Sequential Pattern Mining

Hamidreza Mahdavipanah
Pegah Hajian
Narges Heydarzadeh

Professor: Dr. S. M. Vahidipour

University of Kashan
Fall 2017

Outlines

- Introduction
- Definitions
- GSP
- Constraints that **GSP** supports

Introduction

Studies on **Sequential Pattern Mining**

- First introduced by **Agrawal** and **Srikant** in 1995
  - They presented three algorithms
    - *AprioriAll*
    - *AprioriSome*
    - *DynamicSome*
- Then in 1996 they presented ***GSP*** algorithm which was much faster than former algorithms and it also was generalized for more real life problems
- Pattern-growth methods: **FreeSpan** and **PrefixSpan**
- Mining closed sequential patterns: **CloSpan**
- ...

Applications

- Customer shopping sequences
  - First buy computer, then CD-ROM, and then digital camera, within 3 month.
- Medical treatments
- Natural disasters (e.g., earthquakes)
- Stocks and markets
- DNA sequences and gene structures

Problem Statement

We are given a database *D* of customer transactions. Each transaction consists of the following fields:

customer-id	transaction-time	items

We want to find all **large sequences** that have a certain user-specified **minimum support.**

It is similar to the frequent itemsets mining, but with consideration of ordering.

Example of a database

Customer Id	Transaction Time	Items Bought
1	June 25 ‘93	30
1	June 30 ‘93	90
2	June 10 ‘93	10, 20
2	June 15 ‘93	30
2	June 20 ‘93	40, 60, 70
3	June 25 ‘93	30, 50, 70
4	June 25 ‘93	30
4	June 30 ‘93	40, 70
4	June 25 ‘93	90
5	June 12 ‘93	90

We convert DB to this form

Customer Id	Customer Sequence
1	<(30)(90)>
2	<(10 20) (30) (40 60 70)>
3	<(30 50 70)>
4	<(30) (40 70) (90)>
5	<(90)>

Definitions

Itemset and Sequence

An ***itemset*** is a non-empty set of items.

- We denote an itemset *i* by (i_1 i_2 … i_m)

A ***Sequence*** is an **ordered list of items**.

- We denote a sequence *s* by <s_1 s_2 … s_n>

Subsequence and supersequence

Given two sequences $\boldsymbol{\alpha} = \langle a_1\ a_2 \dots a_n \rangle$ and $\boldsymbol{\beta} = \langle b_1\ b_2 \dots b_m \rangle$:

- α is called a subsequence of β, if there exists integers $1 \leq j_1 < j_2 < \dots < j_n \leq m$ such that $a_1 \subseteq b_1, a_2 \subseteq b_2, \dots, a_n \subseteq b_{jn}$
- β is called a supersequence of α

Example:

α=<(a b) d> and β=<(a b c) (d e)>

Apriori Property of Sequential Patterns

If a sequence *S* is not frequent, then none of the super-sequences of *S* is frequent.

Example:
<h b> is infrequent -> so do <h a b> and <(a h) b>

GSP

Outline of the GSP Method

- Initially, every item in DB is a candidate of length-1
- For each level (i.e., sequences of length-k) do
  - Scan database to collect support count for each candidate sequence
  - Generate candidate length(k + 1) sequences from length-k frequent sequences using Apriori
  - Repeat until no frequent sequence or no candidate can be found

Major strength of **GSP** is its candidate pruning by Apriori property

Finding Length-1 Sequential Patterns

- Initial candidates:
  - <a>, <b>, <c>, <d>, <e>, <f>, <g>, <h>
- Scan database once, count support for candidates

min_sup =2

Seq. ID	Sequence
10	<(bd)cb(ac)>
20	<(bf)(ce)b(fg)>
30	<(ah)(bf)abf>
40	<(be)(ce)d>
50	<a(bd)bcb(ade)>

Cand	Sup
<a>	**3**
<b>	**5**
<c>	**4**
<d>	**3**
<e>	**3**
<f>	**2**
~~<g>~~	1
~~<h>~~	1

Generating Length-2 Candidates

	<a>	<b>	<c>	<d>	<e>	<f>
<a>	<aa>	<ab>	<ac>	<ad>	<ae>	<af>
<b>	<ba>	<bb>	<bc>	<bd>	<be>	<bf>
<c>	<ca>	<cb>	<cc>	<cd>	<ce>	<cf>
<d>	<da>	<db>	<dc>	<dd>	<de>	<df>
<e>	<ea>	<eb>	<ec>	<ed>	<ee>	<ef>
<f>	<fa>	<fb>	<fc>	<fd>	<fe>	<ff>

= 36

Generating Length-2 Candidates

	<a>	<b>	<c>	<d>	<e>	<f>
<a>		<(ab)>	<(ac)>	<(ad)>	<(ae)>	<(af)>
<b>			<(bc)>	<(bd)>	<(be)>	<(bf)>
<c>				<(cd)>	<(ce)>	<(cf)>
<d>					<(de)>	<(df)>
<e>						<(ef)>
<f>						

= 15

Generating Length-2 Candidates

Using Apriori : 36 + 15 = **51** length-2 candidates

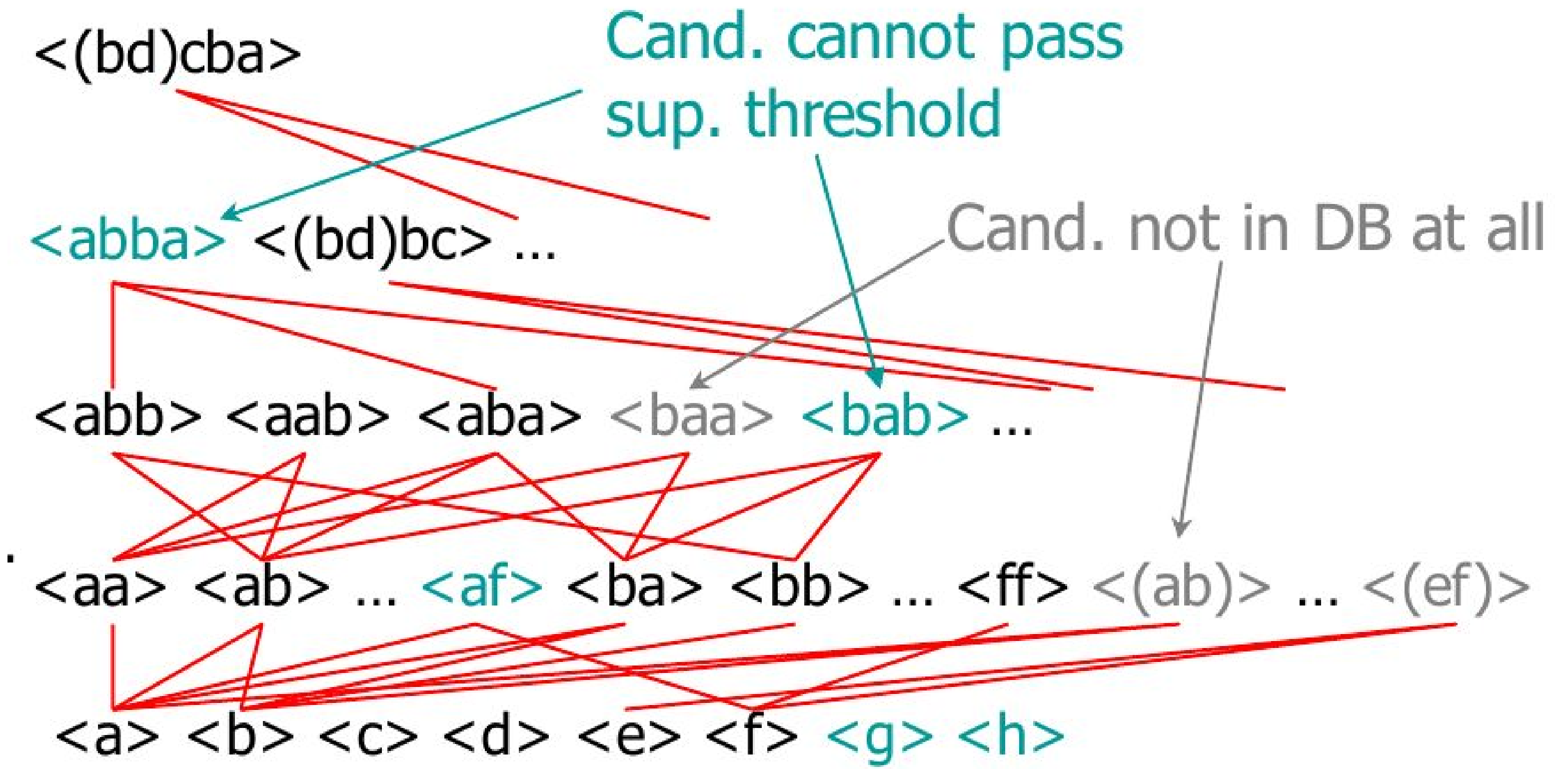
Without Apriori : (8 * 8) + (8 * 7) / 2 = **92** length- candidates

Apriori prunes 44.57% candidates

Finding Length-2 Sequential Patterns

- Scan database one more time, collect support count for each length-2 candidate
- There are 19 length-2 candidates which pass the minimum support threshold
  - **They are length-2 sequential patterns**

The GSP Mining Process



The GSP Algorithm

- Take sequences in form of <x> as length-1 candidates
- Scan database once, find F_1, the set of length-1 sequential patterns
- Let k = 1; while F_k is not empty do
    - Form C_{k+1} the set of length-(k + 1) candidates from F_k
    - If C_{k+1} is not empty, scan database once, find F_{k+1}, the set of length(k + 1) sequential patterns
    - Let k = k + 1

The Good, the Bad, and the Ugly

- *The Good*: benefits from the Apriori pruning which **reduces search space**
- *The Bad:* Scans the database multiple times
- *The Ugly:* Generates a **huge set of candidates** sequences

Why **GSP** is called **Generalized Sequential Pattern Mining**?

For practical use of **SPM**, in 1996 Agrawal and Srikant introduced three type of constraints that makes **SPM** problem more general and practical and since **GSP** support these constraints it is called **Generalized** sequential pattern mining.

Constraints that GSP Supports

Time Constraint

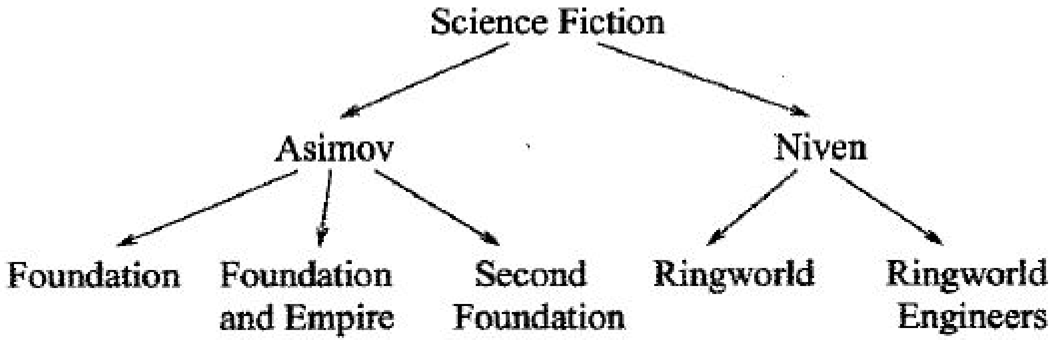
An ability for users to specify maximum and/or minimum time gaps between adjacent elements of the sequential pattern.

Sliding Window

That is, each element of the pattern can be contained in the union of the items bought in a set of transactions, as long as the difference between the maximum and minimum transaction-times is less than the size of a *sliding time window*.

Taxonomy

An ability to define a taxonomy (*is-a* hierarchy) over the items in the data.



References

- R. Agrawal and R. Srikant. Mining Sequential Patterns.1995
- R. Srikant and R. Agrawal. Mining Sequential Patterns.1996
- Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, Helen Pinto, PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. 2001