Efficient sequential and temporal pattern mining

PhD Thesis
Natalia Mordvanyuk Maltseva
The doctoral program in Technology of the University of Girona.

SUPERVISED BY:
PHD BEATRIZ LÓPEZ
PHD ALBERT BIFET
Introduction
Vepreco: sequential patterns
vertTIRP: temporal patterns
TA4L: preprocessing
Conclusions
Introduction
Vepreco: sequential patterns
vertTIRP: temporal patterns
TA4L: preprocessing
Conclusions
Introduction: Motivation

- <<cold, obesity>, <flu, obesity>, <diabetes>, <flu, diabetes>, <retinopathy, diabetic nephropathy>, <glaucoma>>
- <<urinary tract infection, obesity>, <obesity, diabetes>, <retinopathy, diabetic nephropathy>, <glaucoma>>
- <<obesity, hypertension>, <obesity, diabetes>, <retinopathy, diabetic nephropathy>, <glaucoma>>

Pattern discovery

- Obesity, diabetes, (retinopathy, diabetic nephropathy), glaucoma

Sequential pattern mining
Frequent pattern mining

Computationally expensive
Introduction: Motivation

Adding TIME!

Pattern discovery

Adding computational complexity!

Temporal pattern mining

patient 1
<<cold, obesity>, <flu, obesity>, <diabetes>, <flu, diabetes>, <retinopathy, diabetic nephropathy>, <glaucoma>>

01/01/2018 01/01/2021

patient 2
<<urinary tract infection, obesity>, <obesity, diabetes>, <retinopathy, diabetic nephropathy>, <glaucoma>>

01/01/2018 01/01/2021

patient n
<<obesity, hypertension>, <obesity, diabetes>, <retinopathy, diabetic nephropathy>, <glaucoma>>

01/01/2018 01/01/2021
Introduction: Motivation

<<cold, obesity>, <flu, obesity>, <diabetes>, <flu, diabetes>, <retinopathy, diabetic nephropathy>, <glaucoma>>

01/01/2018 | 01/01/2021

<<urinary tract infection, obesity>, <obesity, diabetes>, <retinopathy, diabetic nephropathy>, <glaucoma>>

01/01/2018 | 01/01/2021

<<obesity, hypertension>, <obesity, diabetes>, <retinopathy, diabetic nephropathy>, <glaucoma>>

Preprocessing.
Prepare the data for the sequential and temporal pattern mining

Lack of methods

<table>
<thead>
<tr>
<th>value</th>
<th>timestamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.7234</td>
<td>01/01/2018</td>
</tr>
<tr>
<td>0.8792</td>
<td>01/01/2018</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>0.7878</td>
<td>01/01/2021</td>
</tr>
</tbody>
</table>
Introduction: Objectives

➢ Improve pattern mining algorithms:
  ● Sequential
  ● Temporal

➢ Formalization of preprocessing
Introduction: Contributions

➢ Improve pattern mining algorithms:
  ● Sequential
  ● Temporal

➢ Formalization of preprocessing

VEPRECO

vertTIRP

TA4L
Introduction

Vepreco: sequential patterns

vertTIRP: temporal patterns

TA4L: preprocessing

Conclusions
VEPRECO: Problem statement

The problem of sequential pattern mining is to discover the set of all frequent sequences $S$ in the sequence database $SDB$ at the given minimum support $min\_sup$.
VEPRECO: State of the art

Categories of sequential pattern mining (SPM) based on the taxonomy analysis:

<table>
<thead>
<tr>
<th>Apriori-based</th>
<th>Pattern-growth</th>
<th>Early-pruning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search</td>
<td>Database</td>
<td>Candidates generation</td>
</tr>
<tr>
<td>BFS</td>
<td>horizontal</td>
<td>Generate-and-test</td>
</tr>
<tr>
<td>DFS</td>
<td>projected</td>
<td>Based on DB</td>
</tr>
<tr>
<td>BFS/DFS</td>
<td>vertical</td>
<td>Based on position</td>
</tr>
</tbody>
</table>
VEPRECO: State of the art

Categories of sequential pattern mining (SPM) based on the taxonomy analysis:

<table>
<thead>
<tr>
<th>Search</th>
<th>Database</th>
<th>Candidates generation</th>
<th>Candidate pruning</th>
<th>Multiple DB scans</th>
<th>Suffix/Prefix growth</th>
<th>Compressed representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apriori-based</td>
<td>BFS</td>
<td>horizontal</td>
<td>Generate-and-test</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Pattern-growth</td>
<td>DFS</td>
<td>projected</td>
<td>Based on DB</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Early-pruning</td>
<td>BFS/DFS</td>
<td>vertical</td>
<td>Based on position</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Contributions of VEPRECO
VEPRECO: Contributions of VEPRECO

1. Memory efficient approach
2. Pattern candidates generation
3. Preprunning
VEPRECO: SPM approach

Sequential database:

\(<b)(ab)(ab)>
\(<a)(ab)>
\(<ab)>

min_sup = 0.6
VEPRECO: 1. Memory efficient approach

### SID | Sequences
---|---
1 | \(<b (a \ b) (a \ b)\>
2 | \(<a (a \ b)\>
3 | \(<(a \ b)\>

Sequential database (SDB)

### a-DictMap

<table>
<thead>
<tr>
<th>Pattern</th>
<th>locations</th>
<th>length</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Keys (sequences): 1, 2, 3</td>
<td>Values (transactions): 2, 3</td>
<td>1</td>
</tr>
</tbody>
</table>

### b-DictMap

<table>
<thead>
<tr>
<th>Pattern</th>
<th>locations</th>
<th>length</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>Keys (sequences): 1, 2, 3</td>
<td>Values (transactions): 1, 2, 3</td>
<td>1</td>
</tr>
</tbody>
</table>
**VEPRECO**: 1. Memory efficient approach

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Locations</th>
<th>Length</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1, 2, 3</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

**s-extension of a with b**

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Locations</th>
<th>Length</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>ab</td>
<td>1, 2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

**Sequential database:**
- (b), (a,b), (a,b)
- (a), (a,b)
- (a,b)
VEPRECO: 1. Memory efficient approach

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Locations</th>
<th>Length</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1</td>
<td>2,3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1,2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Locations</th>
<th>Length</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>1</td>
<td>1,2,3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

i-extension of a with b

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Locations</th>
<th>Length</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>(ab)</td>
<td>1</td>
<td>2,3</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Sequential database:
1. (b)
2. (a,b)
3. (a)
4. (a,b)
5. (a,b)
VEPRECO: 2. Pattern candidates generation

Using c-extension

Previous approaches
a,b,c,d,f,k  s-extend
a,b,c,e,g,j  i-extend

We save 6 iterations!
VEPRECO: 3. Preprunning. Strategy 1: based on the last item

![Diagram showing the preprunning strategy with nodes and child nodes labeled a, ab, and a(ab).]
Selection of i-candidates for an s-extended pattern.

Given an s-extended pattern \( P_s = ab \), a set of i-candidates for \( ab \) is the intersection of s-candidates of \( a \) with the i-candidates of \( b \).

**PROBLEM**

Given the following SDB, what are the i-candidates to extend \(< a \ b >\)?

**Sequential database**

\[
(b \ c \ e) \ a \ (b \ d) \ a \\
(b \ c) \ a \ (b \ d) \ a \\
(b \ e) \ a \ b 
\]

\[\text{min}_\text{sup} = 0.6\]

**SOLUTION**

- s-candidates of \( a \)
- i-candidates of \( b \)

- i-candidate to extend \(< a \ b >\) is \{d\} *

\( ab \) with \( d \) results in the frequent extension \( a(bd) \)

*In the case of CM-SPAM, the candidates are all the i-candidates of item \( b=\{c,d,e\} \)
**VEPRECO: Experimental setup**

<table>
<thead>
<tr>
<th>SDB name</th>
<th>NS</th>
<th>SL</th>
<th>TL</th>
<th>NI</th>
</tr>
</thead>
<tbody>
<tr>
<td>data.NS_1000.SL5.TL_5.NI_100</td>
<td>1,000</td>
<td>5</td>
<td>5</td>
<td>100</td>
</tr>
<tr>
<td>data.NS_10000.SL5.TL_5.NI_100</td>
<td>10,000</td>
<td>5</td>
<td>5</td>
<td>100</td>
</tr>
<tr>
<td>data.NS_20000.SL5.TL_5.NI_100</td>
<td>20,000</td>
<td>5</td>
<td>5</td>
<td>100</td>
</tr>
<tr>
<td>data.NS_10000.SL2.TL_2.NI_1000</td>
<td>10,000</td>
<td>2</td>
<td>2</td>
<td>1,000</td>
</tr>
<tr>
<td>data.NS_10000.SL4.TL_4.NI_1000</td>
<td>10,000</td>
<td>4</td>
<td>4</td>
<td>1,000</td>
</tr>
<tr>
<td>data.NS_10000.SL8.TL_8.NI_1000</td>
<td>10,000</td>
<td>8</td>
<td>8</td>
<td>1,000</td>
</tr>
</tbody>
</table>

**Synthetic datasets**

<table>
<thead>
<tr>
<th>SDB name</th>
<th>NS</th>
<th>SL</th>
<th>TL</th>
<th>NI</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSNBC</td>
<td>31,790</td>
<td>3.82</td>
<td>2.48</td>
<td>17</td>
</tr>
<tr>
<td>BIBLE</td>
<td>36,369</td>
<td>4.41</td>
<td>4.47</td>
<td>13,905</td>
</tr>
<tr>
<td>FIFA</td>
<td>20,450</td>
<td>5.7</td>
<td>5.71</td>
<td>2,990</td>
</tr>
<tr>
<td>LEVIATHAN</td>
<td>5,834</td>
<td>5.43</td>
<td>5.53</td>
<td>9,025</td>
</tr>
<tr>
<td>SIGN</td>
<td>730</td>
<td>7.09</td>
<td>7.28</td>
<td>267</td>
</tr>
</tbody>
</table>

**Real datasets**
VEPRECO: Experimental setup

- Efficiency
  - Memory efficient
  - Candidate generation
  - Pruning strategies

- A deeper analysis on Pattern candidate generation with the c extension.

- Further exploration of pre pruning strategies.
  - Strategy 1: based on the last item
  - Strategy 2: based on the last 2 items
VEPRECO: Experimental setup

- Efficiency
  - Memory efficient
  - Candidate generation
  - Pruning strategies

- A deeper analysis on Pattern candidate generation with the c extension.

- Further exploration of pre pruning strategies.
  - Strategy 1: based on the last item
  - Strategy 2: based on the last 2 items
VEPRECO: Results

Real datasets: Time
**VEPRECO: Results**

Real datasets: Memory
VEPRECO: Discussion

- VEPRECO consumes less time and memory than the CM-SPAM.

- Memory efficient approach: DictMap structure reduced both runtime and memory usage.

- Pattern candidate generation:
  - C extension is dataset dependent.

- Pruning strategies:
  - Effective for large NS and NI, with similar SL and TL.
CONTENTS

Introduction
Vepreco: sequential patterns
vertTIRP: temporal patterns
TA4L: preprocessing
Conclusions
vertTIRP: Problem statement

Sequence 1
- 08:00 A 10:00
- 08:00 B 12:00
- 11:00 C 13:00
- 14:00 C 17:00
- 16:00 A 17:00
- 16:00 B 18:00

Sequence 2
- 10:00 C 13:00
- 11:00 B 14:00

Sequence 3
- 12:00 A 13:00
- 14:00 C 15:00

vs – vertical support
mhs – mean horizontal support
md – mean duration (hours)

min sup = 0.33

vs 1.0
mhs 0.31
md 1.33

vs 1.0
mhs 0.31
md 3.0

vs 1.0
mhs 0.39
md 2.33

vs 0.66
mhs 0.33
md 3.0

vs 0.66
mhs 0.29
md 4.0

vs 0.33
mhs 0.29
md 4.0

vs 0.33
mhs 0.25
md 5.0

vs 0.33
mhs 0.33
md 5.0

vs 0.33
mhs 0.25
md 5.0

vs 0.33
mhs 0.25
md 4.0

vs 0.33
mhs 0.29
md 4.0

vs 0.33
mhs 0.25
md 5.0

vs – vertical support
mhs – mean horizontal support
md – mean duration (hours)
vertTIRP: Temporal relations

<table>
<thead>
<tr>
<th>Relation</th>
<th>Description</th>
<th>Timeline</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before (b)</td>
<td>08:00 A to 08:45</td>
<td>08:00 B to 10:00</td>
</tr>
<tr>
<td>Meets (m)</td>
<td>08:00 A to 09:00</td>
<td>08:00 B to 10:00</td>
</tr>
<tr>
<td>Overlaps (o)</td>
<td>08:00 A to 09:30</td>
<td>08:30 B to 10:00</td>
</tr>
<tr>
<td>Contains (c)</td>
<td>08:00 A to 10:00</td>
<td>08:30 B to 09:30</td>
</tr>
<tr>
<td>Finish-by (f)</td>
<td>08:00 A to 10:00</td>
<td>09:00 B to 10:00</td>
</tr>
<tr>
<td>Equal (e)</td>
<td>08:00 A to 10:00</td>
<td></td>
</tr>
<tr>
<td>Starts (s)</td>
<td>08:00 A to 09:00</td>
<td>08:00 B to 10:00</td>
</tr>
</tbody>
</table>
vertTIRP: State of the art

(Kam & Fu, 2000)
"(a overlaps c) overlaps b"
"a overlaps (c during b)"

(Wu & Chen, 2007)
a+ < b+ < c+ < c- < b-
+ start time
- end time

(Patel et al., 2008)
"(a overlap [00010] b) [10010] contain c"
The vectors’ meaning is [cfmos]

(Papapetrou et al., 2009)
overlap(a,b)*overlap(a,c)*contain(b,c)
→ Epsilon

(Moskovitch & Shahar, 2009, 2015, 2021)
→ Transitivity
Algorithms: KarmaLego, DharmaLego, TIRPClo

Introduction
VEPRECO
vertTIRP
vertTIRP:
State of the art

Conclusions
VertTIRP: State of the art

Contributions of vertTIRP

Apriori-based
Pattern-growth
Early pruning

(Kam & Fu, 2000)
"(a overlaps c) overlaps b"
"a overlaps (c during b)"

(Wu & Chen, 2007)
a+ < b+ < c+ < c- < b-
+ start time
- end time

(Patel et al., 2008)
"(a overlap [00010] b) [10010] contain c"
The vectors’ meaning is [cfmos]

(Papapetrou et al., 2009)
overlap(a,b)*overlap(a,c)*contain(b,c)
→ Epsilon

(Yang et al., 2017)
(a+)(b+)(c+)(c-)(b-)

(Moskovitch & Shahar, 2009, 2015, 2021)
→ Transitivity, Epsilon
Algorithms: KarmaLego, DharmaLego, TIRPClo

VertTIRP: State of the art

Introduction
VEPRECO
VertTIRP
TA4L
Conclusions
vertTIRP: Contributions

1. Temporal relations revision
2. Transitivity table revision
3. Efficient TIRPs representation
4. Constraints
5. Pairing strategies

Equals

Symbolic time interval A
Symbolic time interval B

$\varepsilon = 5\text{'}$
### vertTIRP: 1. Temporal relations revision. Epsilon effect.

<table>
<thead>
<tr>
<th>Relation</th>
<th>0</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before (b)</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Meets (m)</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Overlaps (o)</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Contains (c)</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Finish-by (f)</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Equal (e)</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Starts (s)</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Left contain (l)</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

**Introduction**

**vertTIRP:**

1. Temporal relations revision.

**Epsilon effect.**

<table>
<thead>
<tr>
<th>08:00</th>
<th>A</th>
<th>10:00</th>
</tr>
</thead>
<tbody>
<tr>
<td>08:03</td>
<td>B</td>
<td>09:15</td>
</tr>
</tbody>
</table>

Contains, for $\varepsilon=0'$

Left contain, for $\varepsilon=5'$

**Need of a new temporal relationship**
vertTIRP: 2. Transitivity table revision.

If $\epsilon=0'$; 
$r_1(A,B)=o$; 
$r_2(B,C)=f$;
Then, $r_3(A,C)=bmo$

**Transitivity table (original Allen)**

<table>
<thead>
<tr>
<th>$r_2(I^B, I^C)$</th>
<th>b</th>
<th>c</th>
<th>o</th>
<th>m</th>
<th>s</th>
<th>f</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>b &quot;before&quot;</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>c &quot;contains&quot;</td>
<td>b</td>
<td>c</td>
<td>c</td>
<td>f</td>
<td>o</td>
<td>c</td>
<td>o</td>
</tr>
<tr>
<td>m &quot;meets&quot;</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>m</td>
<td>b</td>
<td>m</td>
</tr>
<tr>
<td>s &quot;starts&quot;</td>
<td>b</td>
<td>b</td>
<td>c</td>
<td>m</td>
<td>o</td>
<td>b</td>
<td>m</td>
</tr>
<tr>
<td>f &quot;finished-by&quot;</td>
<td>b</td>
<td>c</td>
<td>o</td>
<td>m</td>
<td>o</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>e &quot;equal&quot;</td>
<td>b</td>
<td>c</td>
<td>o</td>
<td>m</td>
<td>s</td>
<td>f</td>
<td>e</td>
</tr>
</tbody>
</table>

**Example**

- **BEFORE** $r_3(A,C)=b$
  - 08:00 A 08:20
  - 08:10 B 08:40
  - 08:30 C 08:40

- **MEETS** $r_3(A,C)=m$
  - 08:00 A 08:30
  - 08:10 B 08:40
  - 08:20 C 08:40

- **OVERLAPS** $r_3(A,C)=o$
  - 08:00 A 08:30
  - 08:10 B 08:40
  - 08:20 C 08:40
vertTIRP: 2. Transitivity table revision.

Before: \( r_3(A, C) = b \)

\[
\begin{array}{ccc}
08:00 & A & 08:20 \\
08:10 & B & 08:40 \\
08:30 & C & 08:40 \\
\end{array}
\]

Meets: \( r_3(A, C) = m \)

\[
\begin{array}{ccc}
08:00 & A & 08:30 \\
08:10 & B & 08:40 \\
08:20 & C & 08:40 \\
\end{array}
\]

Overlaps: \( r_3(A, C) = o \)

\[
\begin{array}{ccc}
08:00 & A & 08:30 \\
08:10 & B & 08:40 \\
08:20 & C & 08:40 \\
\end{array}
\]

Finished by: \( r_3(A, C) = f \)

\[
\begin{array}{ccc}
08:00 & A & 08:30 \\
08:10 & B & 08:40 \\
08:20 & C & 08:35 \\
\end{array}
\]

If \( \varepsilon = 5' \); \( r_1(A, B) = o; r_2(B, C) = f; \) According the Allen's table: \( r_3(A, C) = bmo \times \) When the \( r_3(A, C) = bmof \)
### Transitivity table for eps=0

<table>
<thead>
<tr>
<th>$r_2(I^B, I^C)$</th>
<th>b</th>
<th>c</th>
<th>o</th>
<th>m</th>
<th>s</th>
<th>f</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r_1(I^A, I^B)$</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>b &quot;before&quot;</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>c &quot;contains&quot;</td>
<td>b</td>
<td>c</td>
<td>e</td>
<td>f</td>
<td>o</td>
<td>c</td>
<td>f</td>
</tr>
<tr>
<td>o &quot;overlaps&quot;</td>
<td>b</td>
<td>b</td>
<td>c</td>
<td>f</td>
<td>m</td>
<td>o</td>
<td>o</td>
</tr>
<tr>
<td>m &quot;meets&quot;</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>m</td>
<td>m</td>
</tr>
<tr>
<td>s &quot;starts&quot;</td>
<td>b</td>
<td>b</td>
<td>c</td>
<td>f</td>
<td>m</td>
<td>o</td>
<td>b</td>
</tr>
<tr>
<td>f &quot;finished-by&quot;</td>
<td>b</td>
<td>c</td>
<td>e</td>
<td>o</td>
<td>m</td>
<td>o</td>
<td>f</td>
</tr>
<tr>
<td>e &quot;equal&quot;</td>
<td>b</td>
<td>c</td>
<td>o</td>
<td>m</td>
<td>s</td>
<td>f</td>
<td>e</td>
</tr>
</tbody>
</table>

### Transitivity table for eps>0

<table>
<thead>
<tr>
<th>$r_2(I^B, I^C)$</th>
<th>b</th>
<th>c</th>
<th>o</th>
<th>m</th>
<th>s</th>
<th>f</th>
<th>e</th>
<th>l</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r_1(I^A, I^B)$</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>b &quot;before&quot;</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>c &quot;contains&quot;</td>
<td>b</td>
<td>c</td>
<td>e</td>
<td>f</td>
<td>o</td>
<td>c</td>
<td>f</td>
<td>c</td>
</tr>
<tr>
<td>o &quot;overlaps&quot;</td>
<td>b</td>
<td>b</td>
<td>c</td>
<td>f</td>
<td>m</td>
<td>b</td>
<td>o</td>
<td>o</td>
</tr>
<tr>
<td>m &quot;meets&quot;</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>s &quot;starts&quot;</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>f &quot;finished-by&quot;</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>e &quot;equal&quot;</td>
<td>b</td>
<td>c</td>
<td>c</td>
<td>f</td>
<td>m</td>
<td>c</td>
<td>f</td>
<td>f</td>
</tr>
<tr>
<td>l &quot;left contain&quot;</td>
<td>b</td>
<td>c</td>
<td>c</td>
<td>f</td>
<td>o</td>
<td>c</td>
<td>f</td>
<td>e</td>
</tr>
</tbody>
</table>
## vertTIRP: 3. Efficient TIRPs representation

The vertTIRP allows for the efficient management of TIRP discovery:

- No DB scans
- Fast support counting and access
- Fast comparison between the TIRPs (using eid instead of time)

### Table A

<table>
<thead>
<tr>
<th>TIRP</th>
<th>sid</th>
<th>eid</th>
<th>starts</th>
<th>ends</th>
<th>Source intervals</th>
<th>vs</th>
<th>mhs</th>
<th>md</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>1</td>
<td>1</td>
<td>08:00</td>
<td>10:00</td>
<td>[08:00, 10:00]</td>
<td>1.0</td>
<td>0.31</td>
<td>1.3</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>16:00</td>
<td>17:00</td>
<td>[16:00, 17:00]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>12:00</td>
<td>13:00</td>
<td>[12:00, 13:00]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table B

<table>
<thead>
<tr>
<th>TIRP</th>
<th>sid</th>
<th>eid</th>
<th>starts</th>
<th>ends</th>
<th>Source intervals</th>
<th>vs</th>
<th>mhs</th>
<th>md</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>1</td>
<td>2</td>
<td>08:00</td>
<td>12:00</td>
<td>[08:00, 12:00]</td>
<td>1.0</td>
<td>0.31</td>
<td>3.0</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>2</td>
<td>16:00</td>
<td>18:00</td>
<td>[16:00, 18:00]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>2</td>
<td>11:00</td>
<td>14:00</td>
<td>[11:00, 14:00]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table AB

<table>
<thead>
<tr>
<th>TIRP</th>
<th>sid</th>
<th>eid</th>
<th>starts</th>
<th>ends</th>
<th>Source intervals</th>
<th>vs</th>
<th>mhs</th>
<th>md</th>
</tr>
</thead>
<tbody>
<tr>
<td>s</td>
<td>1</td>
<td>2</td>
<td>08:00</td>
<td>12:00</td>
<td>[08:00, 10:00],[08:00,12:00]</td>
<td>0.66</td>
<td>0.33</td>
<td>3.0</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>2</td>
<td>16:00</td>
<td>18:00</td>
<td>[16:00, 17:00],[16:00, 18:00]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
VertTIRP: 4. Constraints

**min_duration = 3 hours**
**min_gap = 1 hour**

Implemented in the vertTIRP algorithm

**Before (b):**
(B.s - A.e) & (B.s - A.e) < C_max_gap & (B.s - A.e) > C_min_gap

Filtered out

- **duration = 2 hours**
- **gap = 15 minutes**

2 hours \(\not\gtrsim\) min_duration(3 hours)
15 minutes \(\not\gtrsim\) min_gap (1 hour)

- **duration = 3 hours 30 minutes**
- **gap = 1 hour 15 minutes**

3 hours 30 minutes \(\gtrsim\) min_duration(3 hours)
1 hour 15 minutes \(\gtrsim\) min_gap (1 hour)
vertTIRP: 5. Pairing strategies.

Goal: ordering the testing of the conditions in temporal relations.

HAR dataset: Sorted relations:

bmfseclo

Sorted and grouped by common condition:

b mofc sel
vertTIRP: Experimental setup

- Real and synthetic datasets
- In this thesis, only the results relative to the following real datasets will be displayed.

<table>
<thead>
<tr>
<th>dataset</th>
<th>attribute sequence (NS)</th>
<th>attributes</th>
<th>num. TI</th>
<th>SAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAV_ρ</td>
<td>integer numbers (1000)</td>
<td>integer numbers</td>
<td>61,127</td>
<td>61,127</td>
</tr>
<tr>
<td>ASL_ρ</td>
<td>integer numbers (65)</td>
<td>integer numbers</td>
<td>2,037</td>
<td>31,338</td>
</tr>
</tbody>
</table>
vertTIRP: Experimental setup

- Efficiency of vertTIRP
  - Vertical representation of TIRPs
  - Efficiency by transitivity
  - User controlled patterns
  - Pairing strategies
  - Scalability: Complexity analysis
- Ablation studies on
  - Transitivity
  - User controlled patterns
  - Pairing strategies
  - Scalability
**vertTIRP:** Experimental setup

- **Efficiency of vertTIRP**
  - Vertical representation of TIRPs
  - Efficiency by transitivity
  - User controlled patterns
  - Pairing strategies
  - Scalability: Complexity analysis

- **Ablation studies on**
  - Transitivity
  - User controlled patterns
  - Pairing strategies
  - Scalability
vertTIRP: Results

**ASL_p dataset**
- Algorithm
- vertTIRP
- DharmaLego
- KarmaLego

**MAV_p dataset**
- Algorithm
- vertTIRP
- DharmaLego
vertTIRP: Discussion

• Vertical representation of TIRPs results in a great costs reduction

• Transitivity give a time reduction of up to 10%.

• The transitivity is particularly useful when the more frequent relations are the b, and m relations.

• Patterns discovered
  ○ the left contain relation is a discriminant relationship of the "SITTING" activity (<"BodyAcc_b" "GravityAcc_a", l>)
Introduction

Vepreco: sequential patterns

vertTIRP: temporal patterns

TA4L: preprocessing

Conclusions
**TA4L: Problem statement**

<table>
<thead>
<tr>
<th>sequence</th>
<th>var1</th>
<th>var2</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0,2</td>
<td>H</td>
<td>8:00</td>
</tr>
<tr>
<td>1</td>
<td>L</td>
<td>10:00</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0,2</td>
<td>M</td>
<td>12:00</td>
</tr>
<tr>
<td>1</td>
<td>0,1</td>
<td></td>
<td>13:00</td>
</tr>
<tr>
<td>1</td>
<td>0,2</td>
<td>M</td>
<td>14:00</td>
</tr>
<tr>
<td>1</td>
<td>0,8</td>
<td>M</td>
<td>15:00</td>
</tr>
<tr>
<td>1</td>
<td>0,7</td>
<td>M</td>
<td>16:00</td>
</tr>
</tbody>
</table>

Multivariate Time Series

\[
\text{LSTIS}^* = \langle H, 08:00, 10:00 \rangle, \langle A, 8:00, 15:00 \rangle, \langle L, 10:00, 12:00 \rangle, \langle M, 12:00, 16:00 \rangle, \langle B, 15:00, 16:00 \rangle
\]

*LSTIS: Lexicographical symbolic time interval sequence*
TA4L: State of the art

Patel et al. (2008); Wu & Chen (2007); Moskovitch & Shahar (2009, 2013); Huang et al. (2019).

Interval-based data (Chen et al. (2016))

Kam & Fu (2000); Winarko & Roddick (2007)
**TA4L: State of the art**

Pre-processing for TIRP mining

It depends on the datasets type

Approaches to pre-processing:
- **Internal** Yang et al. (2018); Huang et al. (2019)
- **External** Patel et al. (2008); Moskovitch & Shahar (2015), Chen et al. (2016)

Kam & Fu (2000); Winarko & Roddick (2007)

**Interval-based data** (Chen et al. (2016))
**Pre-processing for TIRP mining**

It depends on the datasets type

**Approaches to pre-processing:**
- **Internal** Yang et al. (2018); Huang et al. (2019)
- **External** Patel et al. (2008); Moskovitch & Shahar (2009, 2013); Huang et al. (2019)

**TA4L contributions**

Kam & Fu (2000); Winarko & Roddick (2007)
TA4L: Contributions

1. Formulation
2. Intervals based on time duration
3. A maximum gap constraint
4. Efficient data structure
5. Parallelism
**TA4L: 1. Formulation of the unsupervised pre-processing**

<table>
<thead>
<tr>
<th>sequence</th>
<th>var1</th>
<th>var2</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.2</td>
<td>H</td>
<td>8:00</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>L</td>
<td>10:00</td>
</tr>
<tr>
<td>1</td>
<td>0.2</td>
<td>M</td>
<td>12:00</td>
</tr>
<tr>
<td>1</td>
<td>0.1</td>
<td></td>
<td>13:00</td>
</tr>
<tr>
<td>1</td>
<td>0.2</td>
<td>M</td>
<td>14:00</td>
</tr>
<tr>
<td>1</td>
<td>0.8</td>
<td>M</td>
<td>15:00</td>
</tr>
<tr>
<td>1</td>
<td>0.7</td>
<td>M</td>
<td>16:00</td>
</tr>
</tbody>
</table>

Multivariate Time Series

$$\text{LSTIS} = \langle \text{H, 08:00,10:00}, \langle \text{A,8:00,15:00}, \langle \text{L,10:00,12:00}, \langle \text{M,12:00,16:00}, \langle \text{B,15:00,16:00} \rangle \rangle \rangle$$

**Diagram:**

- **For each sequence and variable:**
  - **if numeric:**
    - **Z-norm**
  - **if discrete:**
    - **Obtain mean**
    - **Obtain symbol**
- **Insert symbol into sequence and interval sortingly, with maximum duration**
- **Efficient data structure**
LSTIS(sid_i) =

<2016-10-04 07:59:42, 2016-10-04 19:00:22, var2.A>,
<2016-10-04 07:59:42, 2016-10-07 16:30:29, var1.B>,
<2016-10-08 07:57:38, 2016-10-08 16:30:42, var1.C>,
<2016-10-08 11:05:01, 2016-10-09 22:15:10, var2.C>,
<2016-10-09 09:04:46, 2016-10-09 16:30:43, var1.A>
TA4L: 2. Intervals based on time duration
TA4L: 2. Intervals based on time duration
TA4L: 3. A maximum gap constraint
**TA4L**: 3. A maximum gap constraint
TA4L: 4. Efficient data structure for LSTISs

Insert with an original binary search adapted to LSTISs
TA4L: 4. Efficient data structure for LSTISs

Insert with an improved binary search adapted to LSTISs
**TA4L: 5. Parallelism**

Parallelism applied to **sequences**
Data structure: **Sorted List**

- **Input**
  - Thread 1 → Sequence 1
  - Thread q → Sequence q

  - For each variable
    - Temporal abstraction
    - Sequence Generation
      - Interval sortingly inserted

- **Output**
  - Sort

Parallelism applied to **sequences**
Data structure: **Simple List**

- **Input**
  - Thread 1 → Sequence 1
  - Thread q → Sequence q

  - For each variable
    - Temporal abstraction
    - Sequence Generation
      - Interval inserted

- **Output**
  - Sort

Parallelism applied to **variables**
Data structure: **Simple List**

- **Input**
  - Thread 1 → Variable 1
  - Thread n → Variable n

  - For each sequence
    - Temporal abstraction
    - Sequence Generation
      - Interval inserted

- **Output**
  - Sort
TA4L: Experimental setup

- Synthetic and real datasets

- Performance in the function of the algorithm parameters:
  - duration $\delta$
  - max_gap
  - the size of the vocabulary $|\Sigma|$  

- Performance-based on the discretisation method:
  - SAX
  - EWD
  - KBTA

- Performance in the function of the dataset characteristics
  - PM
  - q
  - n
  - n_sid
TA4L: Experimental setup

- Synthetic and real datasets

- Performance in the function of the algorithm parameters:
  - duration $\delta$
  - max_gap
  - the size of the vocabulary $|\Sigma|$

- Performance-based on the discretisation method:
  - SAX
  - EWD
  - KBTA

- Performance in the function of the dataset characteristics
  - PM
  - q
  - n
  - n_sid
**TA4L: Results**

Synthetic datasets. How varying input parameters $\delta$ affects the performance of the algorithms.
TA4L: Results

Synthetic datasets. How varying input parameters $\delta$ affects the performance of the algorithms.
TA4L: Discussion

- As smaller is the $\delta$, more intervals are generated.
- EWD is the more memory efficient method
- Decrease in time, memory and number of TI with an increase in PM values
- The cost is greater for numeric variables and a sorted-list like structure.
- The best option is to parallelize the TA4L algorithm per sequence
Conclusions

➢ Improve pattern mining algorithms:
   • Sequential
   • Temporal

➢ Formalization of preprocessing

VEPRECO

vertTIRP

Efficient data structures
Pruning strategies
Robustness of the discovered patterns
Customisation of discovered patterns
Parallelism
Conclusions: Future work

- To continue the line of efficiency research (parallelism, incremental)
- New metrics to filter out redundant patterns
- Defining a new type of patterns to look for that might be useful
- To generate candidates with machine learning models

**VEPRECO:**
- To create specific SPM subfields, extending VEPRECO

**VertTIRP:**
- To adapt vertTIRP to mine the Top-K high-utility, closed or negative patterns
- Use vertTIRP to predict the side effects of the COVID-19 vaccines

**TA4L:**
- Improve the efficiency and quality of concatenation with clustering methods.
- Allow the same variable and the same timestamp record many different states.
- Classifying with TIRPs discovered from the intervals obtained with TA4L
Acknowledgments

This work was developed with the support of the:

- research group eXiT
- the competitive grant IFUdG2017
- mobility grant MOB2019

Projects:
- MoSHCA
- PEPPER
- ERDF/MINECO (SERAS)
- SITES
- grant MPCUdG2016
- NVIDIA Corporation donated the Titan XP
Thank you very much for your attention.

Questions