FHM+: Faster High-Utility Itemset Mining using Length Upper-Bound Reduction

Philippe Fournier-Viger[1], Jerry Chun-Wei Lin[2],
Quang-Huy Duong[3], Thu-Lan Dam[3,4],

[1] School of Natural Sciences and Humanities, Harbin Institute of Technology Shenzhen Graduate School, China
[2] School of Computer Science and Technology, Harbin Institute of Technology Shenzhen Graduate School, China
[3] College of Computer Science and Electronic Engineering, Hunan University, China
[4] Faculty of Information Technology, Hanoi University of Industry, Vietnam

High-utility itemset mining

Input

a transaction database

TID	Transaction
T_1	$(a,1),(b,5),(c,1),(d,3),(e,1),(f,5)$
T_2	$(b,4),(c,3),(d,3),(e,1)$
T_3	$(a,1),(c,1),(d,1)$
T_4	$(a,2),(c,6),(e,2),(g,5)$
T_5	$(b,2),(c,2),(e,1),(g,2)$

a unit profit table

Item	a	b	c	d	e	f	g
Profit	5	2	1	2	3	1	1

***minutil**: a minimum utility threshold set by the user (a positive integer)*

High-utility itemset mining

Input

a transaction database

TID	Transaction
T_1	$(a,1),(b,5),(c,1),(d,3),(e,1),(f,5)$
T_2	$(b,4),(c,3),(d,3),(e,1)$
T_3	$(a,1),(c,1),(d,1)$
T_4	$(a,2),(c,6),(e,2),(g,5)$
T_5	$(b,2),(c,2),(e,1),(g,2)$

a unit profit table

Item	a	b	c	d	e	f	g
Profit	5	2	1	2	3	1	1

minutil*: a minimum utility threshold set by the user* (*a positive integer*)

Output

All high-utility itemsets (itemsets having a $\text{utility} \geq minutil$)

For example, if *minutil = 33$*, the high-utility itemsets are:

{b,d,e} 36$ 2 transactions	{b,c,d} 34$ 2 transactions
{b,c,d,e} 40$ 2 transactions	{b,c,e} 37 $ 3 transactions

Utility calculation

a transaction database

TID	Transaction
T_1	$(a,1),(b,5),(c,1),(d,3),(e,1),(f,5)$
T_2	$(b,4),(c,3),(d,3),(e,1)$
T_3	$(a,1),(c,1),(d,1)$
T_4	$(a,2),(c,6),(e,2),(g,5)$
T_5	$(b,2),(c,2),(e,1),(g,2)$

a unit profit table

Item	a	b	c	d	e	f	g
Profit	5	2	1	2	3	1	1

The **utility** of the itemset {b,d,e} is calculated as follows:

$$u(\{b,d,e\}) = \underbrace{(5\times2)+(3\times2)+(3\times1)}_{\text{utility in transaction } T_1} + \underbrace{(4\times2)+(2\times3)+(1\times3)}_{\text{utility in transaction } T_2} = \mathbf{36\$}$$

A difficult task!

Why? the **utility** measure is neither monotonic nor anti-monotonic.

a transaction database

TID	Transaction
T_1	$(a,1),(b,5),(c,1),(d,3),(e,1),(f,5)$
T_2	$(b,4),(c,3),(d,3),(e,1)$
T_3	$(a,1),(c,1),(d,1)$
T_4	$(a,2),(c,6),(e,2),(g,5)$
T_5	$(b,2),(c,2),(e,1),(g,2)$

a unit profit table

Item	a	b	c	d	e	f	g
Profit	5	2	1	2	3	1	1

u({**b,d,e**}) = **36$**

u({**b,c,d,e**}) = **40$**

u({**a,b,c,d,e,f**}) = 30 **$**

Previous work

- **Several algorithms**:
  - EFIM, FHM, BAHUI, IHUP, Two-phase, Umining…)
- **Key idea:**
  - calculate an upper-bound on the utility of itemsets (**e.g**. the **TWU**) that is monotonic to be able to prune the search space.

The TWU upper-bound

TWU of an itemset: the sum of the utility of transactions containing the itemset.

a transaction database

TID	Transaction
T_1	$(a,1),(b,5),(c,1),(d,3),(e,1),(f,5)$
T_2	$(b,4),(c,3),(d,3),(e,1)$
T_3	$(a,1),(c,1),(d,1)$
T_4	$(a,2),(c,6),(e,2),(g,5)$
T_5	$(b,2),(c,2),(e,1),(g,2)$

a unit profit table

Item	a	b	c	d	e	f	g
Profit	5	2	1	2	3	1	1

TWU({c,d}) = TU(T1)+TU(T2)+TU(T3)
= 30 + 20 + 8 = 58

Property: The TWU of an itemset is an upper bound on its utility, and all its supersets.

Problem

– Current algorithms are useful for discovering **profitable itemsets**.
– But can find a large amount of itemsets
– **Long itemsets** are often **infrequent** or too **specific**
  {mapleSyrup, pancake,orange,cheese,cereal}
  {mapleSyrup, pancake}
– **A solution:** use a length constraint

Naïve approach

- Introduce a parameter *maxlength*
- Modify an algorithm to not extend an itemset with an item if its number of items is equal to *maxlength*
- **Drawback**:
  - does not reduce upper-bounds on the utilities of itemsets to prune the search space.
  - having tight upper-bounds is crucial for pruning the search space efficiently.

Contribution

- We introduce the idea of reducing upper-bounds on the utilities of itemsets using length constraints.
- Two novel upper-bounds
  - RTWU
  - Revised Remaining Utility
- A modified algorithm called FHM+

Largest utilities of a transaction

The ***maxLength*** largest utility values in each transaction:

maxLength = 3

TID	Transaction
T_1	$(a,1),(b,5),(c,1),(d,3),(e,1),(f,5)$
T_2	$(b,4),(c,3),(d,3),(e,1)$
T_3	$(a,1),(c,1),(d,1)$
T_4	$(a,2),(c,6),(e,2),(g,5)$
T_5	$(b,2),(c,2),(e,1),(g,2)$

$L(T_1)$ = 10, 6, 5

RTU(T_1) = 21

Item	a	b	c	d	e	f	g
Profit	5	2	1	2	3	1	1

Largest utilities of a transaction

Find the ***maxLength*** largest utility values in each transaction:

maxLength = 3

TID	Transaction
T_1	$(a,1),(b,5),(c,1),(d,3),(e,1),(f,5)$
T_2	$(b,4),(c,3),(d,3),(e,1)$
T_3	$(a,1),(c,1),(d,1)$
T_4	$(a,2),(c,6),(e,2),(g,5)$
T_5	$(b,2),(c,2),(e,1),(g,2)$

Item	a	b	c	d	e	f	g
Profit	5	2	1	2	3	1	1

$L(T_1)$ = 10, 6, 5
$L(T_2)$ = 8, 6, 3
$L(T_3)$ = 5, 2, 1
$L(T_4)$ = 10,6,6
$L(T_5)$ = 4, 3, 2

$RTU(T_1)$ = 21
$RTU(T_2)$ = 17
$RTU(T_3)$ = 8
$RTU(T_4)$ = 22
$RTU(T_5)$ = 9

The RTWU upper-bound

- **RTWU of an itemset X** : The sum of the **RTU** values of transactions containing **X**
- It is an upper-bound on its utility and the utility of its supersets

X = {c,d} *maxLength = 3*

TID	Transaction
T_1	$(a,1),(b,5),(c,1),(d,3),(e,1),(f,5)$
T_2	$(b,4),(c,3),(d,3),(e,1)$
T_3	$(a,1),(c,1),(d,1)$
T_4	$(a,2),(c,6),(e,2),(g,5)$
T_5	$(b,2),(c,2),(e,1),(g,2)$

$RTU(T_1) = 21$
$RTU(T_2) = 17$
$RTU(T_3) = 8$
$RTU(T_4) = 22$
$RTU(T_5) = 9$

RTWU({c,d}) = RTU(T1) + RTU(T2) + RTU(T3)
= 21 + 17 + 8 = **48**

Largest utilities w.r.t an itemset in a transaction

Given an itemset **X,** find the ***maxLength*** **- |X|** largest utility values in the transaction that can extend **X**:

X = {a}   *maxLength* = 3

TID	Transaction
T_1	$(a, 1), (b, 5), (c, 1), (d, 3), (e, 1), (f, 5)$
T_2	$(b, 4), (c, 3), (d, 3), (e, 1)$
T_3	$(a, 1), (c, 1), (d, 1)$
T_4	$(a, 2), (c, 6), (e, 2), (g, 5)$
T_5	$(b, 2), (c, 2), (e, 1), (g, 2)$

$RRU(T_1) = 10, 6$

$RRU(T_3) = 2, 1$

$RRU(T_4) = 6,6$

Item	a	b	c	d	e	f	g
Profit	5	2	1	2	3	1	1

The Revised Remaining Utility

- **RREU of an itemset X**: The sum of the utilities of the itemset + the largest remaining utilities w.r.t that itemset
- An upper-bound on the utility of **X** and the utility of its supersets

X = {a} *maxLength = 3*

TID	Transaction
T_1	$(a,1),(b,5),(c,1),(d,3),(e,1),(f,5)$
T_2	$(b,4),(c,3),(d,3),(e,1)$
T_3	$(a,1),(c,1),(d,1)$
T_4	$(a,2),(c,6),(e,2),(g,5)$
T_5	$(b,2),(c,2),(e,1),(g,2)$

RRU(T_1) = 10, 6 *U*(T_1) = 5

RRU(T_3) = 2, 1 *U*(T_3) = 5

RRU(T_4) = 6,6 *U*(T_4) = 10

Item	a	b	c	d	e	f	g
Profit	5	2	1	2	3	1	1

The Revised Remaining Utility

- **RREU of an itemset X**: The sum of the utilities of the itemset + the largest remaining utilities w.r.t that itemset
- An upper-bound on the utility of **X** and the utility of its supersets

X = {a} *maxLength = 3*

TID	Transaction
T_1	$(a,1),(b,5),(c,1),(d,3),(e,1),(f,5)$
T_2	$(b,4),(c,3),(d,3),(e,1)$
T_3	$(a,1),(c,1),(d,1)$
T_4	$(a,2),(c,6),(e,2),(g,5)$
T_5	$(b,2),(c,2),(e,1),(g,2)$

$RRU(T_1) = 10, 6$ $U(T_1) = 5$

$RRU(T_3) = 2, 1$ $U(T_3) = 5$

$RRU(T_4) = 6,6$ $U(T_4) = 10$

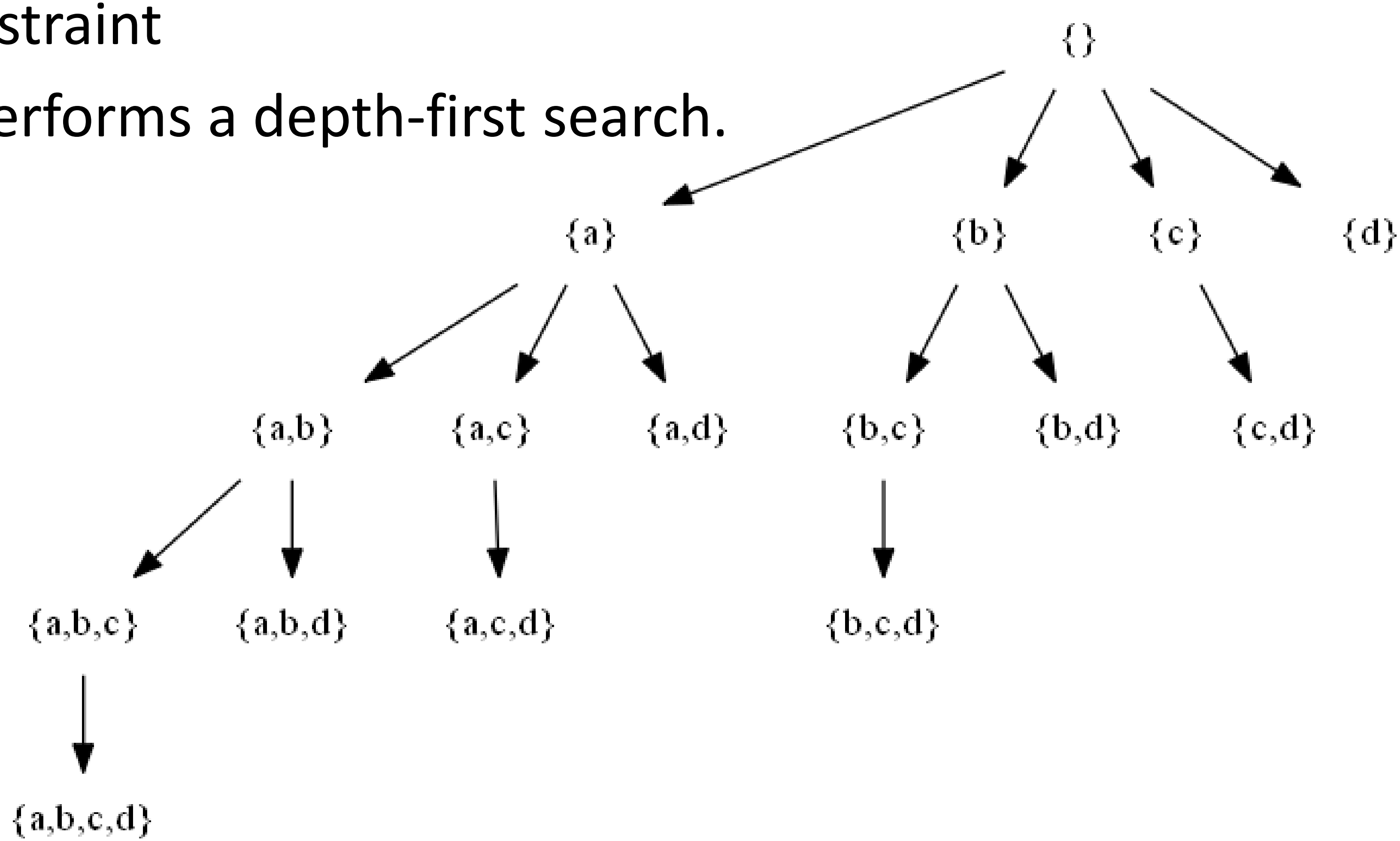
Item	a	b	c	d	e	f	g
Profit	5	2	1	2	3	1	1

RREU({a}) = RRU(T1) + U(T1) + RTU(T2) +U(T2) + RRU(T3) + U(T3)

= (16 + 5) + (3 + 5) + (12 + 10) = **52**

The FHM+ algorithm

- An algorithm for mining high utility-itemsets with length constraint
- It performs a depth-first search.



- It applies pruning strategies to prune the search space based on upper-bounds on the utility.

Creating utility-lists

Scan the database to create a utility-list for each itemset

Itemset {a}

TID	Utility	Largest utilities
T1	5	{10,6]
T3	5	{2, 1}
T4	10	{6, 6}

Itemset {b}

TID	Utility	Largest utilities
T1	10	{6, 3]
T2	8	{6, 3}
T5	4	{3, 2}

Creating utility-lists

Scan the database to create a utility-list for each itemset

Itemset {a}

TID	Utility	Largest utilities
T1	5	{10,6]
T3	5	{2, 1}
T4	10	{6, 6}

$\underbrace{\textbf{20\$} \qquad \textbf{32\$}}$

Upper-bound: **52$**

Itemset {b}

TID	Utility	Largest utilities
T1	10	{6, 3]
T2	8	{6, 3}
T5	4	{3, 2}

Generating larger itemsets

Itemset {a}

TID	Utility	Largest utilities
T1	5	{10,6]
T3	5	{2, 1}
T4	10	{6, 6}

Itemset {b}

TID	Utility	Largest utilities
T1	10	{6, 3]
T2	8	{6, 3}
T5	4	{3, 2}

Itemset {a,b}

TID	Utility	Largest utilities
T1	15	{10}

Generating larger itemsets

Itemset {a}

TID	Utility	Largest utilities
T1	5	{10,6]
T3	5	{2, 1}
T4	10	{6, 6}

Itemset {b}

TID	Utility	Largest utilities
T1	10	{6, 3]
T2	8	{6, 3}
T5	4	{3, 2}

Itemset {a,b}

TID	Utility	Largest utilities
T1	15	{10}

15$** **10$

Upper-bound: **25$**

Pseudocode

Algorithm 1. The FHM+ algorithm

input : D: a transaction database, $minutil, minlength, maxlength$: user-specified parameters
output: the set of high-utility itemsets

Scan D once to calculate the RTWU of single items;
$I^* \leftarrow$ each item i such that RTWU$(i) \geq minutil$;
Let $\succ$ be the total order of RTWU ascending values on I^*;
Scan D to build the revised utility-list of each item $i \in I^*$ and build the $EUCS$ structure;
if $minlength \leq 1$ **then** output each item $i \in I^*$ such that SUM$(\{i\}.utilitylist.iutils) \geq minutil$ **if** $maxlength > 1$ **then** `Search` ($\emptyset$, I^*, $minutil$, $minlength$, $maxlength$, $EUCS$)

Algorithm 2. The *Search* procedure

input : P: an itemset, $ExtensionsOfP$: a set of extensions of P, $minutil$, $minlength$, $maxlength$: user-specified parameters, $EUCS$: the $EUCS$
output: the set of high-utility itemsets

foreach *itemset* $Px \in ExtensionsOfP$ **do**
    if $SUM(Px.utilitylist.iutils)+SUM(Px.utilitylist.llist) \geq minutil$ **then**
        $ExtensionsOfPx \leftarrow \emptyset$;
        foreach *itemset* $Py \in ExtensionsOfP$ *such that* $y \succ x$ **do**
            if $\exists(x, y, c) \in EUCS$ *such that* $c \geq minutil$ **then**
                $Pxy \leftarrow Px \cup Py$;
                $Pxy.utilitylist \leftarrow$ `Construct` (P, Px, Py);
                $ExtensionsOfPx \leftarrow ExtensionsOfPx \cup Pxy$;
                if $SUM(Pxy.utilitylist.iutils) \geq minutil$ *and* $minlength \leq |Pxy| \leq maxlength$ **then** output Px
            end
        end
        if $|Pxy| < maxlength$ **then** `Search` (Px, $ExtensionsOfPx$, $minutil$)
    end
end

Experimental Evaluation

Datasets' characterictics

Dataset	transaction count	distinct item count	average transaction length
Chainstore	1,112,949	46,086	7.2
Retail	88,162	16,470	10.3
Mushroom	8,124	119	23

Retail and **Chainstore** are real-life transaction datasets from retail stores.

Mushroom is a dense dataset with long transactions

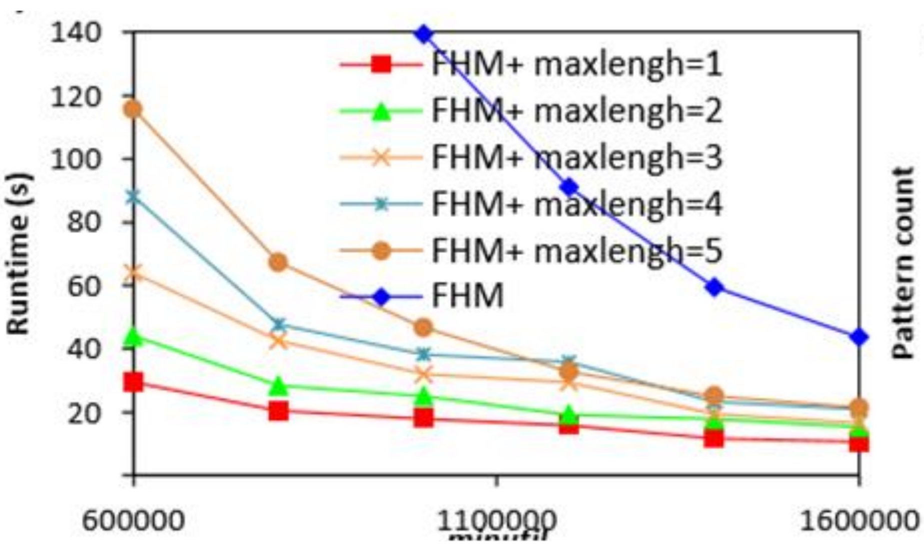
Experimental Evaluation

- We compared the performance:
  - FHM
  - FHM+ with maxLength varied from 1 to 5
- We varied the *minutil* threshold and measured
  - execution time
  - number of patterns
  - memory usage
- Java, 12 GB of RAM, Windows 7, 64 bit Core i5 CPU

Chainstore

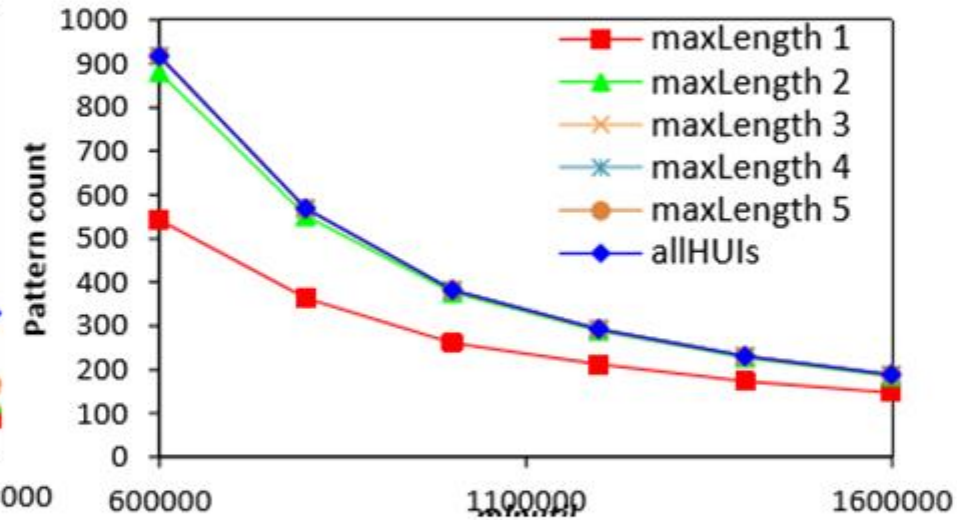
Execution Time

3 to 10 times faster



Number of patterns

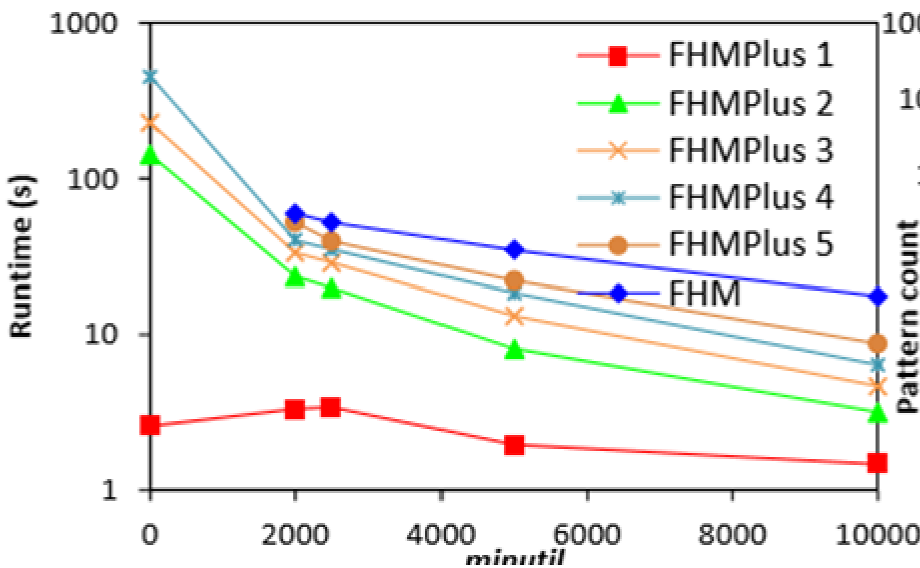
up to 50% less patterns



Retail

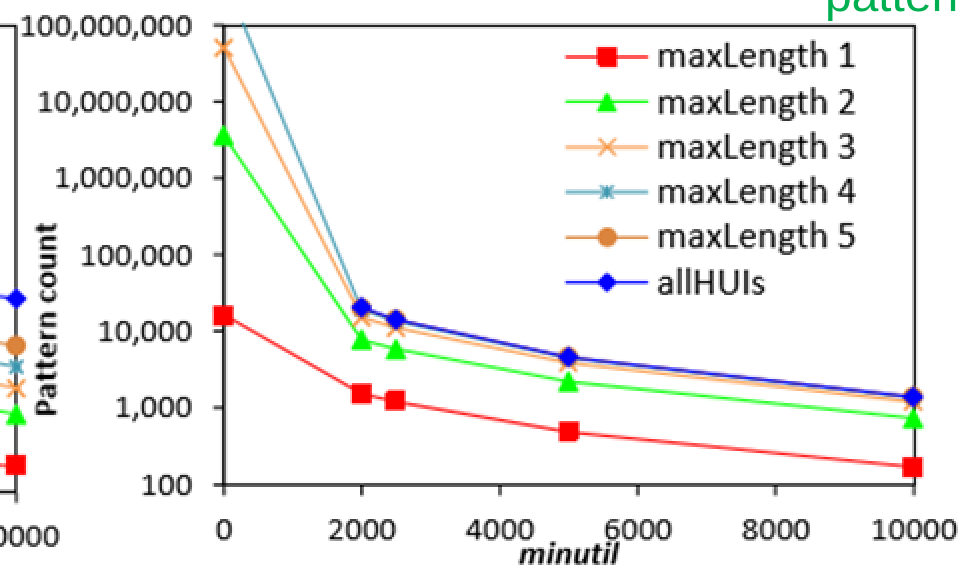
Execution Time

2 to 17 times faster



Number of patterns

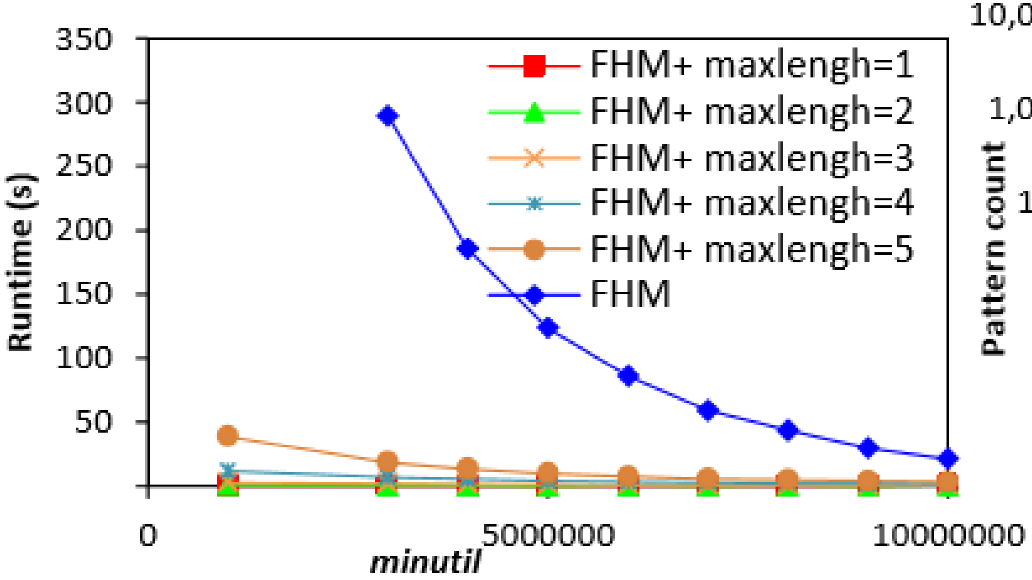
up to 13 times less patterns



Mushroom

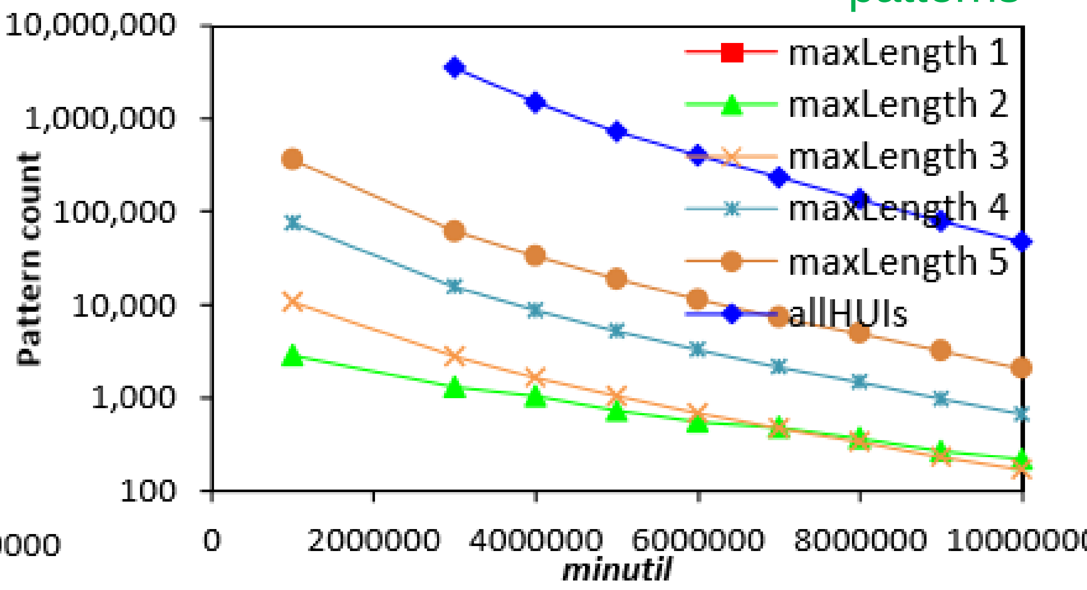
Execution Time

15 to 1400 times faster



Number of patterns

up to 2,700 times less patterns



Maximum Memory usage (MB)

Dataset	Reduction
Chainstore	5% to 50%
Retail	5% to 50%
Mushroom	25 % to 50 %

Efficiency vs Naïve approach

Dataset	FHM+
Chainstore	up to 4 times faster
Retail	up to 2 times faster
Mushroom	up to 2 times faster

Conclusion

- Contribution:
  - ➢ Novel algorithm for mining **high utility itemsets** while considering the **length constraint** named **FHM+**
  - ➢ Novel concept of **Length upper-bound reduction**
  - ➢ Two new upper-bounds: **revised TWU** and **revised remaining utility**
- Experimental results:
  - **FHM+** can greatly reduce execution time, memory usage, and the number of patterns founds
- Source code and datasets available as part of the **SPMF data mining library** (GPL 3).

Open source Java data mining software, **120 algorithms**
http://www.phillippe-fournier-viger.com/spmf/

Thank you. Questions?

Open source Java data mining software, **120 algorithms**
http://www.phillippe-fournier-viger.com/spmf/

References

- Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proceedings of the International Conference on Very Large Databases, pp. 487–499 (1994)
- Fournier-Viger, P., Wu, C.-W., Zida, S., Tseng, V.S.: FHM: faster high-utility itemset mining using estimated utility co-occurrence pruning. In: Andreasen, T., Christiansen, H., Cubero, J.-C., Ra´s, Z.W. (eds.) ISMIS 2014. LNCS, vol. 8502, pp. 83–92. Springer, Heidelberg (2014)
- Fournier-Viger, P., Gomariz, A., Gueniche, T., Soltani, A., Wu, C., Tseng, V.S.: SPMF: a Java open-source pattern mining library. J. Mach. Learn. Res. (JMLR) 15, 3389–3393 (2014)
- Pei, J., Han, J.: Constrained frequent pattern mining: a pattern-growth view. ACM SIGKDD Explor. Newsl. 4(1), 31–39 (2012)
- Lan, G.C., Hong, T.P., Tseng, V.S.: An efficient projection-based indexing approach for mining high utility itemsets. Knowl. Inf. Syst. 38(1), 85–107 (2014)
- Krishnamoorthy, S.: Pruning strategies for mining high utility itemsets. Expert Syst. Appl. 42(5), 2371–2381 (2015)

References (cont'd)

- Lin, J.C.-W., Gan, W., Hong, T.-P., Pan, J.-S.: Incrementally updating high-utility itemsets with transaction insertion. In: Luo, X., Yu, J.X., Li, Z. (eds.) ADMA 2014. LNCS, vol. 8933, pp. 44–56. Springer, Heidelberg (2014)
- Song, W., Liu, Y., Li, J.: BAHUI: fast and memory efficient mining of high utility itemsets based on bitmap. Int. J. Data Warehous. Min. 10(1), 1–15 (2014)
- Liu, M., Qu, J.: Mining high utility itemsets without candidate generation. In: Proceedings of the 22nd ACM International Conference on Information and Knowledge Management, pp. 55–64 (2012)
- Liu, Y., Liao, W., Choudhary, A.K.: A two-phase algorithm for fast discovery of high utility itemsets. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 689–695. Springer, Heidelberg (2005)
- Tseng, V.S., Shie, B.-E., Wu, C.-W., Yu, P.S.: Efficient algorithms for mining high utility itemsets from transactional databases. IEEE Trans. Knowl. Data Eng. 25(8), 1772–1786 (2013)
- Zida, S., Fournier-Viger, P., Wu, C.-W., Lin, J.C.W., Tseng, V.S.: Efficient mining of high utility sequential rules. In: Proceedings of the 11th International Conference on Machine Learning and Data Mining, pp. 1–15 (2015)
- Zida, S., Fournier-Viger, P., Lin, J.C.-W., Wu, C.-W., Tseng, V.S.: EFIM: a highly efficient algorithm for high-utility itemset mining. In: Sidorov, G., Galicia-Haro, S.N. (eds.) MICAI 2015. LNCS, vol. 9413, pp. 530–546. Springer, Heidelberg (2015). doi:10.1007/978-3-319-27060-9 44