LASH: Large-Scale Sequence Mining with Hierarchies

Kaustubh Beedkar and Rainer Gemulla

Data and Web Science Group
University of Mannheim

June 2nd, 2015
SIGMOD 2015
### Syntactic Explorer (Verb to Verb Noun)

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Count</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>
Sequence Mining

- Goal: Discover subsequences as patterns in sequence data
- Input: Collection of sequences of items, e.g.,
  - Text collection (sequence of words)
  - Customer transactions (sequence of products)
Sequence Mining

• Goal: Discover subsequences as patterns in sequence data

• Input: Collection of sequences of items, e.g.,
  ▶ Text collection (sequence of words)
  ▶ Customer transactions (sequence of products)

• Output: subsequences that
  ▶ occur in $\sigma$ input sequences (frequency threshold)
  ▶ have length at most $\lambda$ (length threshold)
  ▶ have gap $\gamma$ (contiguous subsequences or non-contiguous subsequences)
Sequence Mining

- **Goal:** Discover subsequences as patterns in sequence data

- **Input:** Collection of sequences of items, e.g.,
  - Text collection (sequence of words)
  - Customer transactions (sequence of products)

- **Output:** subsequences that
  - occur in $\sigma$ input sequences (frequency threshold)
  - have length at most $\lambda$ (length threshold)
  - have gap $\gamma$ (contiguous subsequences or non-contiguous subsequences)

- **Example:**
  - $S_1$: Anna lives in Melbourne
  - $S_2$: Bob lives in the city of Berlin
  - $S_3$: Charlie likes London
Sequence Mining

• Goal: Discover subsequences as patterns in sequence data

• Input: Collection of sequences of items, e.g.,
  ▶ Text collection (sequence of words)
  ▶ Customer transactions (sequence of products)

• Output: subsequences that
  ▶ occur in $\sigma$ input sequences (frequency threshold)
  ▶ have length at most $\lambda$ (length threshold)
  ▶ have gap $\gamma$ (contiguous subsequences or non-contiguous subsequences)

• Example:
  $S_1$: Anna lives in Melbourne
  $S_2$: Bob lives in the city of Berlin
  $S_3$: Charlie likes London
  ▶ Subsequence: lives in
  $\sigma = 2$, $\lambda = 2$, $\gamma = 0$
Hierarchies

Items can be naturally arranged in a hierarchy, e.g.,
Hierarchies

Items can be naturally arranged in a hierarchy, e.g.,

Syntactic hierarchy
Hierarchies

Items can be naturally arranged in a hierarchy, e.g.,

Syntactic hierarchy

Semantic hierarchy
Hierarchies

Items can be naturally arranged in a hierarchy, e.g.,

**Syntactic hierarchy**

- DET
  - a
  - an
  - the

**Semantic hierarchy**

- PERSON
  - Scientist
  - Politician
  - Barack Obama
  - Melbourne

- Photography

**Product hierarchy**

- DSLR Camera
  - Cannon5D
  - Nikon5100

- Tripod
Sequence Mining with Hierarchies

- Item hierarchies are specifically taken into account
- Discover **non-trivial** patterns
Sequence Mining with Hierarchies

• Item hierarchies are specifically taken into account
• Discover \textbf{non-trivial} patterns
• Example
  \(S_1\): Anna lives in Melbourne
  \(S_2\): Bob lives in the city of Berlin
  \(S_3\): Charlie likes London
Sequence Mining with Hierarchies

- Item hierarchies are specifically taken into account
- Discover non-trivial patterns

Example
- $S_1$: Anna lives in Melbourne
- $S_2$: Bob lives in the city of Berlin
- $S_3$: Charlie likes London
Sequence Mining with Hierarchies

- Item hierarchies are specifically taken into account
- Discover non-trivial patterns

Example

\[ S_1: \text{Anna lives in Melbourne} \]
\[ S_2: \text{Bob lives in the city of Berlin} \]
\[ S_3: \text{Charlie likes London} \]

- Generalized subsequence:
  \[ \text{PERSON lives in CITY} \]
  \[ \sigma = 2, \lambda = 4, \gamma = 3 \]
Sequence Mining with Hierarchies

Applications

• Linguistic patterns, e.g.,
  ▶ read DET book
  ▶ NNP lives in NNP

• Information extraction, e.g.,
  ▶ PERSON lives in CITY

• Market-basket analysis, e.g,
  ▶ buy DSLR camera → photography book → flash

• Web-usage mining

• ...

LASH

- Distributed framework for sequence mining with hierarchies
- Built over MapReduce for large-scale data processing
- Map (Partitioning)
  - Divide data into potentially overlapping partitions
- Reduce (mining)
  - Partitions are mined independently
- No global post-processing

\[ \text{Hierarchy-aware item-based partitioning} \]

\[ D \rightarrow H \]

\[ D_1 \rightarrow H_1 \]
\[ D_2 \rightarrow H_2 \]
\[ \ldots \]
\[ D_n \rightarrow H_n \]

\[ F_1 \rightarrow \text{Local mining} \]
\[ F_2 \rightarrow \text{Local mining} \]
\[ \ldots \]
\[ F_n \rightarrow \text{Local mining} \]

\[ F \]
Outline

1. Introduction
2. Partitioning
3. Local Mining
4. Evaluation
5. Conclusion
Item-based Partitioning

Items are ordered by decreasing frequency, e.g., $a < b < c < \cdots < k$.

Create a partition for each frequent item called pivot item.

Key idea: partition the output space.

- $a \triangleleft H_1$
- $b \triangleleft H_2$
- $c \triangleleft \cdots$
- $k \triangleleft H_n$

Hierarchy-aware item-based partitioning

$D$ for each pivot item

- Reduces communication
- Reduces computation
- Reduces skew

Hierarchy-aware item-based partitioning

$D_1 \rightarrow H_1 \rightarrow D_2 \rightarrow H_2 \rightarrow \cdots \rightarrow D_n \rightarrow H_n$

Local mining

$F_1 \rightarrow F_2 \rightarrow \cdots \rightarrow F_n$

$F_a$: Filter a but not b, ..., k

$F_b$: Filter b but not c, ..., k

$F_k$: Filter k

$F$
Item-based Partitioning

- Items are ordered by decreasing frequency, e.g., \( a < b < c < \cdots < k \)

![Diagram of item-based partitioning]

- Items are ordered by decreasing frequency, e.g., \( a < b < c < \cdots < k \)
- Create a partition for each frequent item called *pivot item*
- Key idea: partition the output space
- Rewrite \( D \) for each pivot item
  - Reduces communication
  - Reduces computation
  - Reduces skew
Item-based Partitioning

- Items are ordered by decreasing frequency, e.g., \( a < b < c < \cdots < k \)
- Create a partition for each frequent item called **pivot item**

Hierarchy-aware item-based partitioning

\[ D \overset{H}{\longrightarrow} F_a: \text{Filter } a \text{ but not } b, \ldots, k \]
\[ D_1 \overset{H_1}{\longrightarrow} F_1 \]
\[ D_2 \overset{H_2}{\longrightarrow} F_2 \]
\[ \cdots \]
\[ D_n \overset{H_n}{\longrightarrow} F_n \]
\[ F \]

\[ F_b: \text{Filter } b \text{ but not } c, \ldots, k \]

\[ F_k: \text{Filter } k \]
Item-based Partitioning

- Items are ordered by decreasing frequency, e.g., \( a < b < c < \cdots < k \)
- Create a partition for each frequent item called **pivot item**
- Key idea: partition the output space
  - \( a < b < c < \cdots < k \)
  - Reduces communication
  - Reduces computation
  - Reduces skew

Hierarchy-aware item-based partitioning

Local mining

\( F_a: \) Filter a but not b,...,k
\( F_b: \) Filter b but not c,...,k
\( F_k: \) Filter k

Kaustubh Beedkar and Rainer Gemulla
Item-based Partitioning

- Items are ordered by decreasing frequency, e.g., \( a < b < c < \cdots < k \)
- Create a partition for each frequent item called **pivot item**
- Key idea: partition the output space
  - \( a < b < c < \cdots < k \)
  - \( F_a \)
  - \( F_b \)
  - \( F_c \)
  - \( F_k \)
- Rewrite \( D \) for each pivot item
  - Reduces communication
  - Reduces computation
  - Reduces skew
Item-based Partitioning

Example ($\sigma = 2, \gamma = 3, \lambda = 4$)

$S_1$: Anna lives in Melbourne
$S_2$: Bob lives in the city of Berlin
$S_3$: Charlie likes London

Semantic hierarchy
Item-based Partitioning

Example \((\sigma = 2, \gamma = 3, \lambda = 4)\)

\(S_1:\) Anna lives in Melbourne
\(S_2:\) Bob lives in the city of Berlin
\(S_3:\) Charlie likes London

Semantic hierarchy

- **PERSON** < **CITY** < in < lives
Item-based Partitioning

Example ($\sigma = 2, \gamma = 3, \lambda = 4$)

$S_1$: Anna lives in Melbourne
$S_2$: Bob lives in the city of Berlin
$S_3$: Charlie likes London

Semantic hierarchy

- PERSON < CITY < in < lives
Item-based Partitioning

Example ($\sigma = 2, \gamma = 3, \lambda = 4$)

$S_1$: Anna lives in Melbourne
$S_2$: Bob lives in the city of Berlin
$S_3$: Charlie likes London

Semantic hierarchy

- PERSON < CITY < in < lives
Item-based Partitioning

Example ($\sigma = 2, \gamma = 3, \lambda = 4$)

$S_1$: Anna lives in Melbourne

$S_2$: Bob lives in the city of Berlin

$S_3$: Charlie likes London

Semantic hierarchy

- PERSON < CITY < in < lives
Item-based Partitioning

Example ($\sigma = 2, \gamma = 3, \lambda = 4$)

$S_1$: Anna lives in Melbourne
$S_2$: Bob lives in the city of Berlin
$S_3$: Charlie likes London

Semantic hierarchy

- **PERSON** < **CITY** < **in** < **lives**
Outline

1. Introduction
2. Partitioning
3. Local Mining
4. Evaluation
5. Conclusion
Local Mining

- Goal: Compute **pivot sequences**
- \( a < b < c < \cdots < k \)

\( F_a \)
\( F_b \)
\( F_c \)
\( F_k \)

Hierarchy-aware item-based partitioning

\( D \)
\( H \)

\( D_1 \)
\( H_1 \)

\( D_2 \)
\( H_2 \)

\( \ldots \)

\( D_n \)
\( H_n \)

Local mining

\( F_1 \)

\( F_2 \)

\( \ldots \)

\( F_n \)

\( F \)

\( F_a: \) Filter a but not b,...,k

\( F_b: \) Filter b but not c,...,k

\( F_k: \) Filter k
Local Mining

- Traditional approach
  - Use any mining algorithm (based on depth-first or breadth-first search)
  - Filter out non-pivot sequences

- Example: depth-first search
  - Pivot item: e
Local Mining

- **Pivot sequence miner (PSM)**
  - Mines only pivot sequences
    - Start with the pivot item
    - **Right expansions**
    - **Left expansions**
  - Optimized search space exploration

- **Example: PSM search space**
  - Pivot item: e

![Diagram](image)
Outline

1 Introduction
2 Partitioning
3 Local Mining
4 Evaluation
5 Conclusion
Overall Runtime

The New York Times Corpus

• ~50M sequences, ~1B items of which ~2.7M distinct
• Syntactic hierarchy (word → lowercase → lemma → POS tag)
• 10 node hadoop cluster

LASH is multiple orders of magnitude faster
Local Mining

PSM is effective, more than $3 \times$ faster
Scalability

(a) Strong Scalability

(b) Weak Scalability

Good strong and weak scalability
Outline

1 Introduction
2 Partitioning
3 Local Mining
4 Evaluation
5 Conclusion
Summary and Contributions

• Sequence mining with hierarchies is an important problem
  ▶ Enables mining non-trivial patterns
Summary and Contributions

- Sequence mining with hierarchies is an important problem
  - Enables mining non-trivial patterns

- LASH: **LArge-scale Sequence mining with Hierarchies**
  - Novel hierarchy-aware form of item-based partitioning
  - Efficient special-purpose algorithm for mining each partition
Summary and Contributions

• Sequence mining with hierarchies is an important problem
  ▶ Enables mining non-trivial patterns

• LASH: LArge-scale Sequence mining with Hierarchies
  ▶ Novel hierarchy-aware form of item-based partitioning
  ▶ Efficient special-purpose algorithm for mining each partition

• First distributed, scalable algorithm to mine such sequences
Summary and Contributions

• Sequence mining with hierarchies is an important problem
  ▶ Enables mining non-trivial patterns

• LASH: LArge-scale Sequence mining with Hierarchies
  ▶ Novel hierarchy-aware form of item-based partitioning
  ▶ Efficient special-purpose algorithm for mining each partition

• First distributed, scalable algorithm to mine such sequences

Thank you!
Questions? / Comments