IMPROVING THE EFFICIENCY OF INTERACTIVE SEQUENTIAL PATTERN MINING BY CLOSED PATTERN DISCOVERY

ADMA2025　208

Yui Aoyagi[1)], Hieu Hanh Le[1)], Ryosuke Matsuo[1)], Tomoyoshi Yamazaki[1)], Kenji Araki[1)], Haruo Yokota [2)], and Masato Oguchi[1)]

1)Ochanomizu University, Japan

2)Josai University, Japan

Contents

Contents

Research Background(1/3)

- With the growth of big data, analysis of sequential data has gained increasing attention

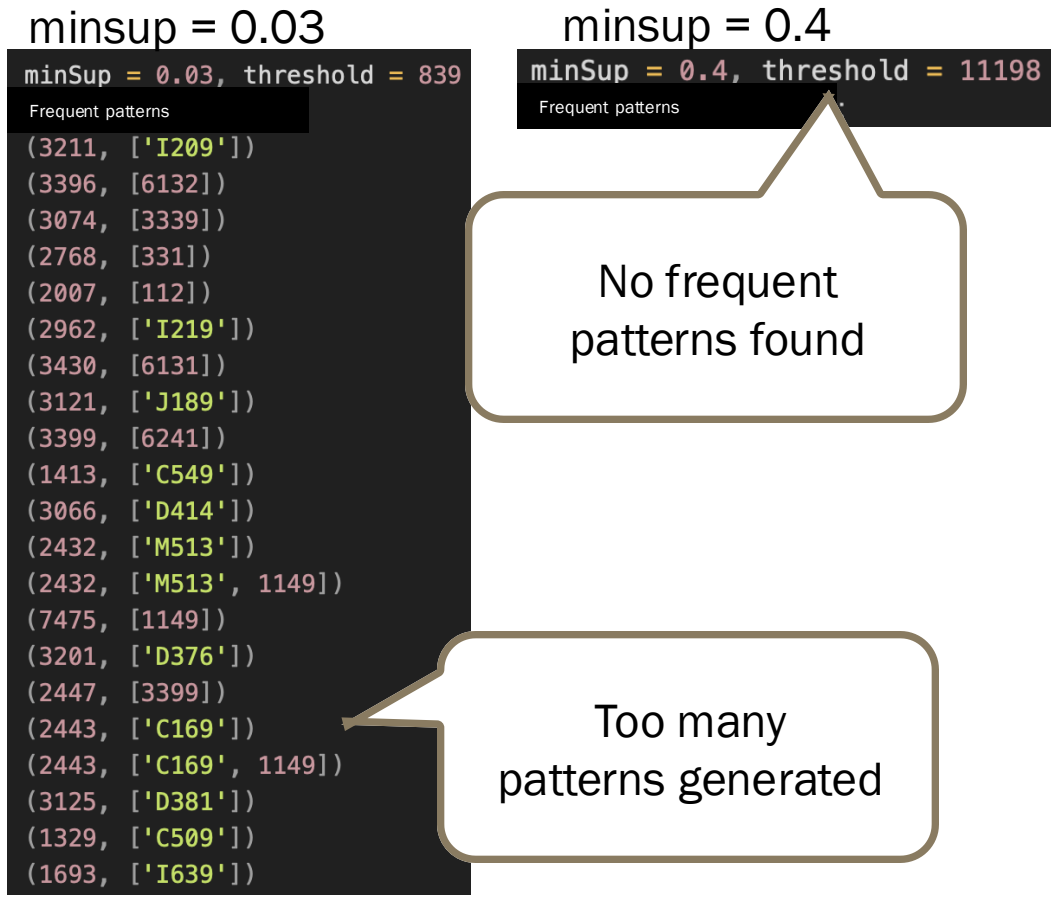
- Sequential Pattern Mining (SPM)
  - *Extracts frequent patterns whose support exceeds a given threshold*
  - *Well-known algorithms*：*SPADE*[1]、*SPAM*[2]、*PrefixSpan*[3]

- Useful for discovering insights from data where the order of items matters, such as:
  - *Purchase behavior analysis, Medical record analysis, and Web clickstream analysis*

[1]SPADE[Mohammed J. Zaki, Machine Learning]
[2]SPAM[R. Agrawal and R. Srikant.,ICCE1995]
[3]PrefixSpan[Pei et al.,ICDE2001]

Research Background(2/3)

- In SPM, the user specifies a minimum support (minsup) threshold
  - *minsup*：*The ratio of the number of sequences containing a pattern*
- The optimal **minsup** depends on the dataset
  - *Too small*：*too many patterns are generate*
  - *Too large*：*few or no patterns are found*
  - *To efficiently analyze while adjusting minsup, interactive SPM is essential*
  - *Major interactive SPM algorithms*：*GSP*[4]、*KISP*[5]



[4]GSP[R. Srikant and R. Agrawal. ,EDBT1996]
[5]KISP[M. Y. Lin and S. Y. Lee. , HICSS2003]

Research Background(3/3)

Research Objective

Conventional method (KISP):
Stores known frequent patterns in a knowledge base (KB) and refers to them when needed

- However, among the large number of generated frequent sequential patterns, many are redundant

Proposed Method：
Reduce the number of candidate sequences by extracting only frequent closed sequential patterns to speed up the algorithm

  - Applied to real data and evaluated by execution time

Contents

Related Work_PrefixSpan(1/2)

PrefixSpan[3] : Discovers frequent sequential patterns through depth-first search

- Efficiency is achieved by using a projected database that keeps only the postfixes matched with the discovered patterns (prefixes)
  - *No need to generate all candidate combinations or count their frequencies*

- **Support value:** the number of occurrences in the dataset

Input: Database, minsup
Output: Frequent sequential patterns(P)
support(P) ≥ minsup

[3]J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu.
PrefixSpan : Mining sequential patterns efficiently by prefix- projected pattern growth. In Proceeding of 2001 International Conference on Data Engineering, pages 215–224, 2001.

Related Work_PrefixSpan(2/2)

- Method for Calculating Support Values
  - Obtain the list of position information for each sequence
  - For (Inspection A): {(1,0), (2,0), (5,0), (6,0)}
  - The number of elements in the list = the support value
- minsup=0.5 :
  Frequent sequential patterns of length 2

In the case of (Inspection A):
<(Inspection A), (Anesthesia)>,
<(Inspection A), (Surgery)>,
<(Inspection A), (Drug D)>,
<(Inspection A), (Nursing)>,
<(Inspection A), (Drug D, Nursing)>

ID	Sequence
1	(Inspection A, Inspection B), (Anesthesia), (Surgery), (Drug D, Nursing)
2	(Inspection A), (Anesthesia), (Surgery), (Drug D, Nursing)
3	(Drug E), (Anesthesia), (Surgery), (Drug D)
4	(Inspection C), (Anesthesia), (Surgery), (Drug F), (Nursing)
5	(Inspection A, Inspection C), (Drug E), (Anesthesia), (Surgery), (Drug D, Nursing)
6	(Inspection A, Inspection B, Inspection C), (Anesthesia), (Surgery), (Drug F, Nursing)

- When examining the above, also check the position information of sequences that do **not** meet minsup
  (e.g., <(Inspection A), (Inspection B)>, <(Inspection A), (Inspection C)>, etc.)

Related Work_KISP(1/2)

- KISP[5] : Interactive Sequential Pattern Mining Using a Knowledge Base (KB)
  - *Suitable for repeated execution with different minsup values*
  - KB stores previously discovered frequent patterns and their support values, along with the minimum minsup (KB.base)

- When the specified minsup is greater than KB.base
  - patterns that satisfy minsup can be directly retrieved from the KB,
  - resulting in a significant performance improvement

[5]Ming-Yen Lin and Suh-Yin Lee
,"Improving the Efficiency of Interactive Sequential Pattern Mining by Incremental Pattern Discovery" ,
Proceedings of the 36th Hawaii International Conference on System Sciences - 2003

Related Work_KISP(2/2)

- KISP：Issues in Analytical Efficiency
  - *Many redundant patterns exist among frequent sequential patterns*
- Focus is placed on Frequent Closed Sequential Patterns (FCSPs)
  - **FCSP:** No sequence β exists that contains sequence α as a subsequence with the same support count
  - In data such as medical records, closed sequences alone can represent the overall treatment process
- By extracting only closed sequential patterns
  - the number of patterns decreases, leading to faster algorithm performance

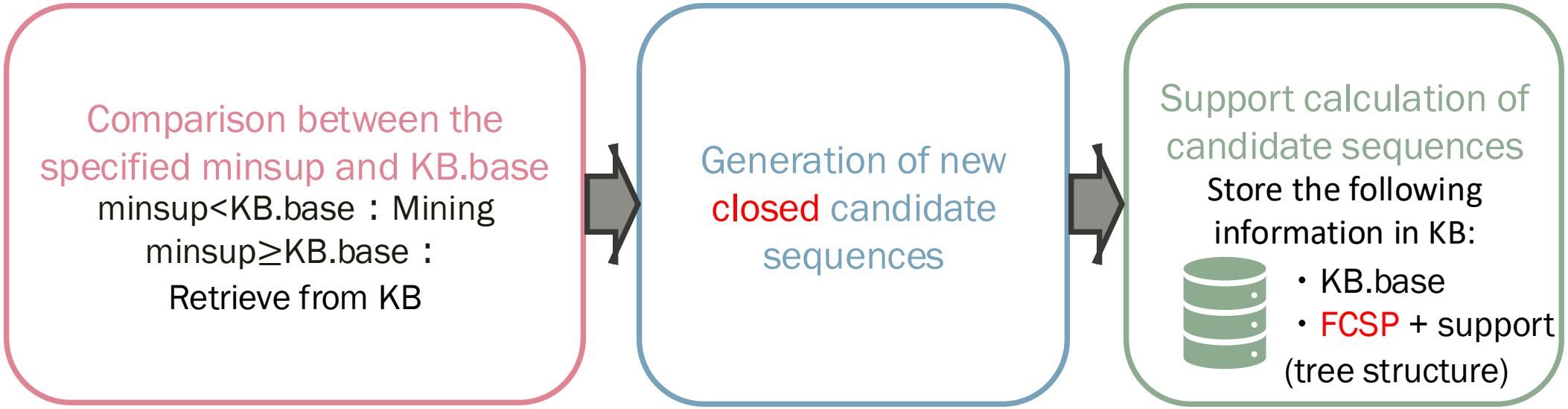
$$FCSP = \{\alpha \mid \alpha \in FSP, \nexists \beta \in FSP, \alpha \sqsubset \beta, Sup(\alpha) = Sup(\beta)\}$$

Contents

Proposed Method(1/2)

- Apply the mechanism of KISP that utilizes existing frequent patterns

Proposal①　Generate only closed sequences as candidates



Proposed Method(2/2)

Proposal ②　Structure of KB

- KB.base ： minimum minsup so far

- Support values and closed frequent sequential patterns:
  - *stored in a tree structure*

- Position information of candidate sequences:
  - *stored in a hash structure*
  - During subsequent mining, support values can be computed without re-checking the dataset

Contents

Experiments

Experiment Overview

① Verification of Execution Time Using the KB

② Verification of Execution Time with Closed Consideration

- Reduction in the number of candidate sequences

③ Retention of Candidate Sequence Position Information

Dataset used in this study

	Number of sequences	Average number of items per sequence	Size (MB)	Description
BMSWebView1	59,601	2.42	1.5MB	Clickstream data from an EC site
BMSWebView2	77,512	4.62	3.6MB	

References ：Fournier-Viger P., An Open-Source Data Mining Library, (https://www.philippe-fournier-viger.com/spmf/index.php?link=datasets.php)

Experiments

Experimental Environment

Server	Dell PowerEdge R740xd
CPU	Intel Xeon Gold 5218 16 cores x 2
OS	Ubuntu 24.04.1 LTS
Memory	64GB x 6
python	3.12.3

- The implementation of the proposed method was based on the source code of PrefixSpan
  - *References* : *https://github.com/chuanconggao/PrefixSpan-py*

Experiment①
Verification of Execution Time Using the KB

Dataset used
BMSWebView1

- The increment of minsup was varied from 0.01 to 0.07
  - Smaller increments result in more executions
    - Each experiment was run 3 times, and the average execution time was calculated
- Result: The proposed method is faster as the number of executions increases
  - *PrefixSpan: Re-mines the dataset whenever minsup changes*
  - **Proposed method:** Performs mining only for minsup = 0.01; for subsequent minsup values, it retrieves patterns from known frequent sequential patterns

Increment step	PrefixSpan(s)	Interactive PrefixSpan(s)	Interactive PrefixSpan/PrefixSpan
0.0005	14.86	15.16	102%
0.0004	18.43	15.29	83%
0.0003	24.75	16.02	65%

Experiment①
Verification of Execution Time Using the KB

- We evaluated the case where minsup decreases
  - including small decrement steps
    - Other conditions were the same as on the previous page
- Result: The proposed method is faster as the number of executions increases
  - *similar to the case when minsup increases*

Dataset used
BMSWebView1

Decrement step	PrefixSpan(s)	Interactive PrefixSpan(s)	Interactive PrefixSpan/PrefixSpan
0.0005	14.67	27.35	186%
0.0004	18.44	29.71	161%
0.0003	24.75	34.41	139%
0.0002	36.74	40.91	111%
0.0001	73.98	62.67	85%

Experiment②
Verification of Execution Time with Closed Consideration

- Comparison of the proposed method with and without considering closed sequences
  - *Executed with minsup ranging from 0.00065 to 0.00070 in increments of 0.00001*

Results

- BMSWebView1
  - Considering closed sequences is faster; the difference in execution time is large
- BMSWebView2
  - Considering closed sequences is faster; the difference in execution time is small

	With Closed Check(s)	Without Closed Check(s)	With / Without Closed Check
BMSWebView1	392.39	816.75	48%
BMSWebView2	323.33	325.61	99%

Experiment②
Verification of Execution Time with Closed Consideration

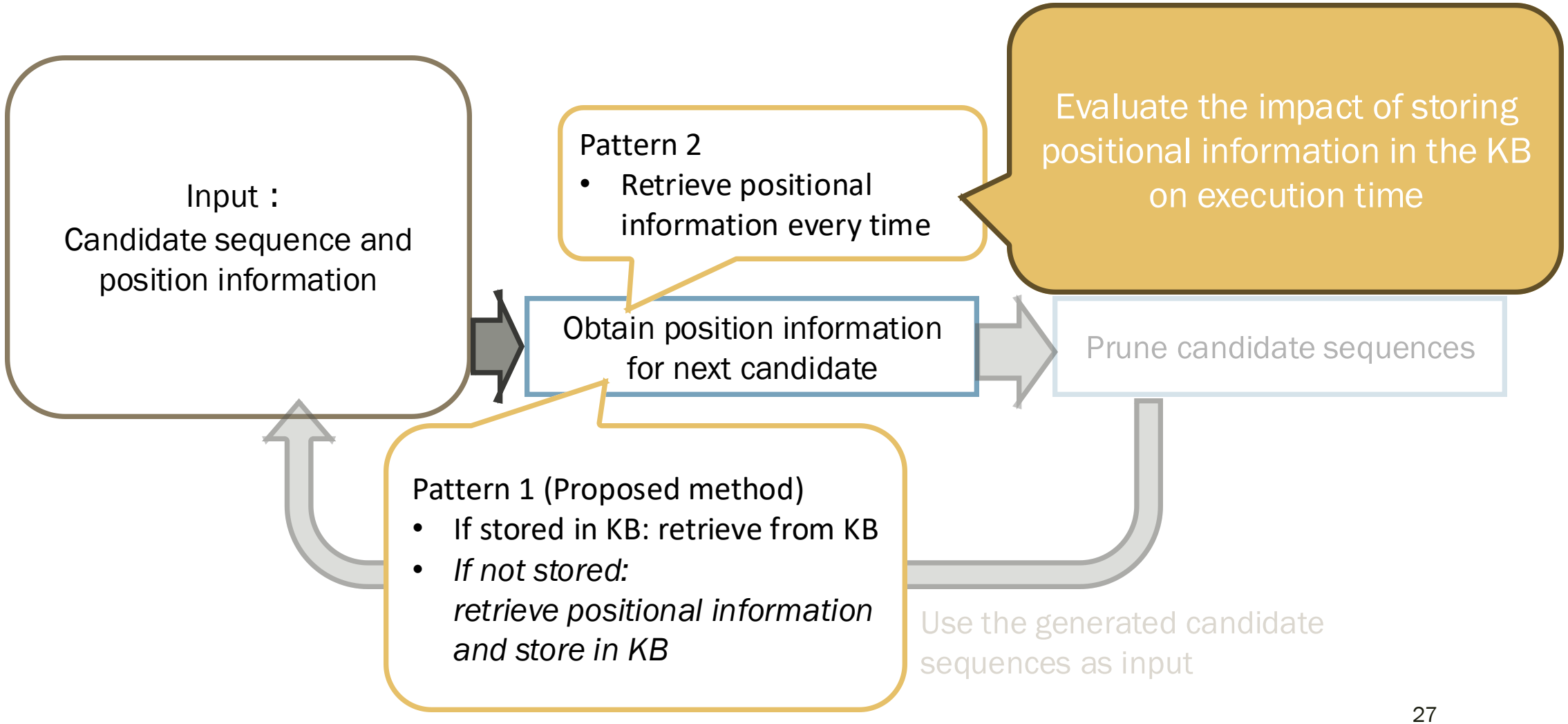
- Comparison of Candidate Sequence Generation

	With Closed Check	Without Closed Check	With / Without Closed Check
BMSWebView1	85,186,590	195,240,402	44%
BMSWebView2	53,330,915	67,693,468	79%

- For both datasets, considering closed sequences results in faster execution
  - Considering closed sequences reduces the number of candidate sequences, leading to faster processing
- The difference in execution time with or without closed sequences varies depending on the dataset
  - Difference in the number of candidate sequences

Considering closed sequences leads to faster execution
→ due to the reduction in the number of candidate sequences

Experiment③
Retention of Candidate Sequence Position Information



Experiment③
Retention of Candidate Sequence Position Information

- Minsup is expressed as a value rather than a ratio
  - Executed by decreasing from 60 to 55 in steps of 1
  - *Pattern 1: Position information stored*
  - *Pattern 2: Position information not stored*
  - *Result: Not storing position information is faster*
- **Reason:** The memory size for position information is too large (over 1.2 GB)

Dataset used
BMSWebView1

minsup	60	59	58	57	56	55
Pattern 1 (s)	43.42	487.45	539.31	600.99	674.84	753.35
Pattern 2 (s)	7.41	7.76	8.14	8.53	9.00	9.56
Memory Size(GB)	1.29	1.36	1.43	1.51	1.61	1.71

Contents

Conclusion

Summary

- ✓ Applied interactive SPM to facilitate finding the optimal minsup
- ✓ Demonstrated the effectiveness of using a KB
- ✓ Extracting only closed sequential patterns reduces the number of candidate sequences, leading to faster algorithm performance

Future Work

- KB: Structure for storing frequent sequential patterns
  - *Tree structure or hash structure*
- Investigate structures for storing position information
- Comparison with KISP