# Combinatorial local patterns in Boolean dataset

# Plan

1. Introduction

2. Generic algorithm for local pattern mining in 0/1 data
   - Biset mining
   - Enumeration strategy
   - Constraint handling
   - Finding upper bounds

3. Data-Peeler : closed n-sets in n-ary relations
   - Closed $n$-sets
   - Applications

# Combinatorial local pattern extraction

## Set pattern extraction

Declaratively defining interesting patterns :

- $\mathcal{L}$ is a language of patterns to be mined.
- $\mathbf{r}$ is a dataset
- $\mathcal{C}$ is a predicate used for selecting potentially interesting patterns.

A combinatorial local pattern extraction under constraints in $\mathbf{r}$ is

$$\mathcal{P}atterns = \{P \in \mathcal{L} \mid \mathcal{C}(P, \mathbf{r})\}$$

# Combinatorial local pattern extraction

## Set pattern extraction

Declaratively defining interesting patterns :

- $\mathcal{L}$ is a language of patterns to be mined.
- $\mathbf{r}$ is a dataset
- $\mathcal{C}$ is a predicate used for selecting potentially interesting patterns.

A combinatorial local pattern extraction under constraints in **r** is

$$\mathcal{P}atterns = \{P \in \mathcal{L} \mid \mathcal{C}(P, \mathbf{r})\}$$

## Such techniques are useful when

- the dataset **r** is large
- there is few knowledge on the studied phenomenon

# Combinatorial local pattern extraction

## Set pattern extraction

Declaratively defining interesting patterns :

- $\mathcal{L}$ is a language of patterns to be mined.
- $\mathbf{r}$ is a dataset
- $\mathcal{C}$ is a predicate used for selecting potentially interesting patterns.

A combinatorial local pattern extraction under constraints in **r** is

$$\mathcal{P}atterns = \{P \in \mathcal{L} \mid \mathcal{C}(P, \mathbf{r})\}$$

## Main Principle

Some of the constraints are sufficiently tight to drastically reduce the search space and turn the computation to be feasible.

# Examples

## 0/1 dataset

$\mathcal{L} = 2^{\mathcal{I}}$
$\mathbf{r} = \{\{a, b, d\}, \{b, c, d\}, \{a, d\}, \{b\}\}$
$\mathcal{C}$ : Patterns of $\mathcal{L}$ that appear at least twice in $\mathbf{r}$ and of size at least 2.
$\mathcal{P}atterns = \{\{a, d\}, \{b, d\}\}$

## String dataset

$\mathcal{L}$ is the set of finite ordered list of $\mathcal{I}$.
$\mathbf{r} = \{cabbac, bba, ccabd, cdad\}$
$\mathcal{C}$ : Patterns of $\mathcal{L}$ that appear at least twice in $\mathbf{r}$ and of size at least 3.
Answer (with strings) : $\mathcal{P}atterns\{bba, cab\}$
Answer (with sequences) : $\mathcal{P}atterns\{bba, cab, cad\}$

# Local pattern extraction and data experts

Complete solvers for local pattern extraction have reached a kind of maturity :

- numerous extractors
- efficient for various kinds of datasets

# Local pattern extraction and data experts

Complete solvers for local pattern extraction have reached a kind of maturity :

- numerous extractors
- efficient for various kinds of datasets

$\Rightarrow$ Slow progress from Data Mining to Knowledge Discovery

# Local pattern extraction and data experts

Complete solvers for local pattern extraction have reached a kind of maturity :

- numerous extractors
- efficient for various kinds of datasets

$\Rightarrow$ Slow progress from Data Mining to Knowledge Discovery
Still lack of :

- taking into account user's expectations
  - increase the interestingness of extracted patterns
  - decrease the number of spurious patterns
- abstraction and generalization
  - new patterns, e.g., fault-tolerant and numerical patterns
  - new constraints
  - algorithm tuning for efficiency purpose

# Local patterns in 0/1 datasets

| | $i_1$ | $i_2$ | $i_3$ |
|---|---|---|---|
| $t_1$ | 0 | 0 | 1 |
| $t_2$ | 1 | 1 | 0 |
| $t_3$ | 1 | 1 | 1 |
| $t_4$ | 0 | 1 | 0 |

## Extraction task

A biset extraction task under-constraints in a 0/1 dataset **r** can be seen as the computation of the following collection :

$$\mathcal{L} = 2^{\mathcal{T}} \times 2^{\mathcal{I}}$$
$$\mathcal{P}atterns = \{P = (X, Y) \in \mathcal{L} \mid \mathcal{C}(P, \mathbf{r})\}$$

# Formal concepts

| | $i_1$ | $i_2$ | $i_3$ |
|---|---|---|---|
| $t_1$ | 0 | 0 | 1 |
| $t_2$ | 1 | 1 | 0 |
| $t_3$ | 1 | 1 | 1 |
| $t_4$ | 0 | 1 | 0 |

## Formal concept extraction

$$\mathcal{L} = 2^{\mathcal{T}} \times 2^{\mathcal{I}}$$
$$\mathcal{C} \equiv \mathcal{C}_{linked} \wedge \mathcal{C}_{closed}$$
$$\mathcal{C}_{linked}(T, I) = \forall i \in I \text{ and } \forall t \in T,\ (t, i) \in \mathbf{r}$$
$$\mathcal{C}_{closed}(T, I) = \forall t \in \mathcal{T} \setminus T,\ \neg\mathcal{C}_{linked}(T \cup \{t\}, I)$$
$$\forall i \in \mathcal{I} \setminus I,\ \neg\mathcal{C}_{linked}(T, I \cup \{i\})$$
$$\mathcal{P} = \{(\{t_2, t_3\}, \{i_1, i_2\}), (\{t_3\}, \{i_1, i_2, i_3\}), (\{t_2, t_3, t_4\}, \{i_2\}), (\{t_1, t_3\}, \{i_3\})\}$$

# Local patterns in 0/1 datasets

| | $i_1$ | $i_2$ | $i_3$ |
|---|---|---|---|
| $t_1$ | 0 | 0 | 1 |
| $t_2$ | 1 | 1 | 0 |
| $t_3$ | 1 | 1 | 1 |
| $t_4$ | 0 | 1 | 0 |

- A gene (column) is over expressed in a biological situation (row) : all the (maximal) groups of genes that are co-over-expressed in different biological situations

# Local patterns in 0/1 datasets

| | $i_1$ | $i_2$ | $i_3$ |
|---|---|---|---|
| $t_1$ | 0 | 0 | 1 |
| $t_2$ | 1 | 1 | 0 |
| $t_3$ | 1 | 1 | 1 |
| $t_4$ | 0 | 1 | 0 |

- A gene (column) is over expressed in a biological situation (row) : all the (maximal) groups of genes that are co-over-expressed in different biological situations
- A user (row) has appropriate rights to access to a service (column) : all the (maximal) groups of users that have the same rights for some services

# Local patterns in 0/1 datasets

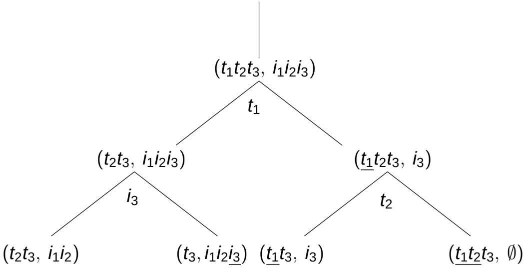
| | $i_1$ | $i_2$ | $i_3$ |
|---|---|---|---|
| $t_1$ | 0 | 0 | 1 |
| $t_2$ | 1 | 1 | 0 |
| $t_3$ | 1 | 1 | 1 |
| $t_4$ | 0 | 1 | 0 |

- A gene (column) is over expressed in a biological situation (row) : all the (maximal) groups of genes that are co-over-expressed in different biological situations
- A user (row) has appropriate rights to access to a service (column) : all the (maximal) groups of users that have the same rights for some services
- A person (row) isn't interested in a topic (column) : all groups of people who have common disinterests.

# Problem setting

| | $i_1$ | $i_2$ | $i_3$ |
|---|---|---|---|
| $t_1$ | 0 | 0 | 1 |
| $t_2$ | 1 | 1 | 0 |
| $t_3$ | 1 | 1 | 1 |
| $t_4$ | 0 | 1 | 0 |

...



# Generalization Problem

**Problem**

- a lot of enumeration strategies
- a lot of different ways of handling constraints
- a lot of "ad-hoc" data structures
- the fundamental ideas hidden in the middle of the haystack

# Generalization Problem

## Problem

- a lot of enumeration strategies
- a lot of different ways of handling constraints
- a lot of "ad-hoc" data structures
- the fundamental ideas hidden in the middle of the haystack

## Goal

Abstracting all these principles while being :

- simple for depicting fundamental principals
- able to use the best ideas of the-state-of-the-art algorithms

# Generalization Problem

## Problem

- a lot of enumeration strategies
- a lot of different ways of handling constraints
- a lot of "ad-hoc" data structures
- the fundamental ideas hidden in the middle of the haystack

## The only assumption

Let $SP$ be a search space. It exists $Upper(\mathcal{C})(SP)$ such that $\forall (X, Y) \in SP\ \neg Upper(\mathcal{C})(SP) \Rightarrow \neg\mathcal{C}(X, Y)$

# Pattern domain

**Pattern domain**

A biset is a couple $(X, Y)$ such that $X \subseteq \mathcal{T}$ and $Y \subseteq \mathcal{I}$

# Pattern domain

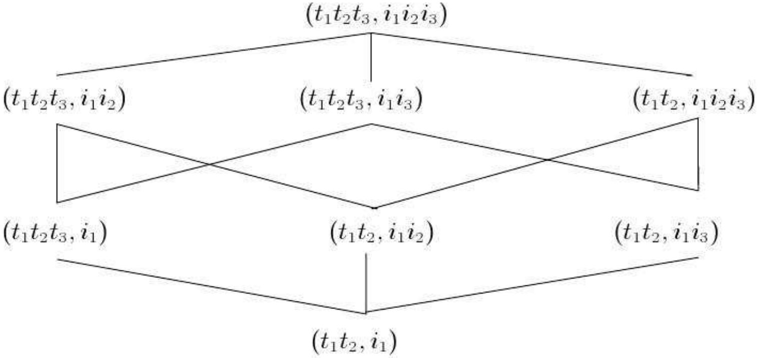
**Search space**

A biset search space *SP* is defined by a lattice $\langle(\bot_{\mathcal{T}}, \bot_{\mathcal{I}}), (\top_{\mathcal{T}}, \top_{\mathcal{I}})\rangle$.

# Pattern domain

## Search space

A biset search space *SP* is defined by a lattice $\langle (\bot_{\mathcal{T}}, \bot_{\mathcal{I}}), (\top_{\mathcal{T}}, \top_{\mathcal{I}}) \rangle$.



# Enumeration

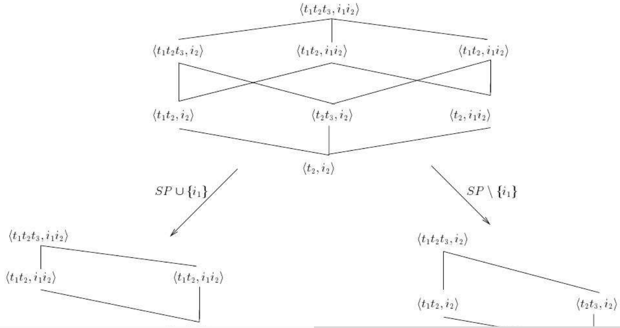
**Let $y$ belongs to $\top_{\mathcal{T}} \setminus \bot_{\mathcal{T}}$ or $\top_{\mathcal{I}} \setminus \bot_{\mathcal{I}}$ :**

$Enumeration(SP, y) =$
$SP \setminus y = \langle(\bot_{\mathcal{T}}, \bot_{\mathcal{I}}), (\top_{\mathcal{T}}, \top_{\mathcal{I}}) \setminus \{y\}\rangle$
$SP \cup y = \langle(\bot_{\mathcal{T}}, \bot_{\mathcal{I}}) \cup \{y\}, (\top_{\mathcal{T}}, \top_{\mathcal{I}})\rangle$

# Enumeration

**Let $y$ belongs to $\top_{\mathcal{T}} \setminus \bot_{\mathcal{T}}$ or $\top_{\mathcal{I}} \setminus \bot_{\mathcal{I}}$ :**

$Enumeration(SP, y) =$
$SP \setminus y = \langle(\bot_{\mathcal{T}}, \bot_{\mathcal{I}}), (\top_{\mathcal{T}}, \top_{\mathcal{I}}) \setminus \{y\}\rangle$
$SP \cup y = \langle(\bot_{\mathcal{T}}, \bot_{\mathcal{I}}) \cup \{y\}, (\top_{\mathcal{T}}, \top_{\mathcal{I}})\rangle$



# Constraint handling

## Checking

$$Checking(SP, \mathcal{C}) = \neg Upper(\mathcal{C})(SP)$$

# Constraint handling

## Checking

$Checking(SP, \mathcal{C}) = \neg Upper(\mathcal{C})(SP)$

## Example - Checking

Extraction task : $\mathcal{C}(X, Y) \equiv \mathcal{C}_{FC} \wedge |X| \times |Y| \geq 8$.
$\langle(t_3, i_2 i_3), (t_1 t_3 t_4, i_2 i_3)\rangle$ can be pruned.

# Constraint handling

## Checking

$$Checking(SP, \mathcal{C}) = \neg Upper(\mathcal{C})(SP)$$

## Propagation

$Propagation(SP, \mathcal{C}) = \langle(\bot'_{\mathcal{T}}, \bot'_{\mathcal{I}}), (\top'_{\mathcal{T}}, \top'_{\mathcal{I}})\rangle$ s.t.

$$\bot'_{\mathcal{T}} = \bot_{\mathcal{T}} \cup \{a \in \top_{\mathcal{T}} \setminus \bot_{\mathcal{T}} | \neg Upper(\mathcal{C}_{IP})(SP \setminus \{a\})\}$$
$$\bot'_{\mathcal{I}} = \bot_{\mathcal{I}} \cup \{a \in \top_{\mathcal{I}} \setminus \bot_{\mathcal{I}} | \neg Upper(\mathcal{C}_{IP})(SP \setminus \{a\})\}$$
$$\top'_{\mathcal{T}} = \top_{\mathcal{T}} \setminus \{a \in \top_{\mathcal{T}} \setminus \bot_{\mathcal{T}} | \neg Upper(\mathcal{C}_{IP})(SP \cup \{a\})\}$$
$$\top'_{\mathcal{I}} = \top_{\mathcal{I}} \setminus \{a \in \top_{\mathcal{I}} \setminus \bot_{\mathcal{I}} | \neg Upper(\mathcal{C}_{IP})(SP \cup \{a\})\}$$

# Constraint handling

## Checking

$Checking(SP, \mathcal{C}) = \neg Upper(\mathcal{C})(SP)$

## Propagation

$Propagation(SP, \mathcal{C}) = \langle(\bot'_{\mathcal{T}}, \bot'_{\mathcal{I}}), (\top'_{\mathcal{T}}, \top'_{\mathcal{I}})\rangle$ s.t.

$$\bot'_{\mathcal{T}} = \bot_{\mathcal{T}} \cup \{a \in \top_{\mathcal{T}} \setminus \bot_{\mathcal{T}} | \neg Upper(\mathcal{C}_{IP})(SP \setminus \{a\})\}$$
$$\bot'_{\mathcal{I}} = \bot_{\mathcal{I}} \cup \{a \in \top_{\mathcal{I}} \setminus \bot_{\mathcal{I}} | \neg Upper(\mathcal{C}_{IP})(SP \setminus \{a\})\}$$
$$\top'_{\mathcal{T}} = \top_{\mathcal{T}} \setminus \{a \in \top_{\mathcal{T}} \setminus \bot_{\mathcal{T}} | \neg Upper(\mathcal{C}_{IP})(SP \cup \{a\})\}$$
$$\top'_{\mathcal{I}} = \top_{\mathcal{I}} \setminus \{a \in \top_{\mathcal{I}} \setminus \bot_{\mathcal{I}} | \neg Upper(\mathcal{C}_{IP})(SP \cup \{a\})\}$$

## Example - Propagation

Extraction task : $\mathcal{C}(X, Y) \equiv \mathcal{C}_{FC} \wedge |X| \times |Y| < 6 \wedge X \cap \{t_4 t_5\} \neq \emptyset$.
$Propagation(\langle(t_3, i_2 i_3), (t_1 t_3 t_4, i_2 i_3)\rangle, \mathcal{C}) = \langle(t_3 t_4, i_2 i_3), (t_3 t_4, i_2 i_3)\rangle$.

# Generic algorithm

$K = (G, M, I)$ a boolean context and $\mathcal{C}_{EP}$ a constraint over $2^{\mathcal{T}} \times 2^{\mathcal{I}}$

**Algorithm**

  **Extract**$(\langle(\emptyset, \emptyset), (\mathcal{T}, \mathcal{I})\rangle)$

End Algorithm

**Extract**$(SP = \langle(\bot_{\mathcal{T}}, \bot_{\mathcal{I}}), (\top_{\mathcal{T}}, \top_{\mathcal{I}})\rangle)$

  $SP \leftarrow \langle(\bot_{\mathcal{T}}, \bot_{\mathcal{I}}), (\top_{\mathcal{T}}, \top_{\mathcal{I}})\rangle$ where

   $\bot_{\mathcal{T}} = \bot_{\mathcal{T}} \cup \{a \in \top_{\mathcal{T}} \setminus \bot_{\mathcal{T}} \mid \neg Upper(\mathcal{C}_{EP})(SP \setminus \{a\})\}$

   $\bot_{\mathcal{I}} = \bot_{\mathcal{I}} \cup \{a \in \top_{\mathcal{I}} \setminus \bot_{\mathcal{I}} \mid \neg Upper(\mathcal{C}_{EP})(SP \setminus \{a\})\}$

   $\top_{\mathcal{T}} = \top_{\mathcal{T}} \setminus \{a \in \top_{\mathcal{T}} \setminus \bot_{\mathcal{T}} \mid \neg Upper(\mathcal{C}_{EP})(SP \cup \{a\})\}$

   $\top_{\mathcal{I}} = \top_{\mathcal{I}} \setminus \{a \in \top_{\mathcal{I}} \setminus \bot_{\mathcal{I}} \mid \neg Upper(\mathcal{C}_{EP})(SP \cup \{a\})\}$

  If $\bot_{\mathcal{T}} \subseteq \top_{\mathcal{T}} \wedge \bot_{\mathcal{I}} \subseteq \top_{\mathcal{I}} \wedge Upper(\mathcal{C}_{EP})(SP)$ Then

   If $(\bot_{\mathcal{T}}, \bot_{\mathcal{I}}) \neq (\top_{\mathcal{T}}, \top_{\mathcal{I}})$ Then

    Let $e \in \top_{\mathcal{T}} \setminus \bot_{\mathcal{T}} \cup \top_{\mathcal{I}} \setminus \bot_{\mathcal{I}}$

    $(SP_1, SP_2) \leftarrow (SP \cup \{e\}, SP \setminus \{e\})$

    **Extract**$(SP_1)$

    **Extract**$(SP_2)$

   Else **Print**$(\bot_{\mathcal{T}}, \bot_{\mathcal{I}})$

   End If

  End If

# Piecewise anti-monotonic constraints

How to define constraint upper-bounds ?

- Simple case : Monotonic and anti-monotonic constraints
- General case : given a constraint and a biset search space

# Piecewise anti-monotonic constraints

## Definition

$\mathcal{C}$ is anti-monotone w.r.t. $\subseteq$ iff $P \subseteq P'$, $\neg\mathcal{C}(P) \Rightarrow \neg\mathcal{C}(P')$

$Upper(\mathcal{C})(SP) = \mathcal{C}(X_1, Y_1)$ where
$X_1 = \top_{\mathcal{T}}$ if $\mathcal{C}$ is monotonic on X
$X_1 = \bot_{\mathcal{T}}$ if $\mathcal{C}$ is anti-monotonic on X
$Y_1 = \top_{\mathcal{I}}$ if $\mathcal{C}$ is monotonic on Y
$Y_1 = \bot_{\mathcal{I}}$ if $\mathcal{C}$ is anti-monotonic on Y

## Examples

- $\mathcal{C}_{division} \equiv |X|/|Y| > \alpha$ :
  $Upper(\mathcal{C}_{division})(SP) \equiv |\top_{\mathcal{T}}|/|\bot_{\mathcal{I}}| > \alpha$
- $\mathcal{C}_{area} \equiv |X| \times |Y| > \alpha$ :
  $Upper(\mathcal{C}_{area}) \equiv |\top_{\mathcal{T}}| \times |\top_{\mathcal{I}}| > \alpha$

# Piecewise anti-monotonic constraints

## How to compute an upper-bound for the constraint $\mathcal{C}_{mean}$ ?

$$\mathcal{C}_{mean}(X) \equiv \frac{\sum_{x \in X} Val^{+}(x)}{\sharp X} > \alpha$$

$\mathcal{C}_{mean}(X)$ can be rewritten as follows :

$$\mathcal{P}_{\mathcal{C}_{mean}}(X_1, X_2) \equiv \frac{\sum_{x \in X_1} Val^{+}(x)}{\sharp X_2} > \alpha$$

Finally, we obtain :

$$Upper(\mathcal{C}_{mean}) \equiv \sum_{i \in \top_{\mathcal{T}}} Val^{+}(i) / |\bot_{\mathcal{T}}| > \alpha$$

# Piecewise anti-monotonic constraints

**How to compute an upper-bound for the constraint $\mathcal{C}_{whateveritis(E,F)}$ ?**

$$\mathcal{C}_{whateveritis(E,F)}(X,Y) \equiv \frac{|X \cap E| \times |Y \cup F|}{|X| \times |Y|} > \alpha$$

We rewrite $\mathcal{C}_{whateveritis(E,F)}(X,Y)$ as follows :

$$\mathcal{P}\mathcal{C}_{whateveritis(E,F)}(X_1, X_2, Y_1, Y_2) \equiv \frac{|X_1 \cap E| \times |Y_1 \cup F|}{|X_2| \times |Y_2|} > \alpha$$

Finally, we obtain :

$$Upper(\mathcal{C}_{whateveritis(E,F)}) \equiv \frac{|\top_{\mathcal{I}} \cap E| \times |\top_{\mathcal{I}} \cup F|}{|\bot_{\mathcal{I}}| \times |\bot_{\mathcal{I}}|} > \alpha$$

# Genericity of the algorithm

The constraints that define formal concepts, itemsets or fault-tolerant patterns are special cases of piecewise (anti)-monotonic constraints. All this patterns can be extracted by means of this framework.

# Introduction to Data-Peeler

Let us consider the following tasks :

**Data - Gene expression**

The gene expression levels of a set of genes in different situations along time (for different timestamps).

**Question**

What are the groups of genes that are co-expressed in different situations during a lot of timestamps ?

# Introduction to Data-Peeler

Let us consider the following tasks :

**Data - Log**

Users from several countries loading web pages about GNU/Linux distributions along time.

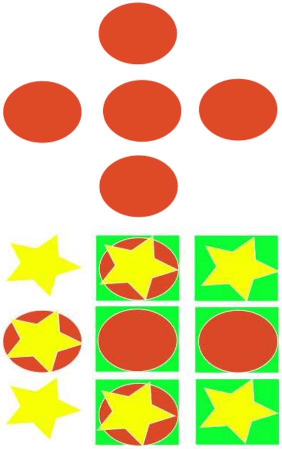
**Question**

What are the sets of countries for which their citizens are "interested in" the same distributions along time.

# $N$-ary relations

In a binary relation $\mathcal{T} \times \mathcal{I}$, a formal concept binds a subset of $\mathcal{T}$ with a subset of $\mathcal{I}$ s.t. both are closed w.r.t. each other.

In a $n$-ary relation $A_1 \times \cdots \times A_n$, a $n$-sets generalizes a formal concept : it binds subsets of every $A_i$ s.t. each of them is closed w.r.t. all the others.

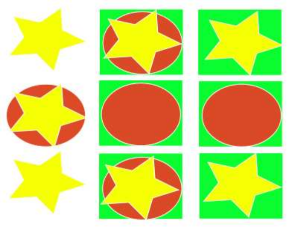


# Example

## $n$-ary boolean relation

$\mathcal{L} = 2^{A_1} \times \cdots \times 2^{A_n}$

Dataset :



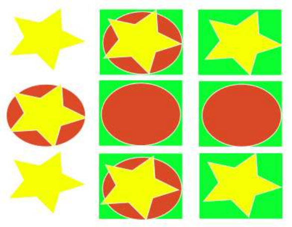
Pattern :

# Example

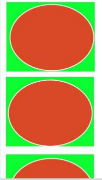
## $n$-ary boolean relation

$\mathcal{L} = 2^{A_1} \times \cdots \times 2^{A_n}$

Dataset :



Pattern :

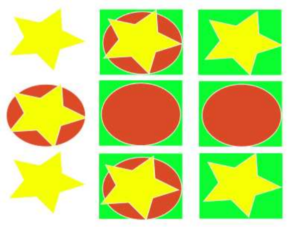


# Example

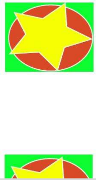
## *n*-ary boolean relation

$\mathcal{L} = 2^{A_1} \times \cdots \times 2^{A_n}$

Dataset :



Pattern :



# Closed $n$-sets

## Constraints

$H = \langle X^1, \cdots, X^n \rangle$ is a closed $n$-set if it satisfies :

- $\mathcal{C}_{linked} \equiv \forall U = (x^1, \cdots, x^n) \in X^1 \times \cdots \times X^n, U \in \mathbf{r}$.
- $\mathcal{C}_{closed} \equiv \forall j \in 1 \cdots n, \forall x^j \in A^j \setminus X^j$, $\langle X^1, \cdots, X^j \cup \{x^j\}, \cdots, X^n \rangle$ does not satisfy $\mathcal{C}_{linked}$.

## Constraints handling

$Propagation(SP, \mathcal{C}_{linked}) = \langle (\perp'_{A_1}, \cdots \perp'_{A_n}), (\top'_{A_1}, \top'_{A_n}) \rangle$ s.t. $y \in A_j$ has been enumerated and $i \neq j$ :

$$\top'_{A_i} = \top_{A_i} \setminus \{v \in \top_{A_i} \mid \neg \mathcal{C}_{linked}(\langle \perp_{A_1}, \cdots, \perp_{A_j} \cup \{y\}, \cdots, \{v\}, \cdots, \perp_{A_n} \rangle\}.$$

# Closed $n$-sets

## Constraints

$H = \langle X^1, \cdots, X^n \rangle$ is a closed $n$-set if it satisfies :

- $\mathcal{C}_{linked} \equiv \forall U = (x^1, \cdots, x^n) \in X^1 \times \cdots \times X^n, U \in \mathbf{r}$.
- $\mathcal{C}_{closed} \equiv \forall j \in 1 \cdots n, \forall x^j \in A^j \setminus X^j$, $\langle X^1, \cdots, X^j \cup \{x^j\}, \cdots, X^n \rangle$ does not satisfy $\mathcal{C}_{linked}$.

## Constraints handling

$Checking(SP, \mathcal{C}_{closed}) \equiv \forall j, \forall y \in A_j \setminus \top_{A_j}$, $\neg\mathcal{C}_{linked}(\top_{A_1}, \cdots, \{y\}, \cdots, \top_{A_n})$

# Data-Peeler

## Difficulties

- Pattern enumeration strategy is not straightforward anymore : lots of degrees of freedom
- No Galois connection
- How to deal with constraints ?
- A 4-ary relation where dimensions are of size 200 contains up to 1.6 $10^9$ tuples. Which data structure should we choose for $n$-sets ?
- No dimension should be privileged

## State-of-the-art

Two algorithms for 3-ary relations : Trias and CubeMiner

# Data-Peeler

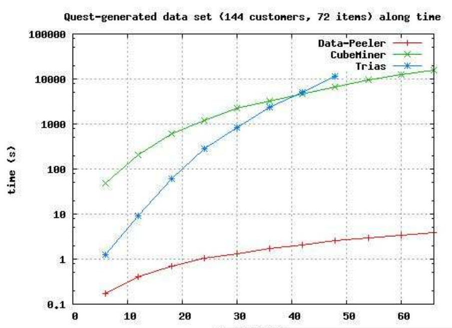
## Overview

- At any recursive call, any element (from any set) can be enumerated
  - A clever enumeration strategy improves the running times by several orders of magnitude
- Piecewise (anti)-monotonic constraints can be pushed (extended to $n$-ary relations)

# Data-Peeler

## Results

- All competitors focusing on ternary relations are much slower
- New relevant constraints



# Applications

**Graph**

How to extract cliques in graphs evolving through 2 dimensions ?

**Example**

For each semester and each country, we have (DistroWatch.com) GNU/Linux distributions for which people have a common interest. How to extract groups of GNU/Linux distributions which are loaded together for several (consecutive) semesters and in several countries.

# Graph Problem

## Extracted patterns

- {Fedora, FreeBSD, Debian, Ubuntu, Gentoo, MEPIS, Slackware, Yellow Dog, Mandriva, openSUSE} - every semester - countries all over the world : general-purpose distributions.
- {Astaro, ClarkConnect, IPCop, m0n0wall, Devil, SmoothWall, CensorNet} - every semester - countries all over the world (*but slightly different from the previous ones*) : firewall oriented distributions.
- {dynebolic, ArtistX, AGNULA, MoviX, GeeXboX} - Every semester - occidental countries and India : movies and music manipulating distributions.
- {B2D, Linpus and PUD for Taiwan and Hong} - two semesters - Taiwan and Hong Kong : traditional Chinese distributions

# Discretization

## Robustness w.r.t. binarization

Let a matrix of dimension $n$, how to binarize such data set when several binarization methods are relevant ? What about relevancy ?

## Example

For each semester and each country, we have (DistroWatch.com) GNU/Linux distributions for which people have a common interest. From initial numerical data set, one can apply different equally relevant discretization methods : "Per-distribution", "Per-semester", "Per-country" and "Global" binarization. How to make binarization more robust ?

# Discretization

## Solution

Add a dimension corresponding to the discretization methods.

| Minimal nb of binarizations | Nb of closed 4-sets | Time (in s) |
|---|---|---|
| 3 | 3580 | 18.93 |
| 4 | 1297 | 15.82 |
| 5 | 229 | 11.62 |
| 6 | 2 | 8.58 |

We constrained the minimal number of binarizations varying from 3 (presence in at least half of the binarizations) to 6 (presence in every binarization).