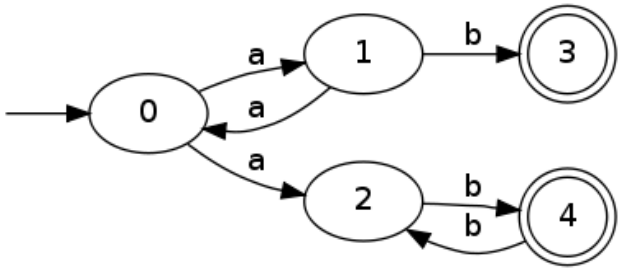
$$L(\mathcal{A}) = a(aa)^*b + ab(bb)^*$$

$$\pi_2 = \{\{0,1,2\},\{3,4\}\}$$



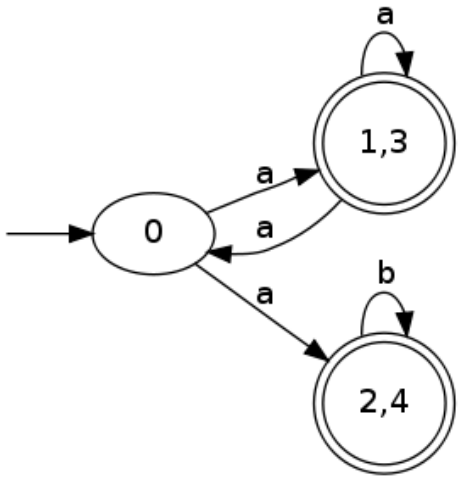
$$L(\mathcal{A}/\pi_2) = a^*b(ba^*b)^*$$

Languages
Example : partition π_3



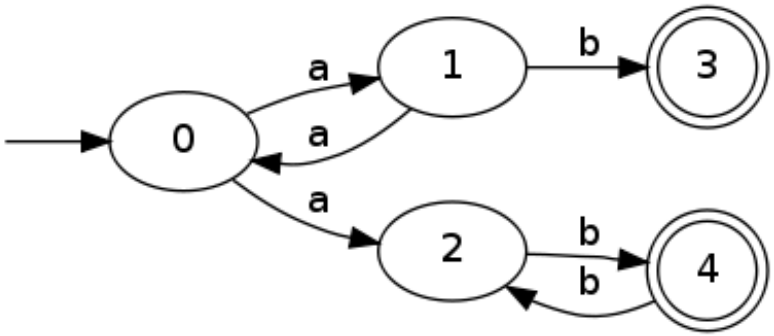
$$L(\mathcal{A}) = a(aa)^*b + ab(bb)^*$$

$$\pi_3 = \{\{0\}, \{1,3\}, \{2,4\}\}$$



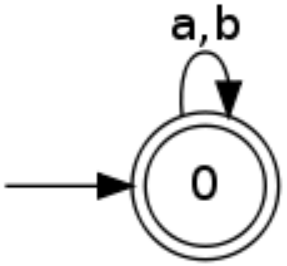
$$L(\mathcal{A}/\pi_3) = (aa^*a)^*(a + ab^*)$$

Languages
Example partition π_4



$$L(\mathcal{A}) = a(aa)^*b + ab(bb)^*$$

$$\pi_4 = \{\{0, 1, 3, 2, 4\}\}$$



$$L(\mathcal{A}/\pi_4) = (a + b)^*$$

Languages
The mining task

- Find an automaton that generalizes, *not too much*, the input sample : characterizable, heuristic methods ;
- the space of possibilities is the number of partitions of the set of states of the sample maximal automaton ;
- the size of the exploration space is the number of partitions of the maximal automaton coding the sample : $B_{n+1} = \Sigma_{k=0}^{n} \left(\begin{array}{c} n \\ k \end{array} \right) B_k$

$B_5 = 52, \ldots, B_{10} = 115975, B_{20} \approx 5.10^{13}, B_{30} \approx 8.10^{23}, B_{50} \approx 2.10^{49} \ldots$

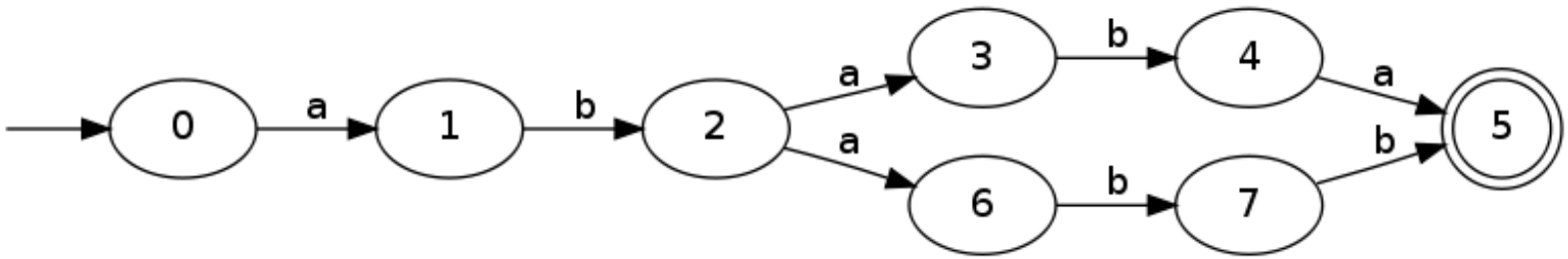
Languages

A characterizable method : the $k - RI$ algorithm

Definition (k-deterministic automaton)

$\mathcal{A}$ is k-deterministic if, from any state, there is at most one path for any given word of length $\geq k$.

Example (A 2-deterministic automaton)



Deterministic automaton are 0-deterministic.

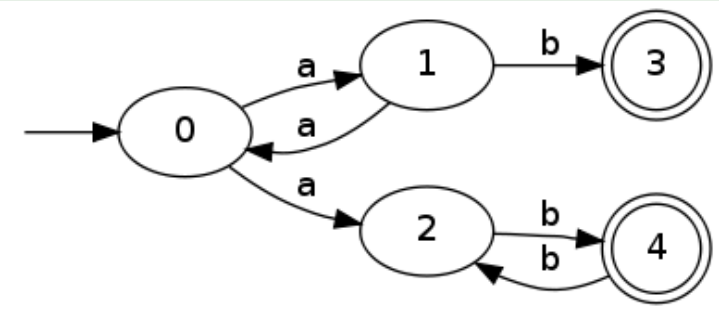
Languages

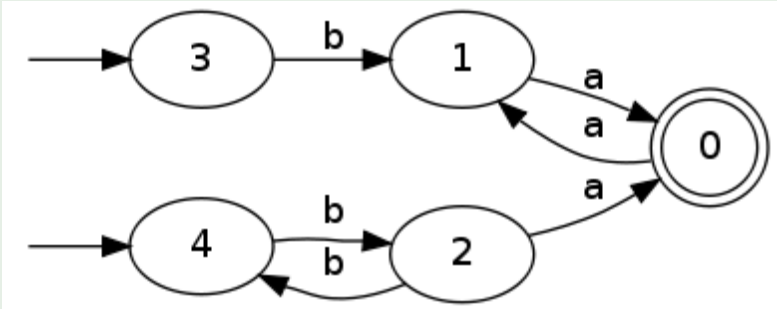
A characterizable method : the $k - RI$ algorithm

Definition (Reverse automaton)

The reverse automaton of $\mathcal{A} = \langle \Sigma, Q, I, F, E \rangle$ is defined by $\mathcal{A}^R = \langle \Sigma, Q, F, I, E' \rangle$ with $E' = \{\langle q, a, q' \rangle \mid \langle q', a, q \rangle \in E\}$.

Example (Reverse automaton)





Languages

A characterizable method : the $k - RI$ algorithm

Definition (k-reversible automaton)

An automaton is k-reversible if it is deterministic and its reverse automaton is k-deterministic.

Dana Angluin [Ang82] gave a characterization of reversible languages and an inference algorithm : $k - RI$.

Dana Angluin.
Inference of Reversible Languages,
Journal of the ACM, Volume 29, 1982.

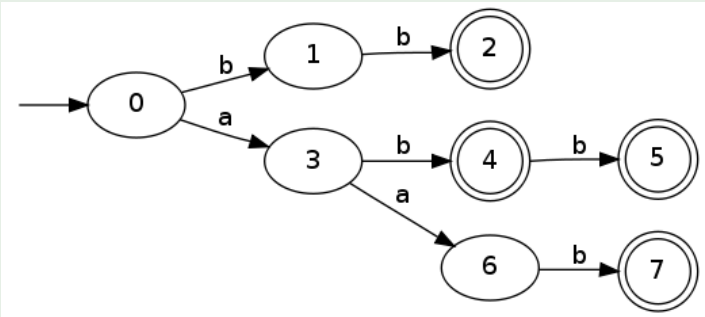
The $k - RI$ algorithm

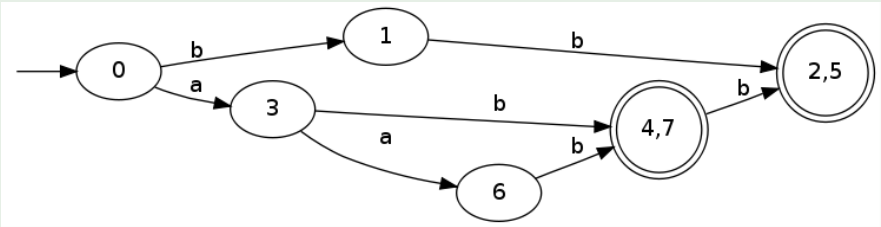
$k \leftarrow$ order of the model, $I_+ \leftarrow$ positive sample
$\mathcal{A} \leftarrow PTA(I_+)$
$\pi \leftarrow Q$
while $\neg$ k-reversible$(\mathcal{A}/\pi)\{$
    $(B_1, B_2) \leftarrow$non-reversible$(\mathcal{A}/\pi, \pi)$
    $\pi \leftarrow \pi \setminus \{B_1, B_2\} \cup \{B_1 \cup B_2\}$
$\}$
return $\mathcal{A}/\pi$

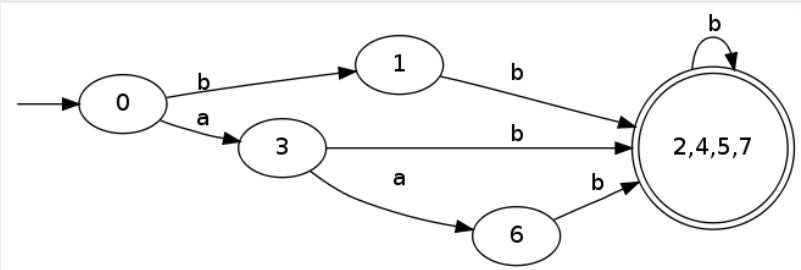
Languages

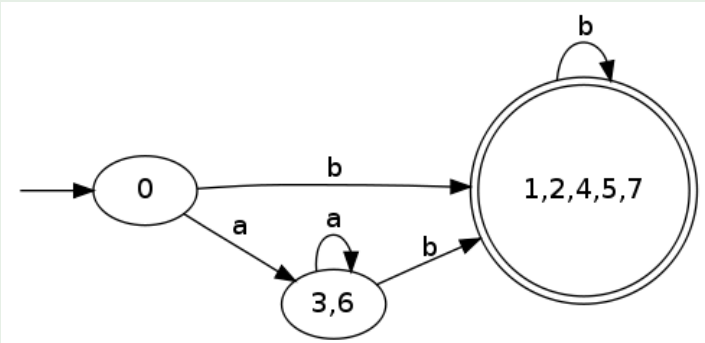
A characterizable method : the $k - RI$ algorithm

Example ($k - RI$ run with $I_+ = \{ab, bb, aab, abb\}, k = 1$)





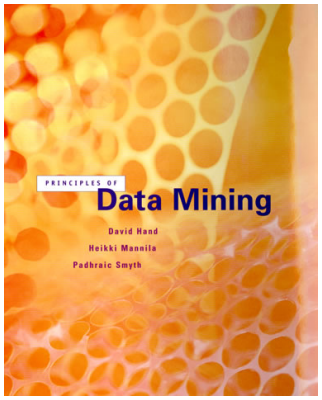




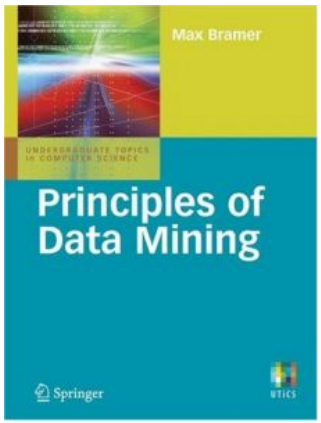
$$L(\mathcal{A}/\pi) = b^+ + a^+b^+ = a^*b^+$$

Going further ...

Some references to go further ...



Principles of Data Mining, *David J. Hand, Heikki Mannila and Padhraic Smyth*, ISBN-13 : 978-0-262-08290-7



Principles of Data Mining, Undergraduate Topic in Computer Science, *Max Bramer*, ISBN-13 : 978-1-846-28765-7

WEKA : associations.apriori