Declarative Sequential Pattern Mining

Prof. Dr. Rainer Gemulla

Data and Web Science Group
Universität Mannheim

BBDC Symposium
November 8, 2016
Data and Web Science Group

- Research group at University of Mannheim, Germany
  - 5 professors, 9 postdocs, 18 Ph.D. students

- European Network of National Big Data Centers of Excellence

- Research focus: Understand and leverage heterogeneous data in order to improve applications using knowledge

- Contribution to community via open data and open software
Outline

1. Sequential Pattern Mining
2. Scalability
3. Usability
4. Summary
Outline

1. Sequential Pattern Mining

2. Scalability

3. Usability

4. Summary
Anni wants to watch a movie. Anni loves LOTR1. But she does not want to see it. She had seen LOTR2 last week!
Let’s look at some data

- Data from Netflix’ online movie-streaming platform
  - 500k users, 18k movies, 100M ratings with timestamps
- 125k users rated both LOTR1 and LOTR2
- In which order?

105k users → 20k users

- Order matters!
  - How to discover patterns in sequential data?
Sequential pattern mining is a fundamental task in data mining. Data modeled as collection of sequences of items or events. Often items are arranged in a hierarchy. We seek useful sequential patterns.

E.g., market-basket data:
- Sequence = purchases of a customer over time
- Item = product (or set of products) + product hierarchy
- Example pattern: DSLR Camera → Tripod → Flash

E.g., natural-language text:
- Sequence = sentence or document
- Item = word + syntactic/semantic hierarchy
- Example pattern: person was born in location

E.g., amino acid sequences:
- Sequence = protein
- Item = amino acid
- Example pattern: S L R
What constitutes a good pattern?

- Extensively studied
  - Interesting patterns should be new, surprising, understandable, actionable
  - No random patterns, common knowledge, redundancy
  - Details application-specific

- Many different variants, many algorithms
  - Constraints: length, positional/temporal, hierarchy, regex, ...
  - Scoring: frequency, utility, information gain, significance, ...
  - Pattern sets: all, top-k, maximality, closedness, MDL, ...

- Our research focuses on unifying sequential pattern mining
  - Study general properties instead of special cases
  - Avoid need for customized mining algorithms
DESQ

- DESQ = system for declarative sequential pattern mining [ICDM16]
  - (Will be) open source
  - Work in progress

Key design goals are

1. **Usefulness**
   - Can be tailored to application
   - Flexible constraints
   - Flexible notions of interestingness

2. **Usability**
   - Describe pattern mining task in an intuitive, declarative way
   - Hide technical and implementation details

3. **Efficiency**
   - Fast
   - Scalable [SIGMOD15, TODS15, SIGMOD13]
   - Competitive to specialized miners
Outline

1. Sequential Pattern Mining
2. Scalability
3. Usability
4. Summary
Special case: $n$-gram mining

An $n$-gram is a sequence of $n$ consecutive words

- Extensively used in text mining and natural-language processing
- Web-scale $n$-gram models published by Google and Microsoft

![Google books Ngram Viewer](image)
Special case: \textit{n-gram mining}

An \textit{n-gram} is a sequence of \textit{n} consecutive words

- Extensively used in text mining and natural-language processing
- Web-scale \textit{n-gram} models published by Google and Microsoft

![Graph showing the frequency of phrases like "had a good day" and "had a bad day" from 1800 to 2000](image)
MG-FSM

- Distributed framework for scalable frequent sequence mining
- Originally built on top of MapReduce

Key idea
- Partition data into smaller overlapping partitions using **item-based partitioning**
  - One partition for each frequent item
  - Inexpensive rewrites
- Mine each partition using any FSM algorithm
- Combine results
How fast is it? (10 node Hadoop cluster)

5-grams from New York Times Data (50M sentences)
- Naive: 21 min
- Suffix-\(\sigma\) \(n\)-gram miner: 217 s
- MG-FSM: 103 s

Gapped 5-grams from New York Times Data (50M sentences)
- Naive: 3.7 h
- Suffix-\(\sigma\) \(n\)-gram miner: N/A
- MG-FSM: 137 s

5-grams from ClueWeb data (1B sentences)
- MG-FSM: 20 min
MG-FSM also mines... 

- Maximal and closed sequences
  - Compact, smaller output (e.g., factor 3 on NYT corpus)
  - No or minimal information loss

- Event sequences
  - Input sequences of time-annotated events
  - E.g., movie views, purchase transactions, session logs
  - Supports temporal constraints

- Hierarchies
  - Canon EOS 70D → DSLR camera → camera → electronics
  - E.g.: some DSLR camera, some photography book, some flash
<table>
<thead>
<tr>
<th>Sequence</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>want to do something</td>
<td>2152</td>
</tr>
<tr>
<td>have to do something</td>
<td>2103</td>
</tr>
<tr>
<td>authorize to seek contribution</td>
<td>1103</td>
</tr>
<tr>
<td>want to be part</td>
<td>1082</td>
</tr>
<tr>
<td>be to take place</td>
<td>1027</td>
</tr>
<tr>
<td>decline to comment yesterday</td>
<td>1011</td>
</tr>
<tr>
<td>try to do something</td>
<td>932</td>
</tr>
<tr>
<td>want to go home</td>
<td>675</td>
</tr>
<tr>
<td>try to take advantage</td>
<td>634</td>
</tr>
<tr>
<td>want to do anything</td>
<td>632</td>
</tr>
<tr>
<td>have to take care</td>
<td>623</td>
</tr>
<tr>
<td>refuse to answer question</td>
<td>618</td>
</tr>
<tr>
<td>expect to announce today</td>
<td>597</td>
</tr>
<tr>
<td>go to do something</td>
<td>594</td>
</tr>
<tr>
<td>adjust to represent sale</td>
<td>590</td>
</tr>
<tr>
<td>weight to represent sale</td>
<td>563</td>
</tr>
<tr>
<td>go to do anything</td>
<td>552</td>
</tr>
</tbody>
</table>
Outline

1. Sequential Pattern Mining
2. Scalability
3. Usability
4. Summary
Going declarative

- If we simply mined all frequent $n$-grams, we may
  1. Produce many uninteresting patterns (low frequency threshold)
  2. Miss out on interesting patterns (high frequency threshold)

- DESQ allows data analysts to focus on what they consider relevant
  - Supports all traditional constraints (length, gap, hierarchy, ...)
  - Supports customized constraints that go beyond traditional constraints

- Based on a declarative **pattern expression language**
  - Describe relevant patterns, let DESQ take care of mining them
  - Syntax like regular expression
  - Adds capture groups and hierarchies
Some examples for text mining

1. **Noun modified by adjective or noun**
   Ex: big country (110), green tea (337), research scientist (473)
   PE: ([ADJ|NOUN] NOUN)

2. **Relational phrase between entities**
   Ex: lives in (847), is being advised by (15), has coached (10)
   PE: ENTITY (VERB+ NOUN+? PREP?) ENTITY

3. **Typed relational phrases**
   Ex: ORG headed by ENTITY (275), PERS born in LOC (481)
   PE: (ENTITY↑ VERB+ NOUN+? PREP? ENTITY↑)

4. **Google n-gram viewer data**
   Ex: a good day, a ADJ day, DET ADJ NOUN, have a good day
   PE: (↑) (↑)? (↑)? | (.....?)
Pattern mining

- Under the hood, DESQ translates pattern expressions to finite state transducers (FST)
  - FST outputs all patterns that occur in a given input sequence

- Naive approach ("WordCount")
  - For every input sequence, simulate FST to obtain all outputs
  - Count how often each output occurred, return the frequent ones
  - Simple, inefficient

- DesqCount ("WordCount" with frequency pruning)
  - Lemma: frequent patterns cannot contain infrequent items
  - As naive, but ignore FST transitions that produce infrequent items
  - Simple, more efficient but still inefficient

- DesqDfs (depth-first search)
  - Lemma: partial outputs more frequent than resp. final outputs
  - Apply a variant of prefix-growth to grow patterns incrementally and prune early
  - Not that simple, efficient
Performance comparison (traditional constraints)

Left: cSPADE, center: prefix-growth, right: DesqDfs

DESQ is competitive to state-of-the-art miners for traditional constraints.
Performance comparison (new constraints)

DesqDfs is method of choice and can be orders of magnitude faster than Naive or DesqCount.
Outline

1. Sequential Pattern Mining

2. Scalability

3. Usability

4. Summary
Summary

- DESQ system for **declarative sequential pattern mining**
  - Find patterns in sequential data
  - First step towards a unifying framework
  - Pattern expressions to express constraints in an intuitive way
  - Item-based partitioning to scale to large datasets

- Directions for future work
  - Better algorithms & analysis
  - More powerful pattern expression language
  - Interestingness beyond frequency
  - Trees & graphs

Make sequential pattern mining useful, usable, and efficient.

Thank you!