Hereditary Families of Sets in Data Mining

Dan A. Simovici

1 Exact Descriptions of Sets of Objects

How It All Began

Combinatorial Challange:

- A typical supermarket may well have several thousand items on its shelves.
- If no customer has more than five items in his shopping cart, there are $\sum_{i=1}^{5} \binom{10000}{i}$ possible contents of this cart!

What Supermarkets Need

- identifying associations between item sets: how many of the customers who bought bread and cheese also bought butter;
- associations have marketing consequences: if it turns out that many of the customers who bought bread and cheese also bought butter, the supermarket will place butter physically close to bread and cheese in order to stimulate the sales of butter.

Rymon Tree

$\mathcal{T}$ is a Rymon tree for $\mathcal{P}(S)$ if

- the root of $\mathcal{T}$ is labelled by S, and
- the set of children of U in $\mathcal{T}$ is

$$\{U - \{e\} \in \mathcal{F} \mid e \in U\}.$$

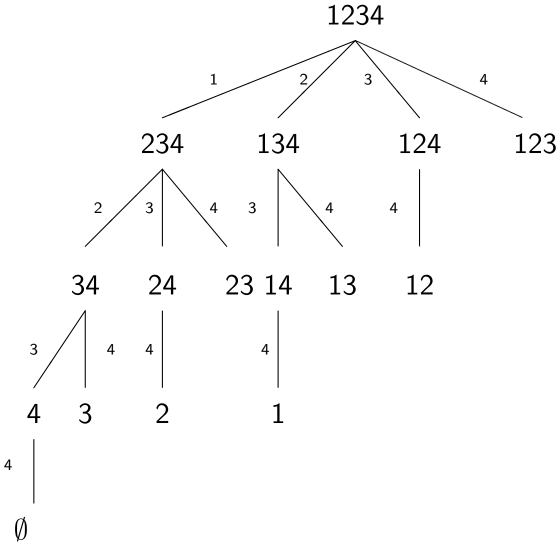
Dual Rymon Tree

$\mathcal{T}$ is a dual Rymon tree for $\mathcal{P}(S)$ if

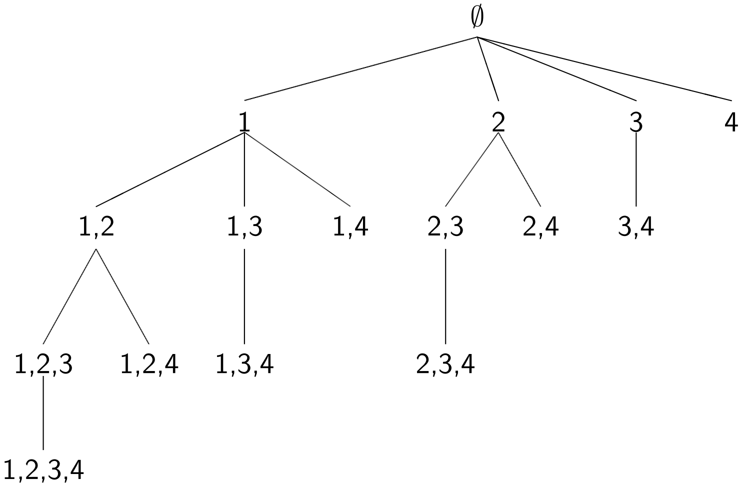
- the root of $\mathcal{T}$ is labelled by the empty set $\emptyset$, and
- the set of children of U in $\mathcal{T}$ is

$$\{U \cup \{e\} \in \mathcal{F} \mid e \in S - U\}.$$

Rymon Tree for $\mathcal{P}(\{1, 2, 3, 4\})$



Dual Rymon tree for $\mathcal{P}(\{1,2,3,4\})$



Formal Setting

I is a finite set: set of items

A *transaction data set on* I is a function $T : \{1, \ldots, n\} \longrightarrow \mathcal{P}(I)$. The set $T(k)$ is the k^{th} *transaction of* T. The numbers $1, \ldots, n$ are the transaction identifiers (*tids*).

Presentation of the Problem - I

Trans.	Content
$T(1)$	{Aspirin, Vitamin C}
$T(2)$	{Aspirin, Sudafed}
$T(3)$	{Tylenol}
$T(4)$	{Aspirin, Vitamin C, Sudafed}
$T(5)$	{Tylenol, Cepacol}
$T(6)$	{Aspirin, Cepacol}
$T(7)$	{Aspirin, Vitamin C}

Presentation of the Problem - II

	Aspirin	Vitamin C	Sudafed	Tylenol	Cepacol
$T(1)$	1	1	0	0	0
$T(2)$	1	0	1	0	0
$T(3)$	0	0	0	1	0
$T(4)$	1	1	1	0	0
$T(5)$	1	0	0	0	1
$T(6)$	1	0	0	0	1
$T(7)$	1	1	0	0	0

Frequent Item Sets

Let $T : \{1, \ldots, n\} \longrightarrow \mathcal{P}(I)$ be a transaction data set on a set of items I.

The *support count* of a subset K of the set of items I in T is the number $\mathrm{suppcount}_T(K)$ given by

$$\mathrm{suppcount}_T(K) = |\{k \mid 1 \leq k \leq n \text{ and } K \subseteq T(k)\}|.$$

The support of an item set K is the number

$$\mathrm{supp}_T(K) = \frac{\mathrm{suppcount}_T(K)}{n}.$$

Example

Let $I = \{i_1, i_2, i_3, i_4\}$ be a collection of items. Consider the transaction data set T given by

$$\begin{aligned} T(1) &= \{i_1, i_2\}, \\ T(2) &= \{i_1, i_3\}, \\ T(3) &= \{i_1, i_2, i_4\}, \\ T(4) &= \{i_1, i_3, i_4\}, \\ T(5) &= \{i_1, i_2\}, \\ T(6) &= \{i_3, i_4\}. \end{aligned}$$

Thus, the support count of the item set $\{i_1, i_2\}$ is 3; similarly, the support count of the item set $\{i_1, i_3\}$ is 2. Therefore, $\text{supp}_T(\{i_1, i_2\}) = \frac{1}{2}$ and $\text{supp}_T(\{i_1, i_3\}) = \frac{1}{3}$.

Central Observation

Let $T : \{1, \ldots, n\} \longrightarrow \mathcal{P}(I)$ be a transaction data set on a set of items I. If K and K' are two item sets, then $K' \subseteq K$ implies $\mathrm{supp}_T(K') \geq \mathrm{supp}_T(K)$.

Frequent Item Sets

- An item set K is μ-*frequent* relative to the transaction data set T if $\text{supp}_T(K) \geq \mu$.
- $\mathcal{F}^{\mu}_{T}$ the collection of all μ-frequent item sets relative to the transaction data set T

$$\mathcal{F}^{\mu}_{T} = \bigcup_{r \geq 1} \mathcal{F}^{\mu}_{T,r}.$$

Crucial fact: $\mathcal{F}^{\mu}_{T}$ is a hereditary collection.

A Property of Dual Rymon Trees

Let $\mathcal{S}_r$ be the collection of item sets that have r elements. and let $\mathcal{T}$ be the dual Rymon tree of $\mathcal{P}(I)$, where $I = \{i_1, \ldots, i_n\}$. If $W \in \mathcal{S}_{r+1}$, where $r \geq 2$, then there exists a unique pair of distinct sets $U, V \in \mathcal{S}_r$ that has a common immediate ancestor $T \in \mathcal{S}_{r-1}$ in $\mathcal{T}$ such that $U \cap V \in \mathcal{S}_{r-1}$ and $W = U \cup V$.

Let T be a transaction data set on a set of items I and let $k \in \mathbb{N}$ such that $k > 1$.
If W is a μ-frequent item set and $|W| = k + 1$, then there exists a μ-frequent item set Z and two items i_m and i_q such that $|Z| = k - 1$, $Z \subseteq W$, $W = Z \cup \{i_m, i_q\}$, and both $Z \cup \{i_m\}$ and $Z \cup \{i_q\}$ are μ-frequent item sets.

The apriori_gen Procedure

Input: a minimum support μ, the collection $\mathcal{F}^{\mu}_{T,k}$ of frequent item sets having k elements;
Output: the set of candidate frequent item sets $\mathcal{C}^{\mu}_{T,k+1}$;
Method:

set $j = 1$;
$\mathcal{C}^{\mu}_{T,j+1} = \emptyset$;
for each $L, M \in \mathcal{F}^{\mu}_{T,k}$ such that
  $L \neq M$ and $L \cap M \in \mathcal{F}^{\mu}_{T,k-1}$ do
  add $L \cup M$ to $\mathcal{C}^{\mu}_{T,k+1}$;
remove all sets K in $\mathcal{C}^{\mu}_{T,k+1}$ where
  there is a subset of K containing k elements
  that does not belong to $\mathcal{F}^{\mu}_{T,k}$.

Features of Apriori Algorithm

- AA operates on "levels" of the form $\mathcal{C}^{\mu}_{T,k}$ of candidate item sets of μ-frequent item sets.
- To build the initial collection of candidate item sets $\mathcal{C}^{\mu}_{T,1}$, every single item set is considered for membership in $\mathcal{C}^{\mu}_{T,1}$.
- The algorithm alternates between a candidate generation phase (accomplished by using `apriori_gen`) and an evaluation phase that involves a data set scan and is therefore the most expensive component of the algorithm.

The AA

Input: transaction data set T and a minimum support μ;
Output: the collection $\mathcal{F}^{\mu}_{T}$ of μ-frequent item sets;
Method: $\mathcal{C}^{\mu}_{T,1} = \{\{i\} \mid i \in I\}$;
  set $i = 1$;
  while $(\mathcal{C}^{\mu}_{T,i} \neq \emptyset)$ **do**
    /* evaluation phase */
    $\mathcal{F}^{\mu}_{T,i} = \{L \in \mathcal{C}^{\mu}_{T,i} \mid \mathsf{supp}_T(L) \geq \mu\}$;
    /* candidate generation */
    $\mathcal{C}^{\mu}_{T,i+1} = \mathtt{apriori\_gen}(\mathcal{F}^{\mu}_{T,i})$;
    $i++$;
  end while;
  output $\mathcal{F}^{\mu}_{T} = \bigcup_{j<i} \mathcal{F}^{\mu}_{T,j}$

Example

	i_1	i_2	i_3	i_4	i_5
$T(1)$	1	1	0	0	0
$T(2)$	0	1	1	0	0
$T(3)$	1	0	0	0	1
$T(4)$	1	0	0	0	1
$T(5)$	0	1	1	0	1
$T(6)$	1	1	1	1	1
$T(7)$	1	1	1	0	0
$T(8)$	0	1	1	1	1

Example (cont'd)

i_1	i_2	i_3	i_4	i_5
5	6	5	2	5

$i_1 i_2$	$i_1 i_3$	$i_1 i_4$	$i_1 i_5$	$i_2 i_3$	$i_2 i_4$	$i_2 i_5$	$i_3 i_4$	$i_3 i_5$	$i_4 i_5$
3	2	1	3	5	2	3	2	3	2
$i_1 i_2 i_3$	$i_1 i_2 i_4$	$i_1 i_2 i_5$	$i_1 i_3 i_4$	$i_1 i_3 i_5$	$i_1 i_4 i_5$	$i_2 i_3 i_4$	$i_2 i_3 i_5$	$i_2 i_4 i_5$	$i_3 i_4 i_5$
2	1	1	1	1	1	2	3	2	2

$i_1 i_2 i_3 i_4$	$i_1 i_2 i_3 i_5$	$i_1 i_2 i_4 i_5$	$i_1 i_3 i_4 i_5$	$i_2 i_3 i_4 i_5$
1	1	1	1	2

$i_1 i_2 i_3 i_4 i_5$

0

Example (cont'd)

$$
\begin{aligned}
\mathcal{C}^{\mu}_{T,1} &= \{i_1, i_2, i_3, i_4, i_5\},\\
\mathcal{F}^{\mu}_{T,1} &= \{i_1, i_2, i_3, i_4, i_5\},\\
\mathcal{C}^{\mu}_{T,2} &= \{i_1 i_2, i_1 i_3, i_1 i_4, i_1 i_5, i_2 i_3, i_2 i_4, i_2 i_5, i_3 i_4, i_3 i_5, i_4 i_5\},\\
\mathcal{F}^{\mu}_{T,2} &= \{i_1 i_2, i_1 i_3, i_1 i_5, i_2 i_3, i_2 i_4, i_2 i_5, i_3 i_4, i_3 i_5, i_4 i_5\},\\
\mathcal{C}^{\mu}_{T,3} &= \{i_1 i_2 i_3, i_1 i_2 i_5, i_1 i_3 i_5, i_2 i_3 i_4, i_2 i_3 i_5, i_2 i_4 i_5, i_3 i_4 i_5\},\\
\mathcal{F}^{\mu}_{T,3} &= \{i_1 i_2 i_3, i_2 i_3 i_4, i_2 i_3 i_5, i_2 i_4 i_5, i_3 i_4 i_5\},\\
\mathcal{C}^{\mu}_{T,4} &= \{i_2 i_3 i_4 i_5\},\\
\mathcal{F}^{\mu}_{T,4} &= \{i_2 i_3 i_4 i_5\},\\
\mathcal{C}^{\mu}_{T,5} &= \emptyset.
\end{aligned}
$$

Association Rules

- An *association rule* on an item set I is a pair of nonempty disjoint item sets (X, Y).
- If $|I| = n$, then there exist possible $3^n - 2^{n+1} + 1$ association rules on I.

An association rule (X, Y) is denoted by $X \Rightarrow Y$. The confidence of $X \Rightarrow Y$ is the number

$$\mathrm{conf}_T(X \Rightarrow Y) = \frac{\mathrm{supp}_T(XY)}{\mathrm{supp}_T(X)}.$$

An association rule holds in a transaction data set T with support μ and confidence c if $\mathrm{supp}_T(XY) \geq \mu$ and $\mathrm{conf}_T(X \Rightarrow Y) \geq c$.

Identifying Association Rules

Z, μ-frequent item set:

- Examine the support levels of the subsets X of Z to ensure that $X \Rightarrow Z - X$ has a sufficient level of confidence, $\text{conf}_T(X \Rightarrow Z - X) = \frac{\mu}{\text{supp}_T(X)}$.
- $\text{supp}_T(X) \geq \mu$ because X is a subset of Z. To obtain a high level of confidence for $X \Rightarrow Z - X$, the support of X must be as small as possible.
- If $X \Rightarrow Z - X$ does not meet the level of confidence, then it is pointless to look for rules of the form $X' \Rightarrow Z - X'$ among the subsets X' of X.

Frequent Item Sets and Galois Connections-I

Let I be a set of items and $T : \{1, \ldots, n\} \longrightarrow \mathcal{P}(I)$ be a transaction data set. Denote by D the set of transaction identifiers $D = \{1, \ldots, n\}$. The functions $\mathrm{items}_T : \mathcal{P}(D) \longrightarrow \mathcal{P}(I)$ and $\mathrm{tids}_T : \mathcal{P}(I) \longrightarrow \mathcal{P}(D)$ are defined by

$$\begin{aligned} \mathrm{items}_T(E) &= \bigcap \{T(k) \mid k \in E\}, \\ \mathrm{tids}_T(H) &= \{k \in D \mid H \subseteq T(k)\}, \end{aligned}$$

for every $E \in \mathcal{P}(D)$ and every $H \in \mathcal{P}(I)$.
Note that $\mathrm{suppcount}_T(H) = |\mathrm{tids}_T(H)|$ for every $H \in \mathcal{P}(I)$.

Closed Item Sets

Let $T : D \longrightarrow \mathcal{P}(I)$ be a transaction data set and let $\mathbf{K}_i : \mathcal{P}(I) \longrightarrow \mathcal{P}(I)$ and $\mathbf{K}_d : \mathcal{P}(D) \longrightarrow \mathcal{P}(D)$ be defined by $\mathbf{K}_i(H) = \text{items}_T(\text{tids}_T(H))$ for $H \in \mathcal{P}(I)$ and $\mathbf{K}_d(E) = \text{tids}_T(\text{items}_T(E))$ for $E \in \mathcal{P}(D)$. Then, $\mathbf{K}_i$ and $\mathbf{K}_d$ are closure operators on I and D, respectively.

A set of items H is closed if and only if, for every set $L \in \mathcal{P}(I)$ such that $H \subset L$, we have $\text{supp}_T(L) < \text{supp}_T(H)$.

Closed Item Sets

The importance of determining the closed item sets is based on the equality $\mathrm{suppcount}_T(\mathrm{items}_T(\mathrm{tids}_T(H))) = |\mathrm{tids}_T(\mathrm{items}_T(\mathrm{tids}_T(H)))| = |\mathrm{tids}_T(H)|$.
If we have the support counts of the closed sets, we have the support count of every set of items and the number of closed sets can be much smaller than the total number of item sets.

Frequent Item Sets and Galois Connections-II

Let $T : \{1, \ldots, n\} \longrightarrow \mathcal{P}(I)$ be a transaction data set. The pair $(\text{items}_T, \text{tids}_T)$ is a Galois connection between the posets $(\mathcal{P}(D), \subseteq)$ and $(\mathcal{P}(I), \subseteq)$:

1. if $E \subseteq E'$, then $\text{items}_T(E') \subseteq \text{items}_T(E)$,
2. if $H \subseteq H'$, then $\text{tids}_T(H') \subseteq \text{tids}_T(H)$,
3. $E \subseteq \text{tids}_T(\text{items}_T(E))$, and
4. $H \subseteq \text{items}_T(\text{tids}_T(H))$

for every $E, E' \in \mathcal{P}(D)$ and every $H, H' \in \mathcal{P}(I)$.

Rough Sets

- Z. Pawlak
- Very useful for approximative descriptions
- Vast number of applications

Approximation Spaces

Approximation space: a pair (S, ρ), where S is a set and ρ is an equivalence on S.

- Lower approximation:

$$\mathrm{lap}_\rho(U) = \bigcup \{[x]_\rho \in S/\rho \mid [x]_\rho \subseteq U\}.$$

Approximation Spaces

Approximation space: a pair (S, ρ), where S is a set and ρ is an equivalence on S.

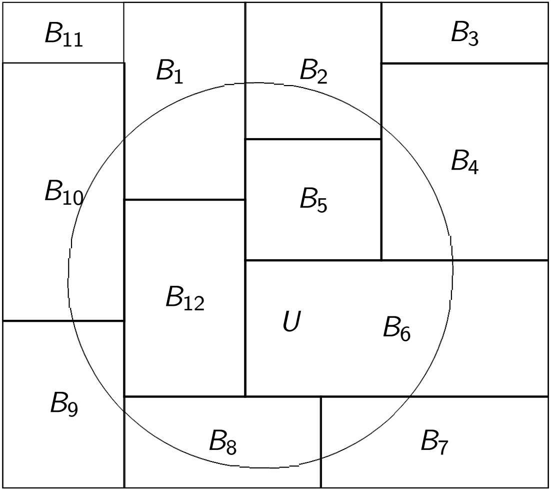
- Lower approximation:

$$\mathrm{lap}_\rho(U) = \bigcup\{[x]_\rho \in S/\rho \mid [x]_\rho \subseteq U\}.$$

- Upper approximation:

$$\mathrm{uap}_\rho(U) = \bigcup\{[x]_\rho \in S/\rho \mid [x]_\rho \cap U \neq \emptyset\}.$$

Lower and Upper Approximations



Borders of Sets

- The positive ρ-border of U:

$$\partial_\rho^+(U) = U - \mathrm{lap}_\rho(U)$$

Borders of Sets

- The positive ρ-border of U:

$$\partial_{\rho}^{+}(U) = U - \mathrm{lap}_{\rho}(U)$$

- The negative ρ-border of U:

$$\partial_{\rho}^{-}(U) = U - \mathrm{lap}_{\rho}(U)$$

Borders of Sets

- The positive ρ-border of U:

$$\partial_\rho^+(U) = U - \mathrm{lap}_\rho(U)$$

- The negative ρ-border of U:

$$\partial_\rho^-(U) = U - \mathrm{lap}_\rho(U)$$

- The ρ-border of U:

$$\partial_\rho(U) = \partial_\rho^+(U) \cup \partial_\rho^-(U) = \mathrm{uap}_\rho(U) - \mathrm{lap}_\rho(U).$$

$\mathrm{lap}_\rho(U) \subseteq \mathrm{uap}_\rho(U)$

$$\begin{aligned} \mathrm{uap}_\rho(U) &= \{t \in S \mid (t,s) \in \rho \text{ for some } s \in U\}, \\ \mathrm{lap}_\rho(U) &= \{t \in U \mid (t,s) \in \rho \text{ implies } s \in U\}. \end{aligned}$$

U is

- ρ-*rough* if $\partial_\rho(U) \neq \emptyset$
- ρ-*crisp* otherwise.

Monotonicity Properties

$\rho \subseteq \sigma$ implies:

- $\text{lap}_\sigma(U) \subseteq \text{lap}_\rho(U) \subseteq U \subseteq \text{uap}_\rho(U) \subseteq \text{lap}_\sigma(U)$
- $\partial_\rho(U) \subseteq \partial_\sigma(U)$
- $\partial_{\rho_1 \wedge \rho_2}(U) \subseteq \partial_{\rho_1}(U) \cap \partial_{\rho_2}(U)$

Data Sets

A data set on H: $T : \{1, \ldots, n\} \times H \longrightarrow \bigcup_{j=1}^{m} \mathrm{Dom}(A_j)$ such that $T(i, A_j) \in \mathrm{Dom}(A_j)$ for $1 \leq i \leq n$ and $1 \leq j \leq m$.

The k^{th} object of T: the sequence $t_k = (T(k,1), \ldots, T(k,m))$.

Object identifiers: $1, \ldots, n$

The set of objects: $\mathcal{O}_T = \{t_1, \ldots, t_n\}$.

Projection of $t_k = (T(k,1), \ldots, T(k,m))$ on $L = \{A_{i_1}, \ldots, A_{i_p}\}$: the p-tuple $(T(k,i_1), \ldots, T(k,i_p))$, denoted by $t_k[L]$.

A set of objects $\mathcal{D} = \{t_5, t_6, t_7, t_8, t_9\}$

T

	A	B	C	D
t_1	a_1	b_2	c_1	d_1
t_2	a_2	b_2	c_1	d_2
t_3	a_3	b_1	c_2	d_1
t_4	a_4	b_1	c_2	d_3
t_5	a_1	b_1	c_1	d_2
t_6	a_3	b_1	c_1	d_2
t_7	a_5	b_3	c_3	d_4
t_8	a_1	b_3	c_3	d_2
t_9	a_2	b_3	c_2	d_3
t_{10}	a_3	b_3	c_2	d_3
t_{11}	a_4	b_2	c_2	d_1
t_{12}	a_1	b_3	c_4	d_4

Equivalences defined by attribute sets

- The equivalence ρ_L on $\mathcal{O}_T$ defined by

$$\rho_L = \{(t, t') \in \mathcal{O}_T^2 \mid t[L] = t'[L]\}.$$

- If L, K are attribute sets, then $\rho_{KL} = \rho_K \cap \rho_L$.
- The border of a set of objects relative to an attribute set is anti-monotonic: $\partial_{\rho_L}(U) \subseteq \partial_{\rho_K}(U)$.

A set of objects $\mathcal{D} = \{t_5, t_6, t_7, t_8, t_9\}$

T

	A	B	C	D
t_1	a_1	b_2	c_1	d_1
t_2	a_2	b_2	c_1	d_2
t_3	a_3	b_1	c_2	d_1
t_4	a_4	b_1	c_2	d_3
t_5	a_1	b_1	c_1	d_2
t_6	a_3	b_1	c_1	d_2
t_7	a_5	b_3	c_3	d_4
t_8	a_1	b_3	c_3	d_2
t_9	a_2	b_3	c_2	d_3
t_{10}	a_3	b_3	c_2	d_3
t_{11}	a_4	b_2	c_2	d_1
t_{12}	a_1	b_3	c_4	d_4

$$\begin{aligned}
\rho_{BC} &= \{\{t_1, t_2\}, \{t_3, t_4\}, \{t_5, t_6\}, \\
&\quad \{t_7, t_8\}, \{t_9, t_{10}\}, \\
&\quad \{t_{11}\}, \{t_{12}\}\} \\
\mathrm{lap}_\rho(\mathcal{D}) &= \{\{t_5, t_6\}, \{t_7, t_8\}\} \\
\partial_{BC}(\mathcal{D}) &= \{\{t_9, t_{10}\}\} \\
\partial^+_{BC}(\mathcal{D}) &= \{\{t_9\}\} \\
\partial^-_{BC}(\mathcal{D}) &= \{\{t_{10}\}\}.
\end{aligned}$$

Exact and Approximative Descriptions

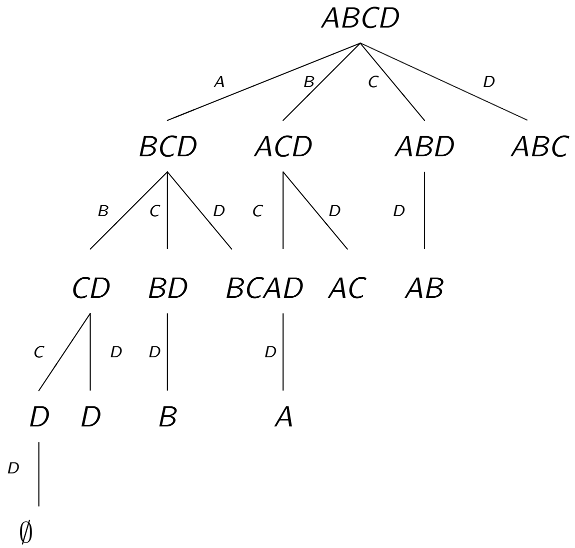
- A set of objects $\mathcal{D}$ is *described by a set of attributes* K if $\partial_K(\mathcal{D}) = \emptyset$ and we refer to K as an *exact description* of $\mathcal{D}$.
- Let ϵ be a number such that $0 \leq \epsilon \leq 1$. A set of objects $\mathcal{D}$ is ϵ-*described by a set of attributes* K if

$$\frac{|\partial_K(\mathcal{D})|}{|\mathcal{D}|} \leq \epsilon.$$

Our goal:

- Finding an exact description as a relational expression of the attributes is intractable.
- Our goal: given T, a set of objects $\mathcal{D} \subseteq \mathcal{O}_T$, determine whether there exists an attribute set K with $|K| \leq k$, such that $|\partial_K(D)| \leq p$.

Rymon Tree of $H = \{A, B, C, D\}$



Main features

- The Rymon tree is searched in a top-down manner.
- Computation of borders take place in breadth-first search fashion.
- In a database having no duplicates the error of the root node is zero.
- If the error at K is greater than the error threshold there is no need for border computing for its descendants because of (the anti-monotonicity property). Thus, we can prune all descendants of K.

Computation of Border $FindBorder(T, \mathcal{D}, K)$

Input: T data set, $\mathcal{D}$ set of objects
Output: Positive and negative borders of $\mathcal{D}$

$Pos := \emptyset$;
$Neg := \emptyset$;
$\bar{\mathcal{D}} = \mathcal{O}_T - \mathcal{D}$;
foreach $t \in \mathcal{D}$ **do**
    foreach $t' \in \bar{\mathcal{D}}$ **do**
        // project on K
        if $t[K] == t'[K]$ **then**
            add t to Pos;
            add t' to Neg;
output $Pos \cup Neg$;

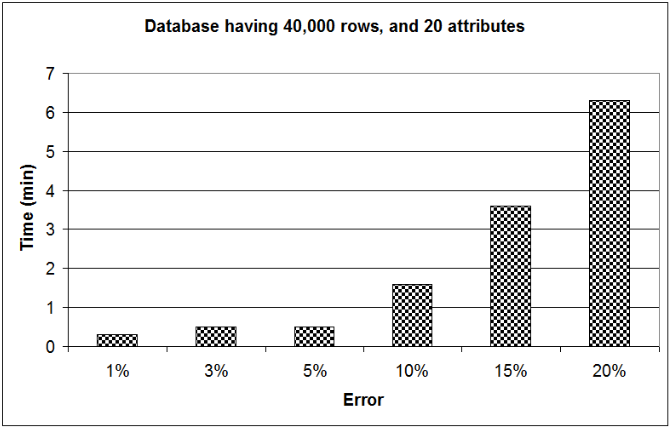
Prunning of Attribute Sets $Prunning(L, R)$

Input: L list of failed descriptors, R set of attributes
Output: all qualified $|R| - 1$ children of R
list all $|R| - 1$-size children of R into P;
foreach $p \in P$ **do**
  if L contains a superset of p **then**
    remove p from P;
output P;

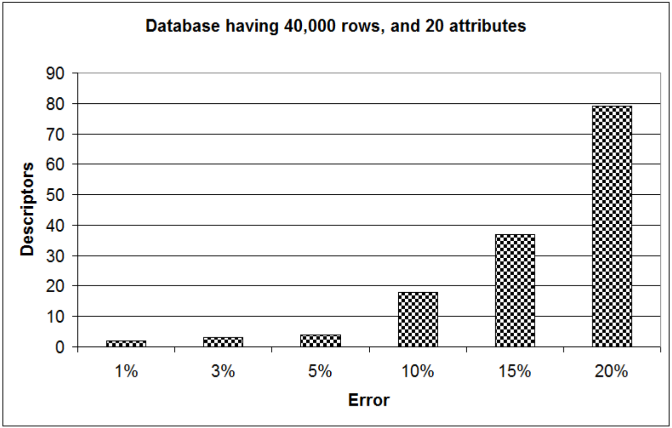
Finding Descriptors of $\mathcal{D}$, $FindAll(T, H, \mathcal{D}, err)$

Input: T data set, $\mathcal{D}$ set of objects, *err* error threshold
Output: all descriptors of $\mathcal{D}$
initialize a queue Q;
initialize a list L;
add H to Q;
while(Q is non-empty) **do**
  $R :=$ remove head of Q;
  if $|FindBorder(T, \mathcal{D}, R)| \leq err$ **then**
    output R;
    $children := Prunning(L, R)$;
    add children to Q;
  else
    add R to L;

Running Time Results for a 40K set



Unique Descriptors for a 40K set



What is next?

- Using genetic algorithms for searching the space of approximative descriptions

What is next?

- Using genetic algorithms for searching the space of approximative descriptions
- Identification of applications for the algorithm

- Problem is suggested by circuit designers who deal with logically programmable arrays for which only a limited number of input tuples are significant.
- We develop an Apriori-like algorithm that takes advantage of the fact that the family of determining sets for a partial function is dually hereditary.

Notations

- $\mathbf{n} = \{0, 1, \ldots, n-1\}$ by $\mathbf{n}$;
- $\mathrm{PF}(\mathbf{r}^n, \mathbf{p})$: set of partial functions with $\mathrm{Dom}(f) \subseteq \mathbf{r}^n$ and **range of (is** $f) \subseteq \mathbf{p}$;

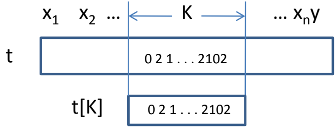
Partial Functions as Tables

T_f

x_1	x_2	x_3	y
0	1	1	0
0	1	2	1
0	2	1	2
0	2	2	2
1	0	1	3
1	0	2	3
2	0	1	3
2	0	2	3
1	1	0	2
1	2	0	2
2	1	0	1
2	2	0	0

Table T_f represents $f \in \mathrm{PF}(\mathbf{3}^3, \mathbf{4})$.
$\mathrm{Dom}(f)$ consists of 44.4% of the possible 27 triplets of $\mathbf{3}^3$.

Projections (Restrictions)



For t in T_f and $K \subseteq \{x_1, \ldots, x_n, y\}$ let $t[K]$ be the *projection* of t on K, that is, the restriction of t to the set K.

Determining Sets for Partial Functions

$V = \{x_{i_0}, \ldots, x_{i_{p-1}}\}$ is a *determining set* for f if for every two tuples t and s from T_f, $t[V] = s[V]$ implies $t[y] = s[y]$.
$\mathrm{DS}(f)$: the collection of determining sets for f
V is a *minimal determining set for f* if $V \in \mathrm{DS}(f)$ and there is no strict subset of V in $\mathrm{DS}(f)$.
$\mathrm{MDS}(f)$: the set of minimal determining sets of f.

A Partial Order on $\mathrm{PF}(\mathbf{r}^n, \mathbf{p})$

Define $f \sqsubseteq g$ if $\mathrm{Dom}(f) \subseteq \mathrm{Dom}(g)$ and $f(a_1, \ldots, a_n) = g(a_1, \ldots, a_n)$ for every $(a_1, \ldots, a_n)$ (equivalently, if g is an extension of f).

- If $V \in \mathrm{DS}(f)$ and $V \subseteq W$, then $W \in \mathrm{DS}(f)$.
- If $f \sqsubseteq g$, then $\mathrm{DS}(g) \subseteq \mathrm{DS}(f)$.

Computing MDS(f)

Input: A partially defined function f.
Output: A collection $\mathcal{D}$ of minimal determining variables sets.

```
dLevel ⟵ ∞
ENQUEUE(Q,∅)
while Q ≠ ∅ do
    X ⟵ DEQUEUE(Q)
    for each V ∈ Child[X] do
        ENQUEUE(Q,V)
        if 𝒟 = ∅ or LEVEL(v) ≤ dLevel then
            if IS_DSET[V] then
                ADD(𝒟, V)
                if dLevel = ∞ then
                dLevel = LEVEL(V)
        else break
end
```

Features of Algorithm

- input is a partially defined function f; the output is a collection of sets with minimum number of variables that f depends on;
- breadth-first search on the Rymon tree for the power-set of the set of variables $E = \{x_1, x_2, ..., x_n\}$ of f;
- the children of a minimal set need not be searched.

The Procedure IS_DET(V)

Input: A node containing a subset of the variables set
Output: true if the set is a determining one, false, otherwise
begin
$S \longleftarrow GET\_VARIABLES(V)$
  for each $tuple \in File$ **do**
    $key \longleftarrow GET\_VALUES(tuple, S)$
    if $key \in MAP$ **then**
      $y \longleftarrow ELEMENT(MAP, key)$
      if $F(tuple) \neq GET\_FVALUE(y)$ **then**
        return false
        break
    else
      ADD(MAP,key,F($tuple$))
return true
end

Experimental Setting - I

- We carried out experiments on a Windows Vista 64-bit machine with 8Gb RAM and 2 $\times$ Quad Core Xeon Proc E5420, running at 2.50 GHz with a 2$\times$6Mb L2 cache. The algorithm was written in Java 6.
- One hundred files were randomly generated for each type of partially defined function (with 8, 16, and 24 variables) using an input radix $r = 3$ and an output radix $p = 5$:
  - 1000 tuples for partially defined functions with 8 variables.
  - 5000 tuples for partially defined functions with 16 and 24 variables.

Experimental Setting - II

- if $(f_1, f_2, \ldots, f_k)$ is a sequence of functions such that

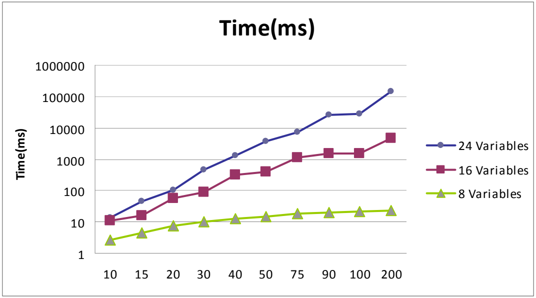
$$f_1 \sqsubseteq f_2 \sqsubseteq \cdots \sqsubseteq f_k,$$

  we have

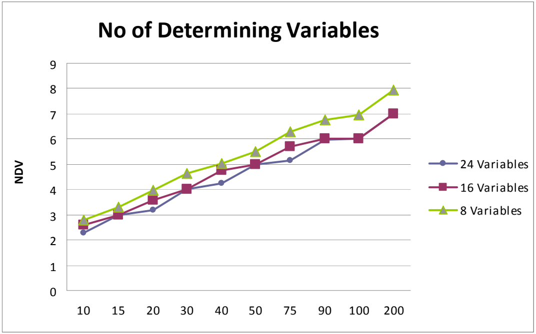
$$\mathrm{DS}(f_k) \subseteq \cdots \subseteq \mathrm{DS}(f_2) \subseteq \mathrm{DS}(f_1).$$

- In our case, $k \in \{10, 15, 20, 30, 40, 50, 75, 90, 100, 200\}$.
- The averages over 100 functions within each group of generated functions (8, 16 and 24 variables).

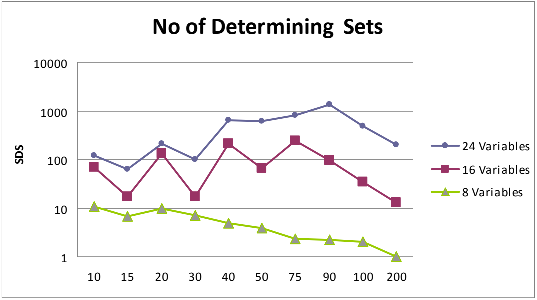
Dependency of average time on number of tuples



Average size of minimal determining set



Average size of MDS(f) for 8, 16 and 24 variables



Conclusions and Future Work

Alternative approaches to be considered:

- a clustering technique for discrete functions starting from a semi-metric that measures the discrepancy between the kernel partitions of these functions;
- using the entropy associated with a set of attributes to determine minimal determining sets.