Specialised vs Declarative Data Mining
Software Testing Applications

Nadjib Lazaar, CNRS, University of Montpellier

Join works with: M. Maamar, Y. Lebbah, S. Loudni, C. Bessiere, et. al.

SIMULA, Oslo, 11 oct. 2018
➤ **Data Mining (DM)** or Knowledge Discovery in Databases (KDD) revolves around the investigation and creation of knowledge, processes, algorithms, and the mechanisms for retrieving potential knowledge from data collections.
Data Mining (DM) or Knowledge Discovery in Databases (KDD) revolves around the investigation and creation of knowledge, processes, algorithms, and the mechanisms for retrieving potential knowledge from data collections.

Mining on:

- Itemsets (Finding itemsets from a collection of transactions)
DATA MINING

➤ **Data Mining (DM)** or Knowledge Discovery in Databases (KDD) revolves around the investigation and creation of knowledge, processes, algorithms, and the mechanisms for **retrieving potential knowledge** from **data collections**.

Mining on:

➤ Itemsets (Finding itemsets from a collection of transactions)

➤ Sequences (Finding subsequences from collection of sequences)
DATA MINING

➤ **Data Mining (DM)** or Knowledge Discovery in Databases (KDD) revolves around the investigation and creation of knowledge, processes, algorithms, and the mechanisms for retrieving potential knowledge from data collections.

Mining on:

➤ Itemsets (Finding itemsets from a collection of transactions)

➤ Sequences (Finding subsequences from collection of sequences)

➤ Graphs (Finding subgraphs from collection of graphs)
Data Mining (DM) or Knowledge Discovery in Databases (KDD) revolves around the investigation and creation of knowledge, processes, algorithms, and the mechanisms for retrieving potential knowledge from data collections.

Mining on:

- Itemsets (Finding itemsets from a collection of transactions)
- Sequences (Finding subsequences from collection of sequences)
- Graphs (Finding subgraphs from collection of graphs)
- Tree, Geometric structures...
DATA MINING APPLICATIONS
DATA MINING APPLICATIONS

➤ Market Basket Analysis [Agrawal93]
DATA MINING APPLICATIONS

➤ Market Basket Analysis [Agrawal93]

➤ Future Healthcare
  ➤ Great potential to improve health systems [Obenshain04]
DATA MINING APPLICATIONS

➤ Market Basket Analysis [Agrawal93]
➤ Future Healthcare
  ➤ Great potential to improve health systems [Obenshain04]
➤ Education
  ➤ Knowledge from data educational environments [Scheuer12]
DATA MINING APPLICATIONS

➤ Market Basket Analysis [Agrawal93]
➤ Future Healthcare
  ➤ Great potential to improve health systems [Obenshain04]
➤ Education
  ➤ Knowledge from data educational environments [Scheuer12]
➤ Fraud and Intrusion detection [Wang10] [Lee98]
DATA MINING APPLICATIONS

➤ Market Basket Analysis [Agrawal93]

➤ Future Healthcare
  ➤ Great potential to improve health systems [Obenshain04]

➤ Education
  ➤ Knowledge from data educational environments [Scheuer12]

➤ Fraud and Intrusion detection [Wang10] [Lee98]

➤ Lie detection and Criminal Investigation [Chen04]
DATA MINING APPLICATIONS

➤ Market Basket Analysis [Agrawal93]

➤ Future Healthcare
   ➤ Great potential to improve health systems [Obenshain04]

➤ Education
   ➤ Knowledge from data educational environments
      [Scheuer12]

➤ Fraud and Intrusion detection [Wang10] [Lee98]

➤ Lie detection and Criminal Investigation [Chen04]

➤ Bio Informatics [Hoffman97]
DATA MINING APPLICATIONS

➤ Market Basket Analysis [Agrawal93]
➤ Future Healthcare
  ➤ Great potential to improve health systems [Obenshain04]
➤ Education
  ➤ Knowledge from data educational environments [Scheuer12]
➤ Fraud and Intrusion detection [Wang10] [Lee98]
➤ Lie detection and Criminal Investigation [Chen04]
➤ Bio Informatics [Hoffman97]
➤ …
# DATA MINING APPLICATIONS

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Mining process</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bioinformatics</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Marketing Mining</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Protein structure prediction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cancer classification</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DNA sequencing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Classes of genes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software Engineering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Program comprehension</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fault localization / prediction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Execution traces</td>
<td>Flow diagram</td>
<td></td>
</tr>
<tr>
<td>Source code</td>
<td>Aurora Project</td>
<td></td>
</tr>
<tr>
<td>Inputs</td>
<td>Mining process</td>
<td>Outputs</td>
</tr>
<tr>
<td>------------------------------</td>
<td>----------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>- shopping basket</td>
<td>Marketing</td>
<td>- Customers behavior</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Frequent products</td>
</tr>
</tbody>
</table>

**DATA MINING APPLICATIONS**
## DATA MINING APPLICATIONS

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Mining process</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>- shopping basket</td>
<td>Marketing</td>
<td>- Customers behavior</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Frequent products</td>
</tr>
<tr>
<td>- DNA sequencing</td>
<td>Bio-Informatics</td>
<td>- Protein structure prediction</td>
</tr>
<tr>
<td>- Classes of genes</td>
<td></td>
<td>- Cancer classification</td>
</tr>
</tbody>
</table>

Aurora Project
DATA MINING APPLICATIONS

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Mining process</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>- shopping basket</td>
<td>Marketing</td>
<td>- Customers behavior</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Frequent products</td>
</tr>
<tr>
<td>- DNA sequencing</td>
<td>Bio-Informatics</td>
<td>- Protein structure prediction</td>
</tr>
<tr>
<td>- Classes of genes</td>
<td></td>
<td>- Cancer classification</td>
</tr>
<tr>
<td>Aurora Project</td>
<td></td>
<td></td>
</tr>
<tr>
<td>- Execution traces</td>
<td>Software Engineering</td>
<td>- Program comprehension</td>
</tr>
<tr>
<td>- Flow diagram</td>
<td></td>
<td>- Fault localization/prediction</td>
</tr>
<tr>
<td>- Source code</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
FREQUENT ITEMSET MINING

[Agrawal et al, 93]
FREQUENT ITEMSET MINING

➤ Aims at finding regularities in datasets (e.g., shopping behavior of customers)

[Agrawal et al, 93]
FREQUENT ITEMSET MINING

➤ Aims at finding regularities in datasets (e.g., shopping behavior of customers)

In market basket analysis:

➤ Find sets of products that are frequently bought together
FREQUENT ITEMSET MINING

➤ Aims at finding regularities in datasets (e.g., shopping behavior of customers)

In market basket analysis:

➤ Find sets of products that are frequently bought together

Often found patterns are expressed as association rules, for example:

➤ If a customer buys bread and wine, then she/he will probably also buy cheese.

[Agrawal et al, 93]
FREQUENT ITEMSET MINING (PROBLEM)

➤ Aims at finding regularities in datasets (e.g., shopping behavior of customers)
FREQUENT ITEMSET MINING (PROBLEM)

➤ Aims at finding regularities in datasets (e.g., shopping behavior of customers)

➤ **Given:**

➤ A set of items \( I = \{i_1, \ldots, i_n\} \)

➤ A set of transactions overs the items \( T = \{t_1, \ldots, t_m\} \)

➤ A minimum support \( \theta \)
FREQUENT ITEMSET MINING (PROBLEM)

➤ Aims at finding regularities in datasets (e.g., shopping behavior of customers)

➤ **Given:**

➤ A set of items \( I = \{i_1, \ldots, i_n\} \)

➤ A set of transactions overs the items \( T = \{t_1, \ldots, t_m\} \)

➤ A minimum support \( \theta \)

➤ **The need:**

➤ The set of itemset \( P \) s.t.:

\[
freq(P) \geq \theta
\]
STANDARD ITEMSET MINING
STANDARD ITEMSET MINING

\[
\begin{align*}
\text{t1:} & \quad B \quad C \quad E \quad F \quad G \quad H \\
\text{t2:} & \quad A \quad D \quad G \\
\text{t3:} & \quad A \quad C \quad D \quad H \\
\text{t4:} & \quad A \quad E \quad F \\
\text{t5:} & \quad B \quad E \quad F \\
\text{t6:} & \quad B \quad E \quad F \quad G
\end{align*}
\]
STANDARD ITEMSET MINING

t1: B C E F G H

t2: A D G

t3: A C D H

t4: A E F

t5: B E F

t6: B E F G
STANDARD ITEMSET MINING

\[ \text{cover}(BEF) = \{t_1, t_5, t_6\} \]
STANDARD ITEMSET MINING

$\text{cover}(BEF) = \{t_1, t_5, t_6\}$

$\text{freq}(BEF) = 50\%$
Brute force enumeration is infeasible

128 items \(10^{68}\) itemsets (atoms in the universe)

\[
\begin{align*}
t1: & \quad \text{B} \quad \text{C} \quad \text{E} \quad \text{F} \quad \text{G} \quad \text{H} \\
t2: & \quad \text{A} \quad \text{D} \quad \text{G} \\
t3: & \quad \text{A} \quad \text{C} \quad \text{D} \quad \text{H} \\
t4: & \quad \text{A} \quad \text{E} \quad \text{F} \\
t5: & \quad \text{B} \quad \text{E} \quad \text{F} \\
t6: & \quad \text{B} \quad \text{E} \quad \text{F} \quad \text{G}
\end{align*}
\]

\[\text{cover}(\text{BEF}) = \{t_1, t_5, t_6\}\]

\[\text{freq}(\text{BEF}) = 50\%\]
STANDARD ITEMSET MINING

- Brute force enumeration is infeasible
  - 128 items $10^{68}$ itemsets (atoms in the universe)
- Several specialised algorithms have been developed:
  Apriori, Eclat, FP-Growth, LCM...

<table>
<thead>
<tr>
<th></th>
<th>t1:</th>
<th>t2:</th>
<th>t3:</th>
<th>t4:</th>
<th>t5:</th>
<th>t6:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>B C E F G H</td>
<td>A D G</td>
<td>A C D H</td>
<td>A E F</td>
<td>B E F</td>
<td>B E F G</td>
</tr>
</tbody>
</table>

$cover(\text{BEF}) = \{t_1, t_5, t_6\}$

$freq(\text{BEF}) = 50\%$
STANDARD ITEMSET MINING

➤ Brute force enumeration is **infeasible**
   ➤ 128 items $10^{68}$ itemsets (atoms in the universe)

➤ Several specialised algorithms have been developed:
  Apriori, Eclat, FP-Growth, LCM…

➤ Dealing with basic user’s constraints:
  Frequency, Condensed representations (closedness, maximality,…), Size…

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>t1:</td>
<td>B</td>
<td>C</td>
<td>E</td>
<td>F</td>
</tr>
<tr>
<td>t2:</td>
<td>A</td>
<td></td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>t3:</td>
<td>A</td>
<td>C</td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>t4:</td>
<td>A</td>
<td></td>
<td>E</td>
<td>F</td>
</tr>
<tr>
<td>t5:</td>
<td>B</td>
<td></td>
<td>E</td>
<td>F</td>
</tr>
<tr>
<td>t6:</td>
<td>B</td>
<td></td>
<td>E</td>
<td>F</td>
</tr>
</tbody>
</table>

$cover(BEF) = \{t_1, t_5, t_6\}$

$freq(BEF) = 50\%$
EXAMPLE
EXAMPLE

$(2^I, \subseteq)$
EXAMPLE

\[(2^I, \subseteq)\]

\[
D
\]

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
\[ \theta = 3 \]

\[ D \]

\[
\begin{array}{cccccc}
 a & b & c & d & e \\
1: & 1 & 0 & 0 & 1 & 1 \\
2: & 0 & 1 & 1 & 1 & 0 \\
3: & 1 & 0 & 1 & 0 & 1 \\
4: & 1 & 0 & 1 & 1 & 1 \\
5: & 1 & 0 & 0 & 0 & 1 \\
6: & 1 & 0 & 1 & 1 & 0 \\
7: & 0 & 1 & 1 & 0 & 0 \\
8: & 1 & 0 & 1 & 1 & 1 \\
9: & 0 & 1 & 1 & 0 & 1 \\
10: & 1 & 0 & 0 & 1 & 1 \\
\end{array}
\]
$\theta = 3$

(2$^I$, $\subseteq$)
\[ \theta = 3 \]

\[
\begin{align*}
M_\theta &= \{ P \in \mathcal{I} \mid \text{freq}(P) \geq \theta \land \forall P' \supset P : \text{freq}(P') < \theta \}\n\end{align*}
\]

\[
D =
\begin{array}{cccccc}
a & b & c & d & e \\
1: & 1 & 0 & 0 & 1 & 1 \\
2: & 0 & 1 & 1 & 1 & 0 \\
3: & 1 & 0 & 1 & 0 & 1 \\
4: & 1 & 0 & 1 & 1 & 1 \\
5: & 1 & 0 & 0 & 0 & 1 \\
6: & 1 & 0 & 1 & 1 & 0 \\
7: & 0 & 1 & 1 & 0 & 0 \\
8: & 1 & 0 & 1 & 1 & 1 \\
9: & 0 & 1 & 1 & 0 & 1 \\
10: & 1 & 0 & 0 & 1 & 1 \\
\end{array}
\]
**EXAMPLE**

\[ \theta = 3 \]

\[ D \]

\[
\begin{array}{cccccc}
|\quad| \quad| \quad| \quad| \quad| \\
\hline
a & b & c & d & e \\
\hline
1: & 1 & 0 & 0 & 1 & 1 \\
2: & 0 & 1 & 1 & 1 & 0 \\
3: & 1 & 0 & 1 & 0 & 1 \\
4: & 1 & 0 & 1 & 1 & 1 \\
5: & 1 & 0 & 0 & 0 & 1 \\
6: & 1 & 0 & 1 & 1 & 0 \\
7: & 0 & 1 & 1 & 0 & 0 \\
8: & 1 & 0 & 1 & 1 & 1 \\
9: & 0 & 1 & 1 & 0 & 1 \\
10: & 1 & 0 & 0 & 1 & 1 \\
\end{array}
\]

\[ (2^I, \subseteq) \]

\[
M_\theta = \{ P \in I | \text{freq}(P) \geq \theta \land \forall P' \supset P : \text{freq}(P') < \theta \}
\]
EXAMPLE

$\theta = 3$

$D$

\[
\begin{array}{cccccc}
\text{a} & \text{b} & \text{c} & \text{d} & \text{e} \\
1: & 1 & 0 & 0 & 1 & 1 \\
2: & 0 & 1 & 1 & 1 & 0 \\
3: & 1 & 0 & 1 & 0 & 1 \\
4: & 1 & 0 & 1 & 1 & 1 \\
5: & 1 & 0 & 0 & 0 & 1 \\
6: & 1 & 0 & 1 & 1 & 0 \\
7: & 0 & 1 & 1 & 0 & 0 \\
8: & 1 & 0 & 1 & 1 & 1 \\
9: & 0 & 1 & 1 & 0 & 1 \\
10: & 1 & 0 & 0 & 1 & 1 \\
\end{array}
\]

$(2^I, \subseteq )$
EXAMPLE

\( \theta = 3 \)

\[
D = \begin{array}{ccccc}
  & a & b & c & d & e \\
1 & 1 & 0 & 0 & 1 & 1 \\
2 & 0 & 1 & 1 & 1 & 0 \\
3 & 1 & 0 & 1 & 0 & 1 \\
4 & 1 & 0 & 1 & 1 & 1 \\
5 & 1 & 0 & 0 & 0 & 1 \\
6 & 1 & 0 & 1 & 1 & 0 \\
7 & 0 & 1 & 1 & 0 & 0 \\
8 & 1 & 0 & 1 & 1 & 1 \\
9 & 0 & 1 & 1 & 0 & 1 \\
10 & 1 & 0 & 0 & 1 & 1 \\
\end{array}
\]

\( (2^I, \subseteq) \)
EXAMPLE

$\theta = 3$

$D$

\[
\begin{array}{cccccc}
 a & b & c & d & e \\
1: & 1 & 0 & 0 & 1 & 1 \\
2: & 0 & 1 & 1 & 1 & 0 \\
3: & 1 & 0 & 1 & 0 & 1 \\
4: & 1 & 0 & 1 & 1 & 1 \\
5: & 1 & 0 & 0 & 0 & 1 \\
6: & 1 & 0 & 1 & 1 & 0 \\
7: & 0 & 1 & 1 & 0 & 0 \\
8: & 1 & 0 & 1 & 1 & 1 \\
9: & 0 & 1 & 1 & 0 & 1 \\
10: & 1 & 0 & 0 & 1 & 1 \\
\end{array}
\]

Closedness

$M_\theta = \{ P \in \mathcal{I} | \text{freq}(P) \geq \theta \land \forall P' \supset P : \text{freq}(P') < \theta \}$
\[ \theta = 3 \]

\[ D = \begin{array}{cccccc}
    a & b & c & d & e \\
1: & 1 & 0 & 0 & 1 & 1 \\
2: & 0 & 1 & 1 & 1 & 0 \\
3: & 1 & 0 & 1 & 0 & 1 \\
4: & 1 & 0 & 1 & 1 & 1 \\
5: & 1 & 0 & 0 & 0 & 1 \\
6: & 1 & 0 & 1 & 1 & 0 \\
7: & 0 & 1 & 1 & 0 & 0 \\
8: & 1 & 0 & 1 & 1 & 1 \\
9: & 0 & 1 & 1 & 0 & 1 \\
10: & 1 & 0 & 0 & 1 & 1 \\
\end{array} \]

\[ M_\theta = \{ P \in \mathcal{I} \mid \text{freq}(P) \geq \theta \land \forall P' \supset P : \text{freq}(P') < \theta \} \]
CONDENSED REPRESENTATION

empty set

item base

maximal (frequent) item sets
CONDENSED REPRESENTATION

empty set

item base

maximal (frequent) item sets

empty set

item base

closed (frequent) item sets
### CONDENSED REPRESENTATION

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Frequent</th>
<th>#Closed</th>
<th>#Maximal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zoo-1</td>
<td>151 807</td>
<td>3 292</td>
<td>230</td>
</tr>
<tr>
<td>Mushroom</td>
<td>155 734</td>
<td>3 287</td>
<td>453</td>
</tr>
<tr>
<td>Lymph</td>
<td>9 967 402</td>
<td>46 802</td>
<td>5 191</td>
</tr>
<tr>
<td>Hepatitis</td>
<td>$27 \cdot 10^7$</td>
<td>1 827 264</td>
<td>189 205</td>
</tr>
</tbody>
</table>
SPECIALIZED VS DECLARATIVE DATA MINING
SPECIALIZED VS DECLARATIVE DATA MINING
SPECIALIZED VS DECLARATIVE DATA MINING

Basic user’s constraints
SPECIALIZED VS DECLARATIVE DATA MINING

Basic user’s constraints

Dataset + Query → Specialised Miner
SPECIALIZED VS DECLARATIVE DATA MINING

Query

Basic user’s constraints

+ dataset

Specialised Miner

Patterns
SPECIALIZED VS DECLARATIVE DATA MINING

Query

Basic user’s constraints

Dataset

Specialised Miner

Patterns

Limitations: Dealing with sophisticated user’s constraints [Wojciechowski and Zakrzewicz, 02]
SPECIALIZED VS DECLARATIVE DATA MINING

**Limitations:** Dealing with sophisticated user’s constraints [Wojciechowski and Zakrzewicz, 02]
SPECIALIZED VS DECLARATIVE DATA MINING

**Limitations:** Dealing with sophisticated user’s constraints [Wojciechowski and Zakrzewicz, 02]
SPECIALIZED VS DECLARATIVE DATA MINING

Limitations: Dealing with sophisticated user’s constraints [Wojciechowski and Zakrzewicz, 02]

Sophisticated user’s constraints

1. Preprocessing

2. Post-processing

Query

Specialised Miner

Patterns
SPECIALIZED VS DECLARATIVE DATA MINING

Limitations: Dealing with sophisticated user’s constraints [Wojciechowski and Zakrzewicz, 02]

1. Dataset preprocessing
2. Post-processing
3. New algorithm
SPECIALIZED VS DECLARATIVE DATA MINING

**Limitations:** Dealing with sophisticated user’s constraints [Wojciechowski and Zakrzewicz, 02]

**Need:** Declarative way to deal with more complex queries

➤ **Declarative data Mining**
**Limitations:** Dealing with sophisticated user’s constraints [Wojciechowski and Zakrzewicz, 02]

**Need:** Declarative way to deal with more complex queries

➤ **Declarative data Mining**
SPECIALIZED VS DECLARATIVE DATA MINING

Limitations: Dealing with sophisticated user’s constraints [Wojciechowski and Zakrzewicz, 02]

Need: Declarative way to deal with more complex queries

→ Declarative data Mining
**SPECIALIZED VS DECLARATIVE DATA MINING**

**Limitations:** Dealing with sophisticated user’s constraints [Wojciechowski and Zakrzewicz, 02]

**Need:** Declarative way to deal with more complex queries

» Declarative data Mining
SPECIALISED VS DECLARATIVE DATA MINING
SPECIALISED VS DECLARATIVE DATA MINING
SPECIALISED VS DECLARATIVE DATA MINING

Specialised is the winner!
Specialised is the winner!
Specialised vs Declarative Data Mining

Specialised is the winner!

Declarative is the winner!
SPECIALISED VS DECLARATIVE DATA MINING
## Preprocessing + Specialised step vs Declarative

<table>
<thead>
<tr>
<th>Instances</th>
<th>(#I_i)</th>
<th>(#T_i)</th>
<th>((lb_i, ub_i))</th>
<th>((lb_T, ub_T))</th>
<th>(#D)</th>
<th>(#FCIs)</th>
<th>PP-LCM</th>
<th>CP-ITEMSET</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zoo_70_6</td>
<td>6</td>
<td>10</td>
<td>(2,3)</td>
<td>(2,3)</td>
<td>5,775</td>
<td>8</td>
<td>39.69</td>
<td>1.75</td>
</tr>
<tr>
<td>Zoo_50_11</td>
<td>6</td>
<td>10</td>
<td>(3,4)</td>
<td>(3,4)</td>
<td>11,550</td>
<td>9</td>
<td>88.66</td>
<td>3.36</td>
</tr>
<tr>
<td>Zoo_85_5</td>
<td>6</td>
<td>10</td>
<td>(2,6)</td>
<td>(2,10)</td>
<td>57,741</td>
<td>8</td>
<td>521.89</td>
<td>31.86</td>
</tr>
<tr>
<td>Primary_82.5</td>
<td>3</td>
<td>12</td>
<td>(2,3)</td>
<td>(2,10)</td>
<td>16,280</td>
<td>8</td>
<td>199.58</td>
<td>36.13</td>
</tr>
<tr>
<td>Vote_70_6</td>
<td>6</td>
<td>29</td>
<td>(2,3)</td>
<td>(2,3)</td>
<td>142,100</td>
<td>2</td>
<td>TO</td>
<td>118.67</td>
</tr>
<tr>
<td>Vote_72.5</td>
<td>8</td>
<td>29</td>
<td>(2,3)</td>
<td>(2,3)</td>
<td>341,040</td>
<td>2</td>
<td>TO</td>
<td>201.79</td>
</tr>
<tr>
<td>Mushroom_80.5</td>
<td>17</td>
<td>12</td>
<td>(2,2)</td>
<td>(2,2)</td>
<td>8,976</td>
<td>10</td>
<td>446.42</td>
<td>102.68</td>
</tr>
<tr>
<td>Mushroom_82.5</td>
<td>17</td>
<td>12</td>
<td>(2,2)</td>
<td>(3,3)</td>
<td>29,920</td>
<td>7</td>
<td>TO</td>
<td>455.19</td>
</tr>
<tr>
<td>Chess_90_16</td>
<td>5</td>
<td>34</td>
<td>(2,3)</td>
<td>(2,2)</td>
<td>11,220</td>
<td>3</td>
<td>286.42</td>
<td>87.22</td>
</tr>
</tbody>
</table>

TO: timeout
SPECIALISED VS DECLARATIVE DATA MINING
## Specialised + postprocessing vs Declarative

<table>
<thead>
<tr>
<th>Instances</th>
<th>ub</th>
<th>lb</th>
<th>ECLAT-Z-PP</th>
<th>SAT</th>
<th>CP</th>
<th>#Tot</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zoo_5</td>
<td>2</td>
<td>11</td>
<td>479.26</td>
<td>3.92</td>
<td>0.36</td>
<td>27</td>
</tr>
<tr>
<td>Zoo_5</td>
<td>1</td>
<td>9</td>
<td>491.48</td>
<td>0.17</td>
<td>0.06</td>
<td>12</td>
</tr>
<tr>
<td>Vote_5</td>
<td>4</td>
<td>8</td>
<td>37.69</td>
<td>282.25</td>
<td>0.66</td>
<td>13</td>
</tr>
<tr>
<td>Vote_5</td>
<td>1</td>
<td>2</td>
<td>38.49</td>
<td>1.41</td>
<td>0.05</td>
<td>23</td>
</tr>
<tr>
<td>Anneal_80</td>
<td>2</td>
<td>13</td>
<td>1567.48</td>
<td>1.14</td>
<td>0.26</td>
<td>76</td>
</tr>
<tr>
<td>Anneal_80</td>
<td>1</td>
<td>12</td>
<td>1622.19</td>
<td>0.53</td>
<td>0.15</td>
<td>73</td>
</tr>
<tr>
<td>Chess_60</td>
<td>2</td>
<td>9</td>
<td>280.60</td>
<td>2.17</td>
<td>0.20</td>
<td>20</td>
</tr>
<tr>
<td>Chess_60</td>
<td>1</td>
<td>8</td>
<td>284.22</td>
<td>1.07</td>
<td>0.08</td>
<td>24</td>
</tr>
<tr>
<td>Mushroom_10</td>
<td>1</td>
<td>11</td>
<td>249.00</td>
<td>47.52</td>
<td>0.07</td>
<td>14</td>
</tr>
<tr>
<td>Connect_90</td>
<td>1</td>
<td>11</td>
<td>61.80</td>
<td>30.41</td>
<td>0.26</td>
<td>12</td>
</tr>
<tr>
<td>T10_0.02</td>
<td>1</td>
<td>11</td>
<td>84.47</td>
<td>TO</td>
<td>5.44</td>
<td>0</td>
</tr>
<tr>
<td>T40_0.1</td>
<td>1</td>
<td>11</td>
<td>TO</td>
<td>TO</td>
<td>8.33</td>
<td>39</td>
</tr>
<tr>
<td>Pumsb_80</td>
<td>1</td>
<td>12</td>
<td>741.49</td>
<td>OOM</td>
<td>0.34</td>
<td>32</td>
</tr>
</tbody>
</table>

TO: timeout     OOM: out-of-memory

---

17
CONCLUSIONS (PART I)
CONCLUSIONS (PART I)

➤ Specialised methods are suitable for:
  ➤ Enumerating Patterns
  ➤ Taking into account classic constraints (simple queries)
CONCLUSIONS (PART I)

➤ Specialised methods are suitable for:
  ➤ Enumerating Patterns
  ➤ Taking into account classic constraints (simple queries)

➤ Declarative methods are suitable for:
  ➤ Taking into account user's constraints (complex queries)
  ➤ Iterative data mining process
CONCLUSIONS (PART I)

➤ Specialised methods are suitable for:
  ➤ Enumerating Patterns
  ➤ Taking into account classic constraints (simple queries)

➤ Declarative methods are suitable for:
  ➤ Taking into account user’s constraints (complex queries)
  ➤ Iterative data mining process

Time left?
FAULT LOCALISATION
FAULT LOCALISATION

➤ The need: identify a subset of statements that are susceptible to explain a fault in a program

➤ Precision <=> Efficiency
FAULT LOCALISATION

➤ The need: identify a subset of statements that are susceptible to explain a fault in a program

➤ Precision <=> Efficiency

➤ Spectrum-based approaches: (ranking metrics - suspiciousness score)

➤ Tarantula [Jones and Harrold 05]

➤ Ochiai [Abreu et al. 07]

➤ Jaccard [Abreu et al. 07]

➤ …
FAULT LOCALISATION (MOTIVATIONS)

(a) Tarantula  
(b) Ochiai  
(c) Jaccard

[Susceptibility]
FAULT LOCALISATION (MOTIVATIONS)

➤ Pros: Quick localisation
Fault Localisation (Motivations)

➤ **Pros:** Quick localisation

➤ **Cons:** independent evaluation of each statement at the expense of accuracy
FAULT LOCALISATION (MOTIVATIONS)
# FAULT LOCALISATION (MOTIVATIONS)

Program: Character counter

```c
function count (char *s) {
    int let, dig, other, i = 0;
    char c;
    e1: while (c = s[i++]) {
        if('A'<=c && 'Z'>=c) let += 2; // fault -
        else if ( 'a'<=c && 'z'>=c ) let += 1;
        else if ( '0'<=c && '9'>=c ) dig += 1;
        else if (isprint (c)) other += 1;
    }
    e10: printf("%d %d %d\n", let, dig, other);
}
```

<table>
<thead>
<tr>
<th>Test cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>tc1</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>

Passing/Failing

F F F F F P P
Fault Localisation (Motivations)

<table>
<thead>
<tr>
<th>Program: Character counter</th>
<th>Test cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>function count (char *s) {</td>
<td>tc1</td>
</tr>
<tr>
<td>int let, dig, other, i = 0;</td>
<td></td>
</tr>
<tr>
<td>char c;</td>
<td></td>
</tr>
<tr>
<td>e1: while (c = s[i++]) {</td>
<td>1</td>
</tr>
<tr>
<td>e2: if (’A’&lt;=c &amp;&amp; ’Z’&gt;=c)</td>
<td>1</td>
</tr>
<tr>
<td>e3: let += 2; // fault -</td>
<td>1</td>
</tr>
<tr>
<td>e4: else if ( ’a’&lt;=c &amp;&amp; ’z’&gt;=c )</td>
<td>1</td>
</tr>
<tr>
<td>e5: let += 1;</td>
<td>1</td>
</tr>
<tr>
<td>e6: else if ( ’0’&lt;=c &amp;&amp; ’9’&gt;=c )</td>
<td>1</td>
</tr>
<tr>
<td>e7: dig += 1;</td>
<td>0</td>
</tr>
<tr>
<td>e8: else if ( isprint (c))</td>
<td>1</td>
</tr>
<tr>
<td>e9: other += 1;</td>
<td>1</td>
</tr>
<tr>
<td>e10: printf(&quot;%d %d %d\n&quot;, let, dig, other);</td>
<td>1</td>
</tr>
<tr>
<td>Passing/Failing</td>
<td>F</td>
</tr>
</tbody>
</table>
FAULT LOCALISATION (MOTIVATIONS)

---

**Program**: Character counter

```c
function count (char *s) {
    int let, dig, other, i = 0;
    char c;
    e1: while (c = s[i++]) {
    e2:     if('A'<=c && 'Z'>=c) 1 1 1 1 1 1 0 1
    e3:       let += 2; // fault - 1 1 1 1 1 1 0 0
    e4:       else if ( 'a'<=c && 'z'>=c ) 1 1 1 1 1 0 0 1
    e5:       let += 1; 1 1 0 0 1 0 0 0
    e6:       else if ( '0'<=c && '9'>=c ) 1 1 1 1 0 0 0 1
    e7:         dig += 1; 0 1 0 1 0 0 0 0
    e8:       else if (isprint (c)) 1 0 1 0 0 0 0 1
    e9:         other += 1; 1 0 1 0 0 0 0 1
    e10:      printf("%d %d %d\n", let, dig, other);}

Passing/Failing
```

<table>
<thead>
<tr>
<th>Test cases</th>
<th>tc1</th>
<th>tc2</th>
<th>tc3</th>
<th>tc4</th>
<th>tc5</th>
<th>tc6</th>
<th>tc7</th>
<th>tc8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 1 1 1 1 1 1 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 1 1 1 1 0 0 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1 1 1 0 0 1 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 1 0 1 0 0 0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0 1 0 0 0 0 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 0 1 0 0 0 0 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1 1 1 1 1 1 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Passing/Failing: F F F F F F P P
FAULT LOCALISATION (MOTIVATIONS)
FAULT LOCALISATION (MOTIVATIONS)

Pros: Quick localisation
FAULT LOCALISATION (MOTIVATIONS)

➤ **Pros:** Quick localisation

➤ **Cons:** independent evaluation of each statement at the expense of accuracy
FAULT LOCALISATION (MOTIVATIONS)

- **Pros:** Quick localisation
- **Cons:** independent evaluation of each statement at the expense of accuracy
- **Need:** more finer-grained localisation, taking into account user’s constraints
FAQULT LOCALISATION (MOTIVATIONS)

➤ **Pros:** Quick localisation

➤ **Cons:** independent evaluation of each statement at the expense of accuracy

➤ **Need:** more finer-grained localisation, taking into account user’s constraints

➤ **How:** Use of Declarative Data Mining
## Fault Localisation (Motivations)

Program: Character counter

```
int let, dig, other, i = 0;
char c;

function count (char *s) {
    while (c = s[i++]) {
        let += 2;  // fault -
        else if ( 'a'<=c && 'z'>=c )
            let += 1;
        else if ( '0'<=c && '9'>=c )
            dig += 1;
        else if (isprint (c))
            other += 1;
    }
    printf("%d %d %d\n", let, dig, other);
}
```

<table>
<thead>
<tr>
<th>Test cases</th>
<th>tc1</th>
<th>tc2</th>
<th>tc3</th>
<th>tc4</th>
<th>tc5</th>
<th>tc6</th>
<th>tc7</th>
<th>tc8</th>
</tr>
</thead>
<tbody>
<tr>
<td>tc1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>tc2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>tc3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>tc4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>tc5</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>tc6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>tc7</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>tc8</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>tc9</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>tc10</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>tc11</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>tc12</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Passing/Failing

| Passing/Failing | F | F | F | F | F | F | P | P |

23
### FAULT LOCALISATION (MOTIVATIONS)

<table>
<thead>
<tr>
<th>Program: Character counter</th>
<th>Test cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>function count (char *s) {</td>
<td>tc1</td>
</tr>
<tr>
<td>int let, dig, other, i = 0;</td>
<td></td>
</tr>
<tr>
<td>char c;</td>
<td></td>
</tr>
<tr>
<td>e1: while (c = s[i++]) {</td>
<td>1</td>
</tr>
<tr>
<td>e2: if (’A’&lt;=c &amp;&amp; ’Z’&gt;=c)</td>
<td>1</td>
</tr>
<tr>
<td>e3: let += 2; //fault -</td>
<td>1</td>
</tr>
<tr>
<td>e4: else if ( ’a’&lt;=c &amp;&amp; ’z’&gt;=c )</td>
<td>1</td>
</tr>
<tr>
<td>e5: let += 1;</td>
<td>1</td>
</tr>
<tr>
<td>e6: else if ( ’0’&lt;=c &amp;&amp; ’9’&gt;=c )</td>
<td>1</td>
</tr>
<tr>
<td>e7: dig += 1;</td>
<td>0</td>
</tr>
<tr>
<td>e8: else if (isprint (c))</td>
<td>1</td>
</tr>
<tr>
<td>e9: other += 1;</td>
<td>1</td>
</tr>
<tr>
<td>e10: printf(&quot;%d %d %d\n&quot;, let, dig, other);}</td>
<td>1</td>
</tr>
</tbody>
</table>

Passing/Failing

Fault localisation = Mining Task
PATTERN SUSPICIOUSNESS DEGREE (PSD)
PATTERN SUSPICIOUSNESS DEGREE (PSD)

➤ PSD function. Given a pattern P of a program:

\[ PSD(P) = freq^{-}(P) + \frac{|FAIL|-freq^{+}(P)}{|PASS|+1} \]
PATTERN SUSPICIOUSNESS DEGREE (PSD)

➤ PSD function. Given a pattern P of a program:

$$PSD(P) = \text{freq}^-(P) + \frac{|\text{FAIL}| - \text{freq}^+(P)}{|\text{PASS}| + 1}$$

➤ PSD-dominance relation. Given two patterns $P_i$ and $P_j$

$$P_i \succ_{PSD} P_j \iff PSD(P_i) > PSD(P_j)$$
PATTERN SUSPICIOUSNESS DEGREE (PSD)

➤ PSD function. Given a pattern P of a program:

$$PSD(P) = freq^-(P) + \frac{|FAIL| - freq^+(P)}{|PASS| + 1}$$

➤ PSD-dominance relation. Given two patterns $P_i$ and $P_j$

$$P_i \succ_{PSD} P_j \iff PSD(P_i) > PSD(P_j)$$

➤ Top-k suspicious patterns.

$$\text{top-k} = \{ P | \exists P_1, \ldots, P_k : \forall 1 \leq j \leq k, \ P_j \succ_{PSD} P \}$$
FCP-MINER TOOL (SOME RESULTS)
CONCLUSIONS (PART II)

➤ Software Testing/Program comprehension tasks can be tackled using Data Mining
  ➤ Trace analysis
  ➤ Test suites mining
  ➤ Source code mining
  ➤ ...
CONCLUSIONS (PART II)

➤ Software Testing/Program comprehension tasks can be tackled using Data Mining
  ➤ Trace analysis
  ➤ Test suites mining
  ➤ Source code mining
  ➤ …
➤ Think about using Declarative methods in Software Testing
CONCLUSIONS (PART II)

➤ Software Testing/Program comprehension tasks can be tackled using Data Mining

➤ Trace analysis
➤ Test suites mining
➤ Source code mining
➤ ...

➤ Think about using Declarative methods in Software Testing