Course overview

T-61.5060 Algorithmic methods of data mining (3 cp) P

- T-61.5060 Tiedon louhinnan algoritmiset menetelmät (3 ov) L
- Data mining, also called knowledge discovery in databases (KDD)
- In Finnish: tiedon louhinta, tietämyksen muodostaminen
- Goal of the course: an overview of pattern discovery
- Biased overview
- Theory and examples
- Course home page:
  http://www.cis.hut.fi/Opinnot/T-61.5060/
- Email: t615060@james.hut.fi

Contents of the course

- Introduction: what is knowledge discobery, types of tasks, etc.
- Discovery of association rules
- Frequent episodes
- Case study: discovery of episodes from telecommunications alarm databases
- Discovery of all frequent patterns
- Basics of cluster analysis
- Link analysis: basic ideas
- (Search for integrity constraints in databases)

Course organization

- Lectures: Thursdays 12–14, Heikki Mannila, Robert Gwadera, Kai Puolamäki
- Exercises: Wednesday, Kai Puolamäki, starting September 19
- Language of intruction: Finnish
- A small project assingment, themes will be given in early October
- Exam on xxx

Material

- H. Mannila, H. Toivonen: Knowledge discovery in databases: the search for frequent patterns; available from the web page of the course
- copies of slides available on the web during the course
- Background material: D. Hand, H. Mannila, P.Smyth: Principles of Data Mining, MIT Press 2001.

Data mining activities at UH/CIS

- basic research: algorithms, theory
- applied research: genetics, ecology, ubiquitous computing, documents, natural language, ...
- FDK "From Data to Knowledge": center of excellence 2002–2007
- HIIT Basic Research Unit
- Neural Networks Research Center. Academy of Finland Center of Excellence

Chapter 1: Introduction

Chapter 1. Introduction

- Introduction
- What is data mining?
- Some examples
- Data mining vs. statistics and machine learning

What is data mining?

Goal: obtain useful knowledge
from large masses of data.

Data mining is the analysis of (often large) observational data sets to find unsuspected relationships and to summarize the data in novel ways that are both understandable and useful to the data analyst

- "Tell something interesting about this data."
- "Describe this data."
- exploratory data analysis on large data sets

Examples of discovered knowledge

- association rules:
  "80 % of customers who buy beer and sausage buy also mustard"
- rules: "if Age $<$ 40 then Income $<$ 10"
- functional dependencies:
  $A \rightarrow B$, i.e., "if $t[A] = u[A]$, then $t[B] = u[B]$"
- models: $Y = aX + b$
- clusterings

Example: sales data

- 0/1 matrix D
- rows:customer transactions
- columns: products
- $t(A) = 1$ iff customer t contained product A
- thousands of columns, millions of rows
- easy to collect
- find something interesting from this?

Association rules

- mustard, sausage, beer $\Rightarrow$ chips
- conditional probability (*accuracy*) of the rule: 0.8
- *frequency* of the rule: 0.2
- arbitrary number of conjuncts on the left-hand side

Find all association rules with frequency at least *min_fr*.

Example: student/course data

- matrix D with $D(t, c) = 1$ iff student t has taken course c

```
1 1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 0 0 0 0 0 0 1 0 0 0 1 0 1 0 0 0 1 0 0 0 1 0 0 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
0 1 0 0 1 1 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1
1 1 0 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Example: student/course data

- example rule:

  `if {Data Communications, Unix, Networks}`

  `then (graduation) (0.3)`

Example: words in documents

- Given a collection of documents
- Find out which words define topics, i.e., some sort of clusters of words that tend to occur together
- Topic 1: "mining", "data", "association", "rule", etc.
- Topic 2: "lecture", "exercise", "exam","material","assignment", etc.
- Some documents talk about topic 1, some about topic 2, some about both
- How to find the topics from data?

Example: SKICAT sky survey

approximately 1000 photographs of 13000x13000 pixels

$\Rightarrow 3 \cdot 10^9$ smudges of light, each with 40 attributes

- task 1: classify the objects to
  - stars
  - galaxies
  - others
- task 2: find some interesting clusters of objects
  - quasars with high redshift
  - ...

machine learning? pattern recognition?

Why data mining?

- raw data is easy to collect, expensive to analyze
- more data than can be analyzed using traditional methods
- suspicion that important knowledge could be there
- successful applications
- methods: machine learning, statistics, databases
- a tool for exploratory data analysis
- can and has to be used in combination with traditional methods
- strong interest from 1989–, hype from 1995–

The KDD process

- understanding the domain,
- preparing the data set,
- discovering patterns, clusters, models, etc
- postprocessing of discovered regularities, and
- putting the results into use

Where is the effort?

iteration, interaction

Phases of finding patterns, models, clusters, etc.

- What is the task? What do we want to find out?
- What is the model, or pattern class?
- What is the score function, i.e., how do we know which model fits the data well?
- What is the algorithm? How do we find the model?

Data mining tasks

- Pattern discovery: find interesting patterns that occur frequently in the data
- Prediction: predict the value of a variable in the data
- Clustering: group the observations (or variables) into groups of similar objects
- Outlier detection: find observations that are unusual
- ...

Example: words in documents

- 0-1 matrix representation of documents
- Rows: document
- Columns: words
- Entry M_{ij} has a 1 if document i contains word j, 0 otherwise
- "Bag-of-words" representation
- Loses a lot of information, but a lot is retained

```
1 1 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 0 0 0 0 0 0 1 0 0 0 1 0 1 0 0 0 1 0 0 0 1 0 0 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
0 1 0 0 1 1 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1
1 1 0 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Data mining tasks in document data

- Find collections of word that occur frequently together in the same documents
- Find topics from data
- Cluster similar documents together
- Will term x occur in the document
- Which are unusual documents?

Another view of data mining tasks

- Exploratory data analysis
- Descriptive modeling
- Predictive modeling
- Discovering patterns and rules
- Retrieval by content

Data mining and related areas

- How does data mining relate to statistics
- How does data mining relate to machine learning?
- Other related areas?

Data mining vs. machine learning

- machine learning methods are used for data mining
  - classification, clustering, ...
- amount of data makes a difference
  - accessing examples can be a problem
- data mining has more modest goals:
  - automating tedious discovery tasks, not aiming at human performance in real discovery
  - helping user, not replacing them
- this course: data mining $\setminus$ machine learning

Data mining vs. statistics

"tell me something interesting about this data" – what else is this than statistics?

- the goal is similar
- different types of methods
- in data mining one investigates a lot of possible hypotheses
- amount of data
- data mining as a preliminary stage for statistical analysis
- challenge to data mining: better contact with statistics

Data mining and databases

- ordinary database usage: deductive
- knowledge discovery: "inductive"
- new requirements for database management systems
- novel data structures, algorithms, and architectures are needed
- SQL queries; OLAP queries; data mining

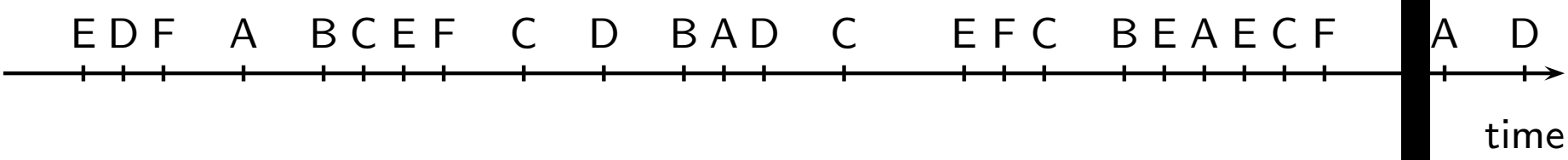
Data mining and algorithms

- Lots of nice connections
- A wealth of interesting research questions
- Some will be treated later in the course

Example of pattern discovery: episodes in sequences

- Data: events in time, e.g.,
  - alarms in telecommunications networks
  - user actions in an user interface
  - database transactions
  - biostatistical events
- Goals: understanding the structure of the process producing the events, prediction of future events.
- How to discover which combinations of events occur frequently?

Example sequence



Observations:

- whenever E occurs, F occurs soon
- whenever A and B occur (in either order), C occurs soon

Telecommunications alarm log

Event	Time
KE82K02-31	780560888
KE82K10-16	780560892
H-M-K09-57	780560917
SOT-K01-03	780560926
MUU-K03-04	780561011
KE82K10-16	780561015
PAK-K14-27	780561119
KE82K10-16	780561138

100–400 different events

events occur recurrently

1 month = 70 000 alarms

Sequences and episodes

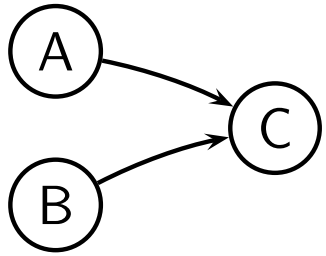
Data a sequence of (event, time) pairs

$(A, 123), (B, 125), (D, 140), (A, 150), (C, 151), (D, 201), (A, 220)$

Patterns episodes: a set of events

occurring close to each other in time

in an order extending a given partial order



Pattern class All serial episodes / all parallel episodes / all episodes / ...

Occurrence pattern occurs frequently in data if there are sufficiently many windows of size W in the data such that the pattern occurs in the window

Discovering frequent patterns

- find all frequent episodes of size 1
- build candidate episodes of size 2
- check which episodes of size 2 occur frequently
- continue
- incremental recognition, using previous rounds of computation, ...

A general model for data mining

- given a class of patterns
- an occurrence criterion
- find all patterns from the class that occur frequently enough

Why is data mining an interesting research area?

- practically relevant
- easy theoretical questions
- the whole spectrum: from theoretical issues to systems issues to concrete data cleaning to novel discoveries
- easy to cooperate with other areas and with industry

Chapter 2: Association rules

Chapter 2. Association rules

- 1. Problem formulation
- 2. Rules from frequent sets
- 3. Finding frequent sets
- 4. Experimental results
- 5. Related issues
- 6. Rule selection and presentation
- 7. Theoretical results

Example

- Customer 1: mustard, sausage, beer, chips

  Customer 2: sausage, ketchup

  Customer 3: beer, chips, cigarettes

  ...

  Customer 236513: coke, chips

- beer $\Rightarrow$ chips
  - accuracy (conditional probability): 0.87
  - frequency (support): 0.34

Problem formulation: data

- a set R of items
- a *0/1 relation* r *over* R is a collection (or multiset) of subsets of R
- the elements of r are called *rows*
- the number of rows in r is denoted by $|r|$
- the *size* of r is denoted by $||r|| = \sum_{t \in r} |t|$

Row ID	Row
t_1	$\{A, B, C, D, G\}$
t_2	$\{A, B, E, F\}$
t_3	$\{B, I, K\}$
t_4	$\{A, B, H\}$
t_5	$\{E, G, J\}$

Figure 1: An example 0/1 relation r over the set $R = \{A, \ldots, K\}$.

Notation

- Sometime we write just ABC for $\{A, B, C\}$ etc.
- Attributes = variables
- An observation in the data is
  - A set of attributes, or
  - a row of 0s and 1s

Patterns: sets of items

- r a 0/1 relation over R
- $X \subseteq R$
- X *matches* a row $t \in r$, if $X \subseteq t$
- the set of rows in r matched by X is denoted by $\mathcal{M}(X, r)$, i.e., $\mathcal{M}(X, r) = \{t \in r \mid X \subseteq t\}$.
- the *(relative) frequency* of X in r, denoted by *fr*(X, r), is

$$\frac{|\mathcal{M}(X, r)|}{|r|}.$$

- Given a *frequency threshold min_fr* $\in [0, 1]$, the set X is *frequent*, if *fr*$(X, r) \geq$ *min_fr*.

Example

Row ID	A	B	C	D	E	F	G	H	I	J	K
t_1	1	1	1	1	0	0	1	0	0	0	0
t_2	1	1	0	0	1	1	0	0	0	0	0
t_3	0	1	0	0	0	0	0	0	1	0	1
t_4	1	1	0	0	0	0	0	1	0	0	0
t_5	0	0	0	0	1	0	1	0	0	1	0

a 0/1 relation over the schema $\{A, \ldots, K\}$

- $fr(\{A, B\}, r) = 3/5 = 0.6$
- $\mathcal{M}(\{A, B\}, r) = \{t_1, t_2, t_4\}$

Frequent sets

- given R (a set), r (a 0/1 relation over R), and *min_fr* (a frequency threshold)
- the collection of frequent sets $\mathcal{F}(r, \mathit{min\_fr})$

$$\mathcal{F}(r, \mathit{min\_fr}) = \{X \subseteq R \mid \mathit{fr}(X, r) \geq \mathit{min\_fr}\},$$

- In the example relation:
  $\mathcal{F}(r, 0.3) = \{\emptyset, \{A\}, \{B\}, \{E\}, \{G\}, \{A, B\}\}$

Association rules

- Let R be a set, r a 0/1 relation over R, and $X, Y \subseteq R$ sets of items
- $X \Rightarrow Y$ is an *association rule* over r.
- The *accuracy* of $X \Rightarrow Y$ in r, denoted by $conf(X \Rightarrow Y, r)$, is $\frac{|\mathcal{M}(X \cup Y, r)|}{|\mathcal{M}(X, r)|}$.
  - The accuracy $conf(X \Rightarrow Y, r)$ is the conditional probability that a row in r matches Y given that it matches X

Association rules II

- The *frequency* *fr*$(X \Rightarrow Y, r)$ of $X \Rightarrow Y$ in r is *fr*$(X \cup Y, r)$.
  - frequency is also *called support*
- a *frequency threshold min_fr* and a *accuracy threshold min_conf*
- $X \Rightarrow Y$ *holds* in r if and only if *fr*$(X \Rightarrow Y, r) \geq$ *min_fr* and *conf* $(X \Rightarrow Y, r) \geq$ *min_conf*.

Discovery task

- given R, r, *min_fr*, and *min_conf*
- find all association rules $X \Rightarrow Y$ that hold in r with respect to *min_fr* and *min_conf*
- X and Y are disjoint and non-empty

- *min_fr* $= 0.3$, *min_conf* $= 0.9$
- The only association rule with disjoint and non-empty left and right-hand sides that holds in the database is $\{A\} \Rightarrow \{B\}$
- frequency 0.6, accuracy 1
- when is the task feasible? interesting?
- note: asymmetry between 0 and 1

How to find association rules

- Find all frequent item sets $X \subseteq R$ and their frequencies.
- Then test separately for all $Y \subset X$ with $Y \neq \emptyset$ whether the rule $X \setminus Y \Rightarrow Y$ holds with sufficient accuracy.
- Latter task is easy.
- exercise: rule discovery and finding frequent sets are equivalent problems

Rule generation

Algorithm

Input: A set R, a 0/1 relation r over R, a frequency threshold *min_fr*, and a accuracy threshold *min_conf*.

Output: The association rules that hold in r with respect to *min_fr* and *min_conf*, and their frequencies and accuracies.

Method:

1. // Find frequent sets (Algorithm 52):
2. compute $\mathcal{F}(r, \mathit{min\_fr}) := \{X \subseteq R \mid \mathit{fr}(X, r) \geq \mathit{min\_fr}\}$;
3. // Generate rules:
4. **for** all $X \in \mathcal{F}(r, \mathit{min\_fr})$ **do**
5. **for** all $Y \subset X$ with $Y \neq \emptyset$ **do**
6. **if** $\mathit{fr}(X)/\mathit{fr}(X \setminus Y) \geq \mathit{min\_conf}$ **then**
7. output the rule $X \setminus Y \Rightarrow Y$, $\mathit{fr}(X)$, and $\mathit{fr}(X)/\mathit{fr}(X \setminus Y)$;

Correctness and running time

- the algorithm is correct
- running time?

Finding frequent sets: reasoning behind Apriori

- trivial solution: look at all subsets of R
- not feasible
- iterative approach
- first frequent sets of size 1, then of size 2, etc.
- a collection $\mathcal{C}_l$ of candidate sets of size l
- then obtain the collection $\mathcal{F}_l(r)$ of frequent sets by computing the frequencies of the candidates from the database
- minimize the number of candidates?

- monotonicity: assume $Y \subseteq X$
- then $\mathcal{M}(Y) \supseteq \mathcal{M}(X)$, and $fr(Y) \geq fr(X)$
- if X is frequent then Y is also frequent
- Let $X \subseteq R$ be a set. If any of the proper subsets $Y \subset X$ is not frequent then (1) X is not frequent and (2) there is a non-frequent subset $Z \subset X$ of size $|X| - 1$.

Example

$\mathcal{F}_2(r) = \{\{A, B\}, \{A, C\}, \{A, E\}, \{A, F\}, \{B, C\}, \{B, E\}, \{C, G\}\},$

- then $\{A, B, C\}$ and $\{A, B, E\}$ are the only possible members of $\mathcal{F}_3(r)$,
- levelwise search: generate and test
- candidate collection:

  $\mathcal{C}(\mathcal{F}_l(r)) = \{X \subseteq R \| |X| = l+1 \text{ and } Y \in \mathcal{F}_l(r) \text{ for all } Y \subseteq X, |Y| = l\}$.

Apriori algorithm for frequent sets

Algorithm

Input: A set R, a 0/1 relation r over R, and a frequency threshold *min_fr*.

Output: The collection $\mathcal{F}(r, \mathit{min\_fr})$ of frequent sets and their frequencies.

Method:

1. $\mathcal{C}_1 := \{\{A\} \mid A \in R\}$;
2. $l := 1$;
3. **while** $\mathcal{C}_l \neq \emptyset$ **do**
4. // Database pass (Algorithm 59):
5. compute $\mathcal{F}_l(r) := \{X \in \mathcal{C}_l \mid \mathit{fr}(X, r) \geq \mathit{min\_fr}\}$;
6. $l := l + 1$;
7. // Candidate generation (Algorithm 56):
8. compute $\mathcal{C}_l := \mathcal{C}(\mathcal{F}_{l-1}(r))$;
9. **for** all l and **for** all $X \in \mathcal{F}_l(r)$ **do** output X and $\mathit{fr}(X, r)$;

Correctness

- reasonably clear
- optimality in a sense?
- For any collection $\mathcal{S}$ of subsets of X of size l, there exists a 0/1 relation r over R and a frequency threshold *min_fr* such that $\mathcal{F}_l(r) = \mathcal{S}$ and $\mathcal{F}_{l+1}(r) = \mathcal{C}(\mathcal{S})$.
- fewer candidates do not suffice

Additional information can change things...

- frequent sets: $\{A, B\}$, $\{A, C\}$, $\{A, D\}$, $\{B, C\}$, and $\{B, D\}$
- candidates: $\{A, B, C\}$ and $\{A, B, D\}$
- what if we know that $fr(\{A, B, C\}) = fr(\{A, B\})$
- can infer $fr(\{A, B, D\}) < min\_fr$
- how?

Candidate generation

- how to generate the collection $\mathcal{C}(\mathcal{F}_l(r))$?
- trivial method: check all subsets
- compute potential candidates as unions $X \cup Y$ of size $l+1$
- here X and Y are frequent sets of size l
- check which are true candidates
- not optimal, but fast
- collections of item sets are stored as arrays, sorted in the lexicographical order

Candidate generation algorithm

Algorithm

Input: A lexicographically sorted array $\mathcal{F}_l(r)$ of frequent sets of size l.

Output: $\mathcal{C}(\mathcal{F}_l(r))$ in lexicographical order.

Method:

1. **for** all $X \in \mathcal{F}_l(r)$ **do**
2. **for** all $Y \in \mathcal{F}_l(r)$ such that $X < Y$ and X and Y share their $l-1$ lexicographically first items **do**
3. **for** all $Z \subset (X \cup Y)$ such that $|Z| = l$ **do**
4. **if** Z is not in $\mathcal{F}_l(r)$ **then** continue with the next Y at line 2;
5. output $X \cup Y$;

Correctness and running time

Theorem 1 *Algorithm 56 works correctly.*

Theorem 2 *Algorithm 56 can be implemented to run in time* $\mathcal{O}(l^2 \ |\mathcal{F}_l(r)|^2 \log |\mathcal{F}_l(r)|)$.

Optimizations

compute many levels of candidates at a single pass

$$\begin{aligned}\mathcal{F}_2(r) &= \{\{A,B\},\{A,C\},\{A,D\},\{A,E\},\\ &\quad \{B,C\},\{B,D\},\{B,G\},\{C,D\},\{F,G\}\}.\end{aligned}$$

$$\begin{aligned}\mathcal{C}(\mathcal{F}_2(r)) &= \{\{A,B,C\},\{A,B,D\},\{A,C,D\},\{B,C,D\}\},\\ \mathcal{C}(\mathcal{C}(\mathcal{F}_2(r))) &= \{\{A,B,C,D\}\},\ \text{and}\\ \mathcal{C}(\mathcal{C}(\mathcal{C}(\mathcal{F}_2(r)))) &= \emptyset.\end{aligned}$$

Database pass

- go through the database once and compute the frequencies of each candidate
- thousands of candidates, millions of rows

Algorithm

Input: R, r over R, a candidate collection $\mathcal{C}_l \supseteq \mathcal{F}_l(r, \mathit{min\_fr})$, and $\mathit{min\_fr}$.

Output: Collection $\mathcal{F}_l(r, \mathit{min\_fr})$ of frequent sets and frequencies.

Method:

1. // Initialization:
2. **for** all $A \in R$ **do** $A.\mathit{is\_contained\_in} := \emptyset$;
3. **for** all $X \in \mathcal{C}_l$ and **for** all $A \in X$ **do**
4. $\quad A.\mathit{is\_contained\_in} := A.\mathit{is\_contained\_in} \cup \{X\}$;
5. **for** all $X \in \mathcal{C}_l$ **do** $X.\mathit{freq\_count} := 0$;
6. // Database access:
7. **for** all $t \in r$ **do**
8. $\quad$ **for** all $X \in \mathcal{C}_l$ **do** $X.\mathit{item\_count} := 0$;
9. $\quad$ **for** all $A \in t$ **do**
10. $\quad\quad$ **for** all $X \in A.\mathit{is\_contained\_in}$ **do**
11. $\quad\quad\quad X.\mathit{item\_count} := X.\mathit{item\_count} + 1$;
12. $\quad\quad\quad$ **if** $X.\mathit{item\_count} = l$ **then** $X.\mathit{freq\_count} := X.\mathit{freq\_count} + 1$;
13. // Output:
14. **for** all $X \in \mathcal{C}_l$ **do**
15. $\quad$ **if** $X.\mathit{freq\_count}/|r| \geq \mathit{min\_fr}$ **then** output X and $X.\mathit{freq\_count}/|r|$;

Data structures

- for each $A \in R$ a list $A.$*is_contained_in* of candidates that contain A
- For each candidate X we maintain two counters:
  - $X.$*freq_count* the number of rows that X matches,
  - $X.$*item_count* the number of items of X

Correctness

- clear (?)

Time complexity

- $\mathcal{O}(||r|| + l\,|r|\,|\mathcal{C}_l| + |R|)$

Experimental results

- small course registration database
- 4 734 students
- 127 courses
- frequency thresholds 0.01–0.2

Size	Frequency threshold					
	0.200	0.100	0.075	0.050	0.025	0.010
1	6	13	14	18	22	36
2	1	21	48	77	123	240
3	0	8	47	169	375	898
4	0	1	12	140	776	2 203
5	0	0	1	64	1 096	3 805
6	0	0	0	19	967	4 899
7	0	0	0	2	524	4 774
8	0	0	0	0	165	3 465
9	0	0	0	0	31	1 845
10	0	0	0	0	1	690
11	0	0	0	0	0	164
12	0	0	0	0	0	21
13	0	0	0	0	0	1

Table 1: Number of frequent sets of each size with different frequency thresholds.

	Frequency threshold					
	0.200	0.100	0.075	0.050	0.025	0.010
Candidate sets:						
Count	142	223	332	825	4 685	24 698
Generation time (s)	0.1	0.1	0.2	0.2	1.1	10.2
Frequent sets:						
Count	7	43	122	489	4 080	23 041
Maximum size	2	4	5	7	10	13
Database pass time (s)	0.7	1.9	3.5	10.3	71.2	379.7
Match	5 %	19 %	37 %	59 %	87 %	93 %
Rules ($min\_conf = 0.9$):						
Count	0	3	39	503	15 737	239 429
Generation time (s)	0.0	0.0	0.1	0.4	46.2	2 566.2
Rules ($min\_conf = 0.7$):						
Count	0	40	193	2 347	65 181	913 181
Generation time (s)	0.0	0.0	0.1	0.8	77.4	5 632.8
Rules ($min\_conf = 0.5$):						
Count	0	81	347	4 022	130 680	1 810 780
Generation time (s)	0.0	0.0	0.1	1.1	106.5	7 613.62

Different statistics of association rule discovery with course database.

	Candidates		Frequent sets		
Size	Count	Time (s)	Count	Time (s)	Match
1	127	0.05	22	0.26	17 %
2	231	0.04	123	1.79	53 %
3	458	0.04	375	5.64	82 %
4	859	0.09	776	12.92	90 %
5	1 168	0.21	1 096	18.90	94 %
6	1 058	0.30	967	18.20	91 %
7	566	0.24	524	9.69	93 %
8	184	0.11	165	3.09	90 %
9	31	0.04	31	0.55	100 %
10	3	0.01	1	0.15	33 %
11	0	0.00			
Total	4 685	1.13	4 080	71.19	87 %

Number of sets and time used for set of different sizes

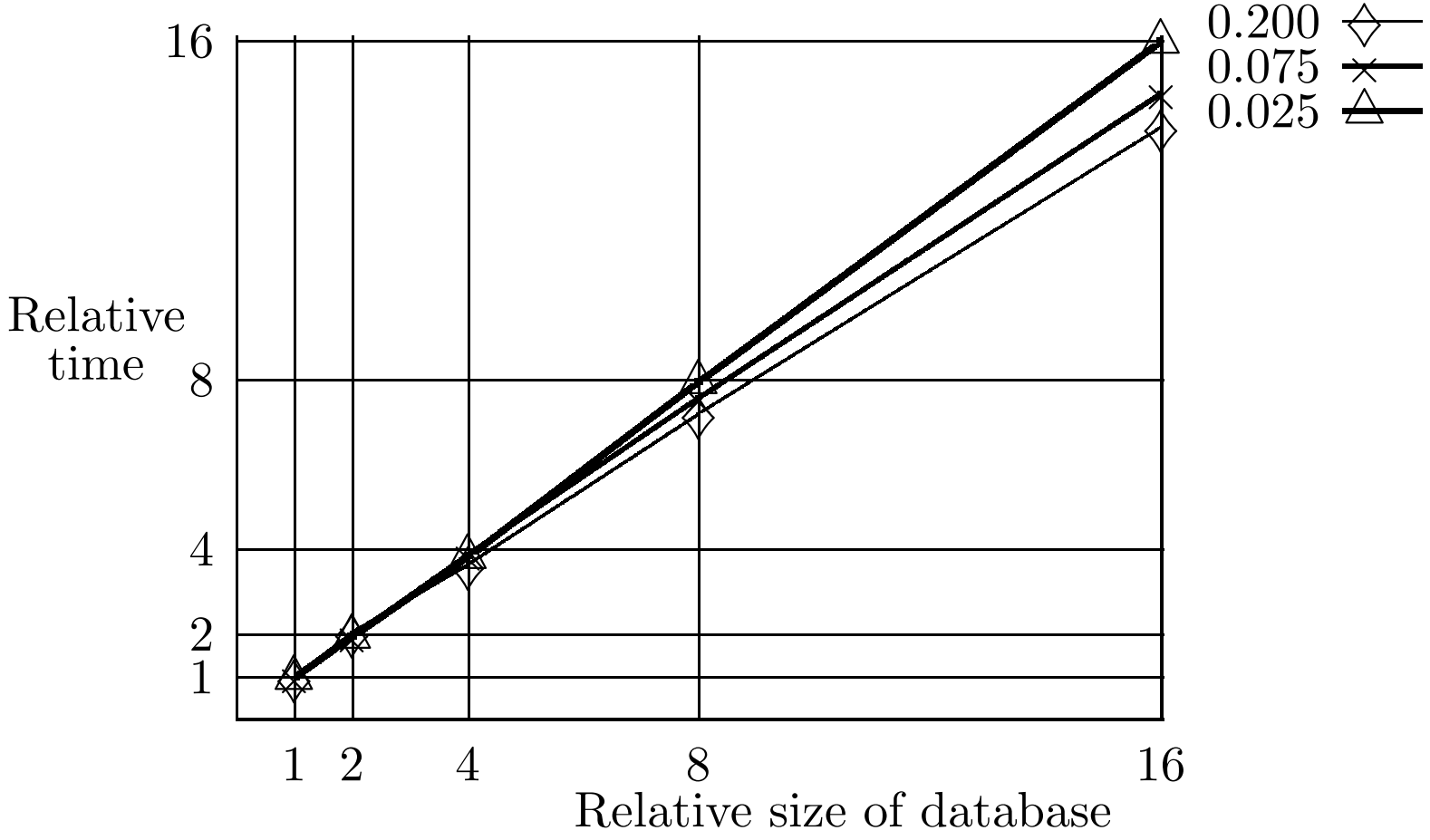


Figure 2: Results of scale-up tests.

Extensions

- candidate generation
- rule generation
- database pass
  - inverted structures
  - Partition method
  - hashing to determine which candidates match a row or to prune candidates
- item hierarchies
- attributes with continuous values

Rule selection and presentation

- recall the KDD process
- association rules etc.: idea is to generate **all** rules of a given form
- lots of rules
- all rules won't be interesting
- how to make it possible for the user to find the truly interesting rules?
- second-order knowledge discovery problem
- provide tools for the user

Uninteresting rules

- There are 2010 association rules in the course enrollment database that match at least 11 students (i.e., the frequency (or support) threshold is 0.01).
- prior knowledge: Design and Analysis of Algorithms $\Rightarrow$ Introduction to Computers (0.97, 0.03).
- uninteresting attributes or attribute combinations. Introduction to Computers $\Rightarrow$ Programming in Pascal (0.95, 0.60) is useless, if the user is only interested in graduate courses.
- Rules can be redundant. Data Communications, Programming in C $\Rightarrow$ Unix Platform (0.14, 0.03) and Data Communications, Programming in C, Introduction to Unix $\Rightarrow$ Unix Platform (0.14, 0.03).

Iteration

- filter out rules referring to uninteresting courses
- all rules containing basic courses away: only half are left
- focus to, e.g., all rules containing the course "Programming in C"
- filter out "Unix Platform"
- etc.

Operations

- pruning: reduction of the number of rules;
- ordering: sorting of rules according, e.g., to statistical significance; and
- structuring: organization of the rules, e.g., to clusters or hierarchies.
- other operations?

Pruning using templates

- hierarchies among attributes {Artificial Intelligence, Programming in C, Data Communications} $\subset$ Undergraduate Course $\subset$ Any Course,
- a template is an expression $A_1, \ldots, A_k \Rightarrow A_{k+1}, \ldots, A_l$,
- A_i: an attribute name, a class name, or an expression $C+$ or $C*$
- Graduate Course, Any Course$* \Rightarrow$ Design and Analysis of Algorithms
- selective/unselective template

Theoretical analyses

- fairly good algorithm
- is a better one possible?
- how good will this algorithm be on future data sets
- a lower bound (skipped)
- association rules on random data sets (skipped)
- sampling

Sampling for finding association rules

- two causes for complexity
- lots of attributes
- lots of rows
- potentially exponential in the number of attributes
- linear in the number of rows
- too many rows: take a sample from them
- in detail later