Sequential Patterns

Huiping Cao

Examples of Sequence

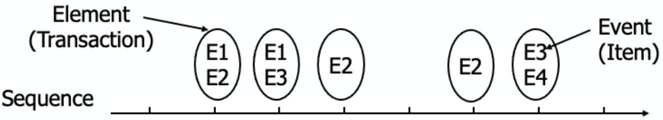
- Sequence of different transactions by a customer at an online store:
  - $\langle\{Digital\ Camera, iPad\}, \{memorycard\}, \{headphone, iPad\ cover\}\rangle$
- Sequence of initiating events causing the nuclear accident at 3-mile Island
  - `https://en.wikipedia.org/wiki/Three_Mile_Island_accident`
  - $\langle\{clogged\ resin\}\{outlet\ valve\ closure\}\{loss\ of\ feedwater\}\{condenser\ polisher\ outlet\ valve\ shut\}\{booster\ pumps\ trip\}\{main\ waterpump\ trips\}\{main\ turbine\ trips\}\{reactor\ pressure\ increases\}\rangle$
- Sequence of books checked out at a library:
  - $\langle\{Fellowship\ of\ the\ Ring\}, \{The\ Two\ Towers\}, \{Return\ of\ the\ King\}\rangle$

Sequential Pattern Discovery: Examples

- In telecommunications alarm logs,
  - Inverter_Problem:
    (Excessive_Line_Current) (Rectifier_Alarm) → (Fire_Alarm)
- In point-of-sale transaction sequences,
  - Computer Bookstore:
    (Intro_To_Visual_C) (C++_Primer) → (Perl_for_dummies)
- Athletic Apparel Store:
  - (Shoes) (Racket, Racketball) → (Sports_Jacket)

Sequence Data

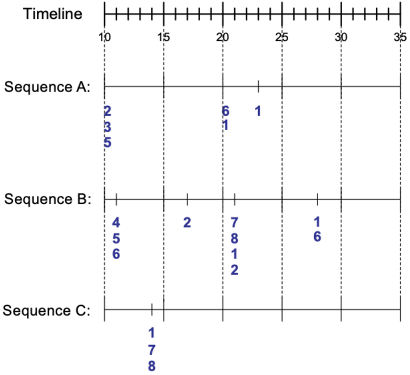
Sequence Database	Sequence	Element (Transaction)	Event (Item)
Customer	Purchase history of a given customer	A set of items bought by a customer at time t	Books, diary products, CDs, etc
Web Data	Browsing activity of a particular Web visitor	A collection of files viewed by a Web visitor after a single mouse click	Home page, index page, contact info, etc
Event data	History of events generated by a given sensor	Events triggered by a sensor at time t	Types of alarms generated by sensors
Genome sequences	DNA sequence of a particular species	An element of the DNA sequence	Bases A,T,G,C



Sequence Data

Sequence Database:

Sequence ID	Timestamp	Events
A	10	2, 3, 5
A	20	6, 1
A	23	1
B	11	4, 5, 6
B	17	2
B	21	7, 8, 1, 2
B	28	1, 6
C	14	1, 7, 8



Sequence Data vs. Market-basket Data

Sequence Database:

Customer	Date	Items bought
A	10	2, 3, 5
A	20	1,6
A	23	1
B	11	4, 5, 6
B	17	2
B	21	1,2,7,8
B	28	1, 6
C	14	1,7,8

Market- basket Data

Events
2, 3, 5
1,6
1
4,5,6
2
1,2,7,8
1,6
1,7,8

Formal Definition of a Sequence

- A sequence is an ordered list of elements

$$s = \langle e_1 \ e_2 \ e_3 \ \cdots \rangle$$

  - Each element contains a collection of events (items)

$$e_i = \{i_1, i_2, \cdots, i_k\}$$

- Length of a sequence, $|s|$, is given by the number of elements in the sequence

- A k-sequence is a sequence that contains k events (items)

Formal Definition of a Sequence

- A sequence $\langle a_1 \ a_2 \cdots a_n \rangle$ is contained in another sequence $\langle b_1 \ b_2 \ \cdots b_m \rangle$ $(m \geq n)$ if there exist integers $i_1 < i_2 < \cdots < i_n$ such that $a - 1 \subseteq b_{i_1}, a_2 \subseteq b_{i_2}, \cdots, a_n \subseteq b_{i_n}$

- Illustrative Example:

s:	b_1	b_2	b_3	b_4	b_5
t:		a_1	a_2		a_3

t is a subsequence of s if $a_1 \subseteq b_2, a_2 \subseteq b_3, a_3 \subseteq b_5$

Data sequence	Subsequence	Contain?
< {2,4} {3,5,6} {8} >	< {2} {8} >	Yes
< {1,2} {3,4} >	< {1} {2} >	No
< {2,4} {2,4} {2,5} >	< {2} {4} >	Yes
<{2,4} {2,5}, {4,5}>	< {2} {4} {5} >	No
<{2,4} {2,5}, {4,5}>	< {2} {5} {5} >	Yes
<{2,4} {2,5}, {4,5}>	< {2, 4, 5} >	No

Sequential Pattern Mining: Definition

- The support of a subsequence w is defined as the fraction of data sequences that contain w
- A sequential pattern is a frequent subsequence (i.e., a subsequence whose support is $\geq$ minsup)
- Given
  - a database of sequences
  - a user-specified minimum support threshold, *minsup*
- Task
  - Find all subsequences with support $\geq$ minsup

Sequential Pattern Mining: Example

Object	Timestamp	Events
A	1	1,2,4
A	2	2,3
A	3	5
B	1	1,2
B	2	2,3,4
C	1	1, 2
C	2	2,3,4
C	3	2,4,5
D	1	2
D	2	3, 4
D	3	4, 5
E	1	1, 3
E	2	2, 4, 5

Minsup = 50%

Examples of Frequent Subsequences:

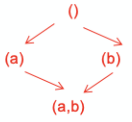
< {1,2} >	s=60%
< {2,3} >	s=60%
< {2,4}>	s=80%
< {3} {5}>	s=80%
< {1} {2} >	s=80%
< {2} {2} >	s=60%
< {1} {2,3} >	s=60%
< {2} {2,3} >	s=60%
< {1,2} {2,3} >	s=60%

Extracting Sequential Patterns

- Given n events: $i_1, i_2, i_3, \cdots, i_n$
- Candidate 1-subsequences:
  $\langle\{i_1\}\rangle, \langle\{i_2\}\rangle, \langle\{i_3\}\rangle, \cdots, \langle\{i_n\}\rangle$
- Candidate 2-subsequences:
  $\langle\{i_1, i_2\}\rangle, \langle\{i_1, i_3\}\rangle, \cdots,$
  $\langle\{i_1\}\{i_1\}\rangle, \langle\{i_1\}\{i_2\}\rangle, \cdots, \langle\{i_n\}\{i_n\}\rangle$
- Candidate 3-subsequences:
  $\langle\{i_1, i_2, i_3\}\rangle, \langle\{i_1, i_2, i_4\}\rangle, \cdots,$
  $\langle\{i_1, i_2\}\{i_1\}\rangle, \langle\{i_1, i_2\}\{i_2\}\rangle, \cdots,$
  $\langle\{i_1\}\{i_1, i_2\}\rangle, \langle\{i_1\}\{i_1, i_3\}\rangle, \cdots,$
  $\langle\{i_1\}\{i_1\}\{i_1\}\rangle, \langle\{i_1\}\{i_1\}\{i_2\}\rangle, \cdots$

Extracting Sequential Patterns: Simple example

- Given 2 events: a, b
- Candidate 1-subsequences:
  $\langle\{a\}\rangle, \langle\{b\}\rangle$
- Candidate 2-subsequences:
  $\langle\{a\}\{a\}\rangle, \langle\{a\}\{b\}\rangle, \langle\{b\}\{a\}\rangle, \langle\{b\}\{b\}\rangle, \langle\{a, b\}\rangle$.
- Candidate 3-subsequences:
  $\langle\{a\}\{a\}\{a\}\rangle, \langle\{a\}\{a\}\{b\}\rangle, \langle\{a\}\{b\}\{a\}\rangle, \langle\{a\}\{b\}\{b\}\rangle,$
  $\langle\{b\}\{b\}\{b\}\rangle, \langle\{b\}\{b\}\{a\}\rangle, \langle\{b\}\{a\}\{b\}\rangle, \langle\{b\}\{a\}\{a\}\rangle,$
  $\langle\{a, b\}\{a\}\rangle, \langle\{a, b\}\{b\}\rangle, \langle\{a\}\{a, b\}\rangle, \langle\{b\}\{a, b\}\rangle$



Item-set patterns

Generalized Sequential Pattern (GSP)

- **Step 1**: Make the first pass over the sequence database D to yield all the 1-element frequent sequences
- **Step 2**: Repeat until no new frequent sequences are found
  - **Candidate Generation**: Merge pairs of frequent subsequences found in the $(k\text{-}1)$th pass to generate candidate sequences that contain k items
  - **Candidate Pruning**: Prune candidate k-sequences that contain infrequent $(k\text{-}1)$-subsequences
  - **Support Counting**: Make a new pass over the sequence database D to find the support for these candidate sequences
  - **Candidate Elimination**: Eliminate candidate k-sequences whose actual support is less than *minsup*

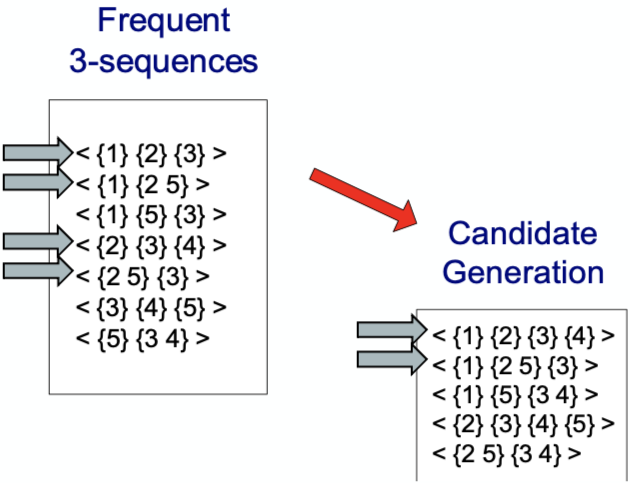
Candidate Generation

- Base case (k=2):
  - Merging two frequent 1-sequences $\langle\{i_1\}\rangle$ and $\langle\{i_2\}\rangle$ will produce the following candidate 2-sequences: $\langle\{i_1\}\{i_1\}\rangle, \langle\{i_1\}\{i_2\}\rangle, \langle\{i_2\}\{i_2\}\rangle, \langle\{i_2\}\{i_1\}\rangle, \langle\{i_1, i_2\}\rangle$
- General case ($k > 2$):
  - A frequent (k-1)-sequence $w1$ is merged with another frequent (k-1)-sequence $w2$ to produce a candidate k-sequence if the subsequence obtained by removing an event from the first element in $w1$ is the same as the subsequence obtained by removing an event from the last element in $w2$
  - The resulting candidate after merging is given by extending the sequence $w1$ as follows
    - If the last element of $w2$ has only one event, append it to $w1$
    - Otherwise, add the event from the last element of $w2$ (which is absent in the last element of $w1$) to the last element of $w1$

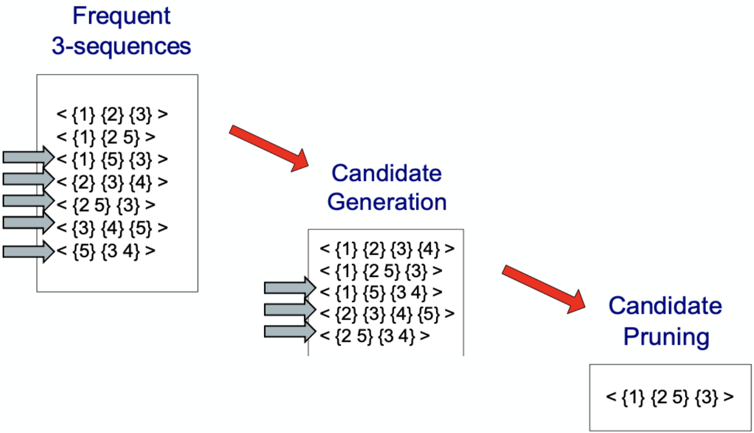
Candidate Generation Examples

- Merging $w1 = \langle\{1,2,3\}\{4,6\}\rangle$ and $w2 = \langle\{2,3\}\{4,6\}\{5\}\rangle$ produces the candidate sequence $\langle\{1,2,3\}\{4,6\}\{5\}\rangle$ because the last element of w2 has only one event
- Merging $w1 = \langle\{1\}\{2,3\}\{4\}\rangle$ and $w2 = \langle\{2,3\}\{4,5\}\rangle$ produces the candidate sequence $\langle\{1\}\{2,3\}\{4,5\}\rangle$ because the last element in $w2$ has more than one event
- Merging $w1 = \langle\{1,2,3\}\rangle$ and $w2 = \langle\{2,3,4\}\rangle$ produces the candidate sequence $\langle\{1,2,3,4\}\rangle$ because the last element in $w2$ has more than one event
- We do not have to merge the sequences $w1 = \langle\{1\}\{2,6\}\{4\}\rangle$ and $w2 = \langle\{1\}\{2\}\{4,5\}\rangle$ to produce the candidate $\langle\{1\}\{2,6\}\{4,5\}\rangle$ because if the latter is a viable candidate, then it can be obtained by merging $w1$ with $\langle\{2,6\}\{4,5\}\rangle$

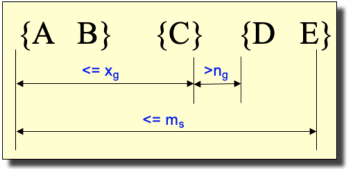
GSP example


Frequent
3-sequences
< {1} {2} {3} >
< {1} {2 5} >
< {1} {5} {3} >
< {2} {3} {4} >
< {2 5} {3} >
< {3} {4} {5} >
< {5} {3 4} >
Candidate
Generation
< {1} {2} {3} {4} >
< {1} {2 5} {3} >
< {1} {5} {3 4} >
< {2} {3} {4} {5} >
< {2 5} {3 4} >

GSP example (cont.)



Timing Constraints



x_g: max-gap

n_g: min-gap

m_s: maximum span

$x_g = 2$, $n_g = 0$, $m_s = 4$

Data sequence, d	Sequential Pattern, s	d contains s?
< {2,4} {3,5,6} {4,7} {4,5} {8} >	< {6} {5} >	Yes
< {1} {2} {3} {4} {5}>	< {1} {4} >	No
< {1} {2,3} {3,4} {4,5}>	< {2} {3} {5} >	Yes
< {1,2} {3} {2,3} {3,4} {2,4} {4,5}>	< {1,2} {5} >	No

Mining Sequential Patterns with Timing Constraints

- Approach 1: Mine sequential patterns without timing constraints
  - Postprocess the discovered patterns
- Approach 2: Modify GSP to directly prune candidates that violate timing constraints
  - Question: Does Apriori principle still hold?

Apriori Principle for Sequence Data

Object	Timestamp	Events
A	1	1,2,4
A	2	2,3
A	3	5
B	1	1,2
B	2	2,3,4
C	1	1, 2
C	2	2,3,4
C	3	2,4,5
D	1	2
D	2	3, 4
D	3	4, 5
E	1	1, 3
E	2	2, 4, 5

Suppose:

$x_g = 1$ (max-gap)

$n_g = 0$ (min-gap)

$m_s = 5$ (maximum span)

minsup = 60%

<{2} {5}> support = 40%

but

<{2} {3} {5}> support = 60%

Problem exists because of max-gap constraint

No such problem if max-gap is infinite

Object D does not support the pattern $\langle\{2\}\{5\}\rangle$ since the time gap between events 2 and 5 is greater than *maxgap*.

Contiguous Subsequences

- s is a contiguous subsequence of $w = \langle e_1, e_2, \cdots, e_k \rangle$ if any of the following conditions hold:
  - s is obtained from w by deleting an item from either e_1 or e_k
  - s is obtained from w by deleting an item from any element e_i that contains at least 2 items
  - s is a contiguous subsequence of s' and s' is a contiguous subsequence of w (recursive definition)
- Examples: $s = \langle\{1\}\{2\}\rangle$ is a contiguous subsequence of $\langle\{1\}\{2,3\}\rangle$, $\langle\{1,2\}\{2\}\{3\}\rangle$, and $\langle\{3,4\}\{1,2\}\{2,3\}\{4\}\rangle$ is not a contiguous subsequence of $\langle\{1\}\{3\}\{2\}\rangle$ and $\langle\{2\}\{1\}\{3\}\{2\}\rangle$

Modified Candidate Pruning Step

- Without *maxgap* constraint:
  - A candidate k-sequence is pruned if at least one of its (k-1)-subsequences is infrequent
- With *maxgap* constraint:
  - A candidate k-sequence is pruned if at least one of its contiguous (k-1)-subsequences is infrequent

Research articles (1)

- R. Srikant and R. Agrawal. Mining sequential patterns: Generalizations and performance improvements. EDBT96.
- H. Mannila, H Toivonen, and A. I. Verkamo. Discovery of frequent episodes in event sequences. DAMI:97.
- Roberto J. Bayardo Jr.: Efficiently Mining Long Patterns from Databases. SIGMOD Conference 1998: 85-93
- M. Zaki. SPADE: An Efficient Algorithm for Mining Frequent Sequences. Machine Learning, 2001.
- J. Pei, J. Han, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. ICDE'01 (TKDE04).
- J. Pei, J. Han and W. Wang, Constraint-Based Sequential Pattern Mining in Large Databases, CIKM'02.
- X. Yan, J. Han, and R. Afshar. CloSpan: Mining Closed Sequential Patterns in Large Datasets. SDM'03.

Research articles (2)

- J. Wang and J. Han, BIDE: Efficient Mining of Frequent Closed Sequences, ICDE'04.
- H. Cheng, X. Yan, and J. Han, IncSpan: Incremental Mining of Sequential Patterns in Large Database, KDD'04.
- J. Han, G. Dong and Y. Yin, Efficient Mining of Partial Periodic Patterns in Time Series Database, ICDE'99.
- J. Yang, W. Wang, and P. S. Yu, Mining asynchronous periodic patterns in time series data, KDD'00.

References

- Chapter 6: Introduction to Data Mining (2nd Edition) by Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, and Vipin Kumar
- Implementation of GSP algorithm: `https://github.com/jacksonpradolima/gsp-py`