

# Mining Minimal High-Utility Itemsets

Philippe Fournier-Viger<sup>1</sup>(✉), Jerry Chun-Wei Lin<sup>2</sup>, Cheng-Wei Wu<sup>3</sup>,  
Vincent S. Tseng<sup>3</sup>, and Usef Faghihi<sup>4</sup>

- <sup>1</sup> Harbin Institute of Technology Shenzhen, China
- <sup>2</sup> National Chiao Tung University, Taiwan
- <sup>3</sup> University of Indianapolis, Indianapolis, USA



# 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 utility  $\geq$  minutil)

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

<b>{b,d,e}</b> 36\$ 2 transactions	<b>{b,c,d}</b> 34\$ 2 transactions
<b>{b,c,d,e}</b> 40\$ 2 transactions	<b>{b,c,e}</b> 37 \$ 3 transactions

# Utility calculation

a transaction database

TID	Transaction
$T_1$	(a, 1), ( <u>b, 5</u> ), (c, 1), ( <u>d, 3</u> ), ( <u>e, 1</u> ), (f, 5)
$T_2$	( <u>b, 4</u> ), (c, 3), ( <u>d, 3</u> ), ( <u>e, 1</u> )
$T_3$	(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	<u>2</u>	1	<u>2</u>	<u>3</u>	1	1

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

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

utility in  
transaction  $T_1$

utility in  
transaction  $T_2$

# Challenge in Utility Mining

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 measure is not **monotonic** or **anti-monotonic**

$$u(\{a,b,c,d,e\}) = 25 \$$$

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

$$u(\{b,d\}) = 30 \$$$

The utility of an itemset can be lower, greater or equal to the utility of its subsets.

Previous work uses monotonic upper-bounds to reduce the search space.

# Problem

High-utility itemset mining

- is **useful** for discovering **profitable itemsets**.
- but it can find **a large amount of itemsets**
- **some itemsets are redundant**
- **solution:**
  - discover **concise representations of HUIs**
  - Previous work →

# Concise representations

- **Maximal HUIs:** HUIs not included in another HUI.
  - [GUIDE](#)
- **Closed HUIs:** HUIs having no supersets appearing in the same transactions
  - [CHUI-Miner, CHUD...](#)
- **Generators of HUIs:** smallest set of itemsets common to a set of transactions containing a HUI
  - [GHUI-Miner...](#)

# Limitations of previous work

- Other algorithms often find many long itemsets
- It is easier to co-promote a small set of items targeted at many customers rather than a large set of items targeted at few customers.
- **Proposal:**
  - **Minimal HUIs:** smallest sets of items that yield a high profit.
  - Not considered in previous work.



# What is a minimal high utility itemset?

A **MinHUI** is a **high-utility itemset** that has no proper subset that is a high-utility itemset.

**For example:**

$\{b,d,e\}$ 36\$ 2 transactions	$\{b,c,d\}$ 34\$ 2 transactions
<del><math>\{b,c,d,e\}</math> 40\$ 2 transactions</del>	<del><math>\{b,c,e\}</math> 37 \$ 3 transactions</del>

# Interesting properties

## Property 2

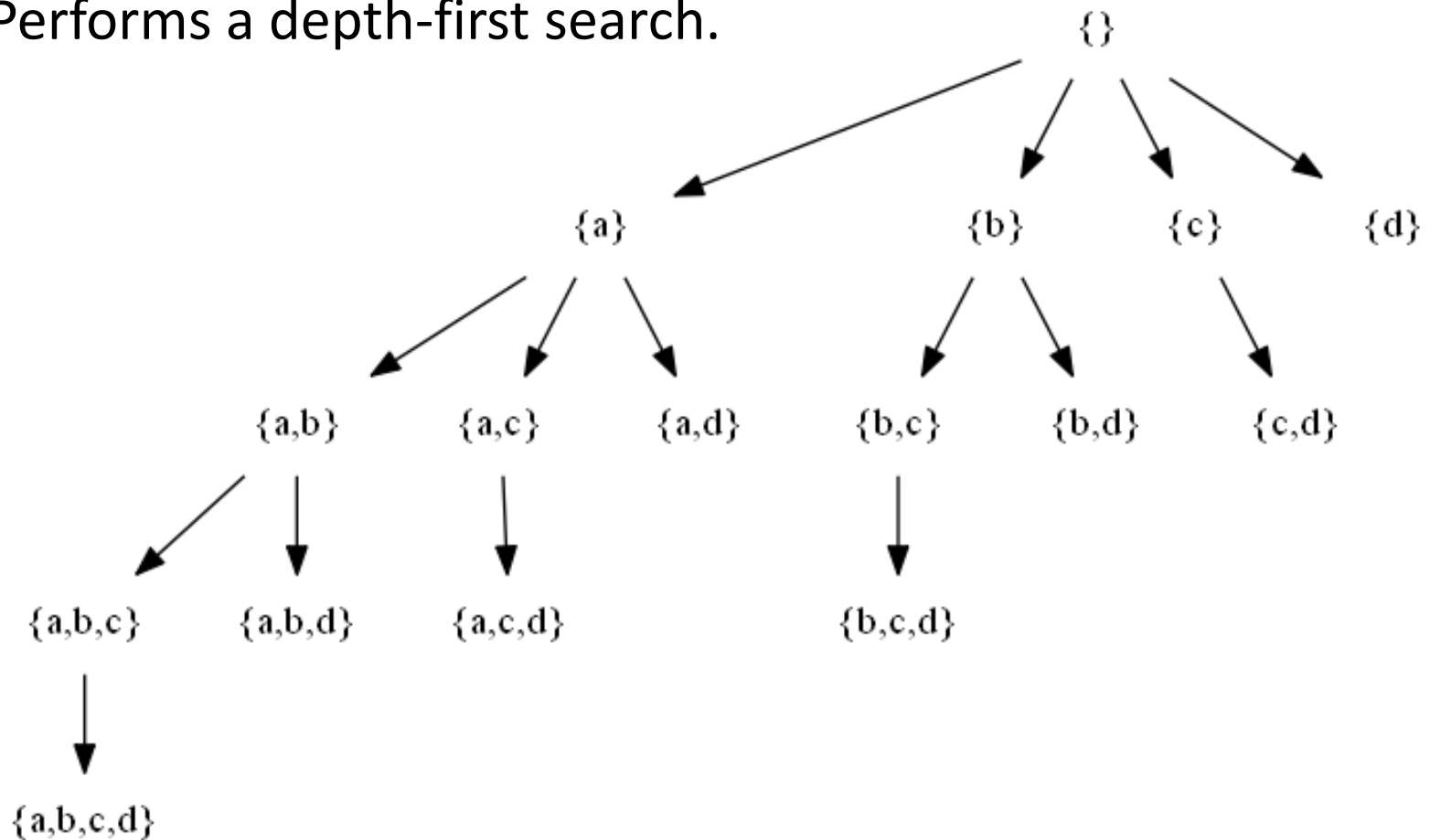
- If an itemset  $X$  is a **MinHUI**, then its subsets and supersets are not **MinHUIs**.

## Property 1

- If *minutil* is lowered, the number of **MinHUIs** may increase, decrease or stay the same.
- If *minutil* is set to **0**, the number of **MinHUIs** is equal to the number of items.

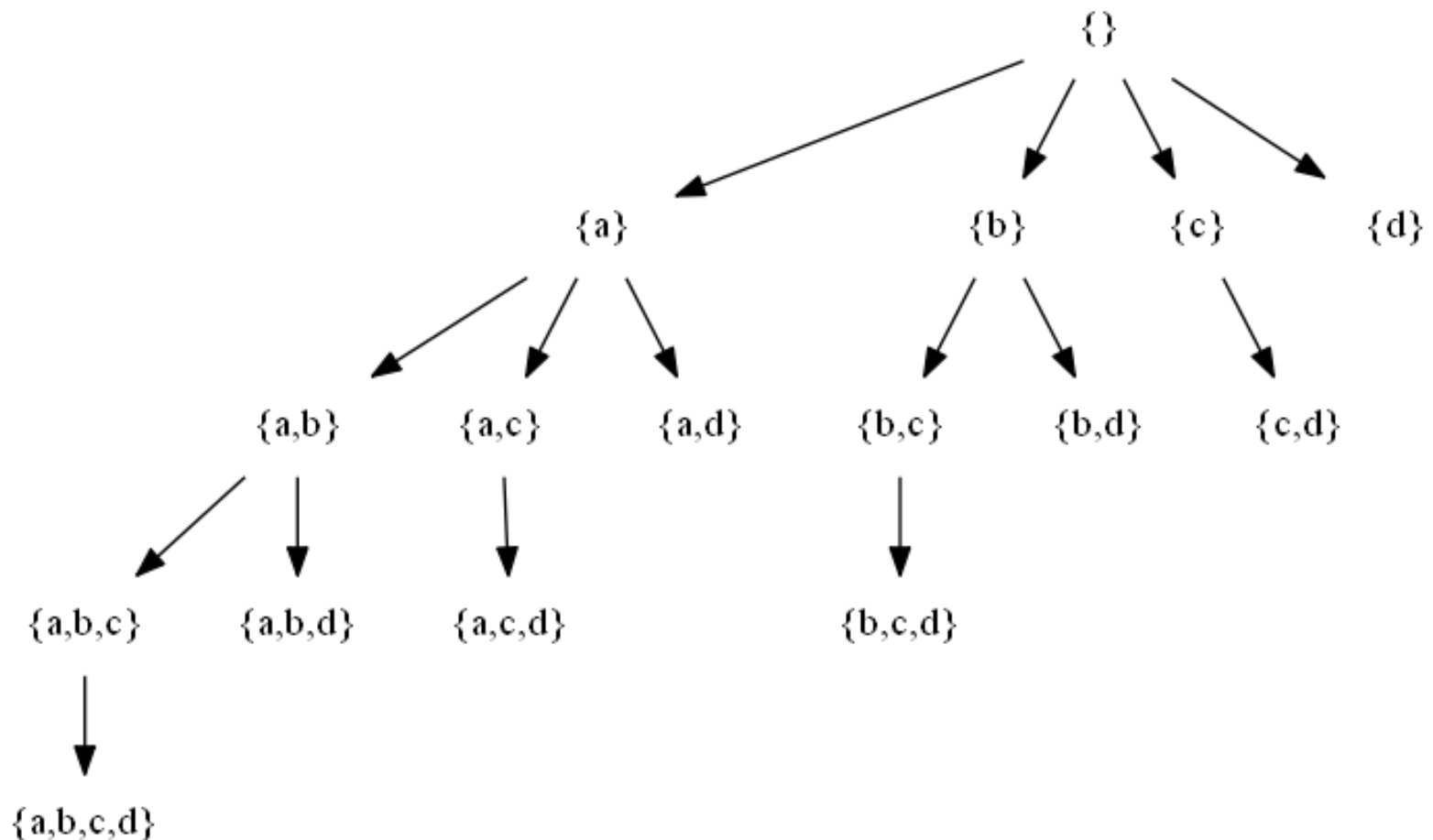
# The MinFHM algorithm

- An algorithm for mining minimal high utility-itemsets
- Extends FHM
- Performs a depth-first search.



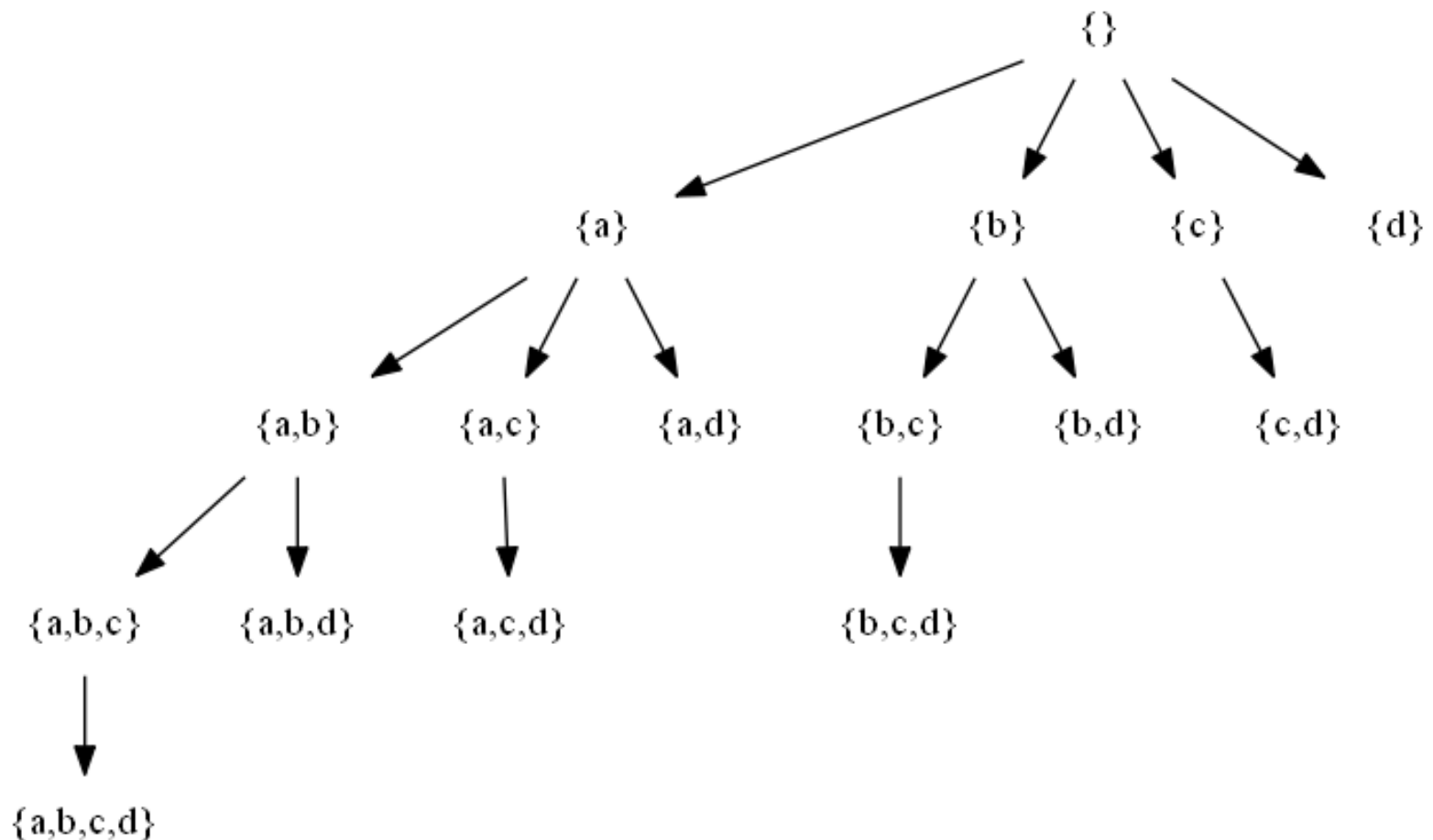
# The MinFHM algorithm

- Applies Property 2 to prune the search space.
- If an itemset is a MinHUI, its supersets are not MinHUIs



# The MinFHM algorithm

Calculates an upper bound on the utility of extensions of each itemset to decide whether its extensions should be explored.



# Creating utility-lists of items

The algorithm scans the database to create a *utility-list* structure for each item

Itemset {a}

TID	Utility	Remaining Utility
T1	5	20
T3	5	3
T4	10	12

Itemset {d}

TID	Utility	Remaining Utility
T1	6	3
T2	6	3
T3	2	0

# Calculating their utility

The utility of each item is calculated to determine if it is a high utility itemset

Itemset {a}

TID	Utility	Remaining Utility
T1	5	20
T3	5	3
T4	10	12

Utility: **20\$**

Itemset {d}

TID	Utility	Remaining Utility
T1	6	3
T2	6	3
T3	2	0

Utility: **14\$**

# Calculating their upper-bounds

An upper-bound is calculated on the utility of extensions of each item.

Itemset {a}

TID	Utility	Remaining Utility
T1	5	20
T3	5	3
T4	10	12

20\$      35\$

Upper-bound: **55\$**

Itemset {d}

TID	Utility	Remaining Utility
T1	6	3
T2	6	3
T3	2	0

This indicates that extensions of {a} cannot have a utility higher than 55\$



# Generating a larger itemset

Itemset {a}

TID	Utility	Remaining Utility
T1	5	20
T3	5	3
T4	10	12

Itemset {d}

TID	Utility	Remaining Utility
T1	6	3
T2	6	3
T3	2	0



Itemset {a,d}

TID	Utility	Remaining Utility
T1	11	3
T3	7	0

Utility-lists of larger itemsets are generated by joining the utility-lists of some of its subsets. No need to scan the database!

# Calculating its utility

Itemset {a}

TID	Utility	Remaining Utility
T1	5	20
T3	5	3
T4	10	12

Itemset {d}

TID	Utility	Remaining Utility
T1	6	3
T2	6	3
T3	2	0



Itemset {a,d}

TID	Utility	Remaining Utility
T1	11	3
T3	7	0

Utility:

**18\$**

# Calculating its upper-bound

Itemset {a}

TID	Utility	Remaining Utility
T1	5	20
T3	5	3
T4	10	12

Itemset {d}

TID	Utility	Remaining Utility
T1	6	3
T2	6	3
T3	2	0



Itemset {a,d}

TID	Utility	Remaining Utility
T1	11	3
T2	7	0
<b>18\$</b>		<b>3\$</b>

Upper-bound: **21\$**

Extensions cannot have a utility greater than 21\$ <sup>19</sup>

# Pseudocode

---

## Algorithm 1. The MinFHM algorithm

---

**input** :  $D$ : a transaction database,  $minutil$ : a user-specified threshold

**output**: the set of high-utility itemsets

- 1 Scan  $D$  to calculate the TWU of single items;
  - 2  $I^* \leftarrow$  each item  $i$  such that  $TWU(i) \geq minutil$ ;
  - 3 Let  $\succ$  be the total order of TWU ascending values on  $I^*$ ;
  - 4 Scan  $D$  to build the utility-list of each item  $i \in I^*$  and build the  $EUCS$ ;
  - 5 Output each item  $i \in I^*$  such that  $SUM(\{i\}.utilitylist.iutils) \geq minutil$ ;
  - 6 Search  $(\emptyset, I^*, minutil, EUCS)$ ;
- 

---

## Algorithm 2. The Search procedure

---

**input** :  $P$ : an itemset,  $ExtensionsOfP$ : a set of extensions of  $P$ ,  $minutil$ : a user-specified threshold,  $EUCS$ : the  $EUCS$

**output**: the set of high-utility itemsets

- 1 **foreach** itemset  $P_x \in ExtensionsOfP$  **do**
  - 2 | **if**  $SUM(P_x.utilitylist.iutils) + SUM(P_x.utilitylist.rutils) \geq minutil$  **then**
  - 3 | |  $ExtensionsOfP_x \leftarrow \emptyset$ ;
  - 4 | | **foreach** itemset  $P_y \in ExtensionsOfP$  such that  $y \succ x$  **do**
  - 5 | | | **if**  $\exists(x, y, c) \in EUCS$  such that  $c \geq minutil$  **then**
  - 6 | | | |  $P_{xy} \leftarrow P_x \cup P_y$ ;
  - 7 | | | |  $P_{xy}.utilitylist \leftarrow \text{Construct}(P, P_x, P_y)$ ;
  - 8 | | | |  $ExtensionsOfP_x \leftarrow ExtensionsOfP_x \cup \{P_{xy}\}$ ;
  - 9 | | | | **if**  $SUM(P_{xy}.utilitylist.iutils) \geq minutil$  **then** output  $P_x$ ;
  - 10 | | | **end**
  - 11 | | **end**
  - 12 |  $Search(P_x, ExtensionsOfP_x, minutil)$ ;
  - 13 | **end**
  - 14 **end**
- 

---

## Algorithm 3. The Construct procedure

---

**input** :  $P$ : an itemset,  $P_x$ : the extension of  $P$  with an item  $x$ ,  $P_y$ : the extension of  $P$  with an item  $y$

**output**: the utility-list of  $P_{xy}$

- 1  $UtilityListOfP_{xy} \leftarrow \emptyset$ ;
  - 2 **foreach** tuple  $ex \in P_x.utilitylist$  **do**
  - 3 | **if**  $\exists ey \in P_y.utilitylist$  and  $ex.tid = ey.tid$  **then**
  - 4 | | **if**  $P.utilitylist \neq \emptyset$  **then**
  - 5 | | | Search element  $e \in P.utilitylist$  such that  $e.tid = ex.tid$ ;
  - 6 | | |  $exy \leftarrow (ex.tid, ex.iutil + ey.iutil - e.iutil, ey.rutil)$ ;
  - 7 | | **end**
  - 8 | | **else**
  - 9 | | |  $exy \leftarrow (ex.tid, ex.iutil + ey.iutil, ey.rutil)$ ;
  - 10 | | **end**
  - 11 | |  $UtilityListOfP_{xy} \leftarrow UtilityListOfP_{xy} \cup \{exy\}$ ;
  - 12 | **end**
  - 13 **end**
  - 14 **return**  $UtilityListP_{xy}$ ;
- 

Some details have not been explained in the presentation. See the paper for more details.

# Experimental Evaluation

## Datasets' characteristics

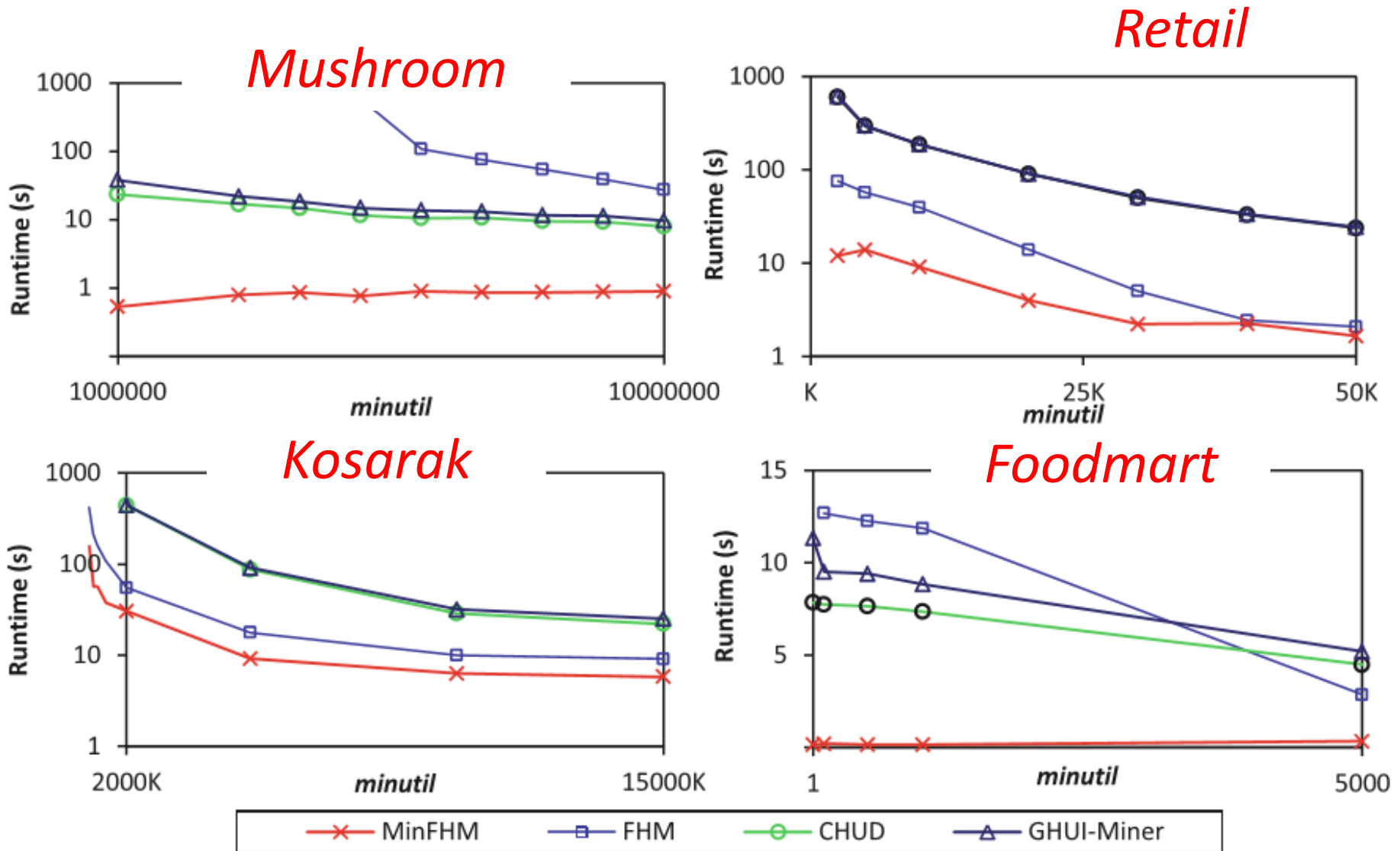
Dataset	transaction count	distinct item count	average transaction length
Mushroom	8,124	119	23.0
Kosarak	990,000	41,270	8.1
Retail	16,470	88,162	10.3
Foodmart	4,141	1,559	4.14

- **Foodmart** is a real-life transaction datasets from retail stores.
- External and internal utility values have been generated in the  $[1, 000]$  and  $[1, 5]$  intervals using a log-normal distribution

# Experimental Evaluation

- We compared the performance of **MinFHM** with
  - **FHM** for mining HUIs
  - **CHUD** for mining closed HUIs
  - **GHUI-Miner** for mining generators of HUIs
- We varied the *minutil* threshold and compared execution time, memory usage, number of patterns
- Computer with 12 GB of RAM, Java, Windows 7, 64 bit Core i5 Processor

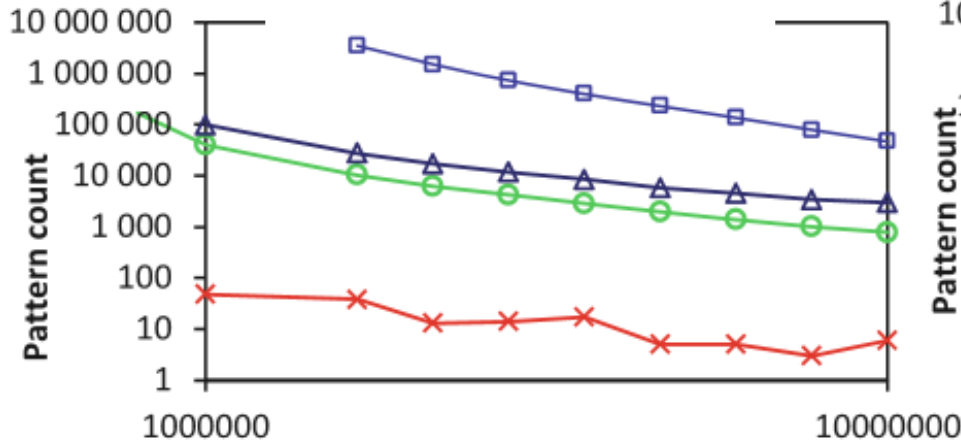
# Execution times



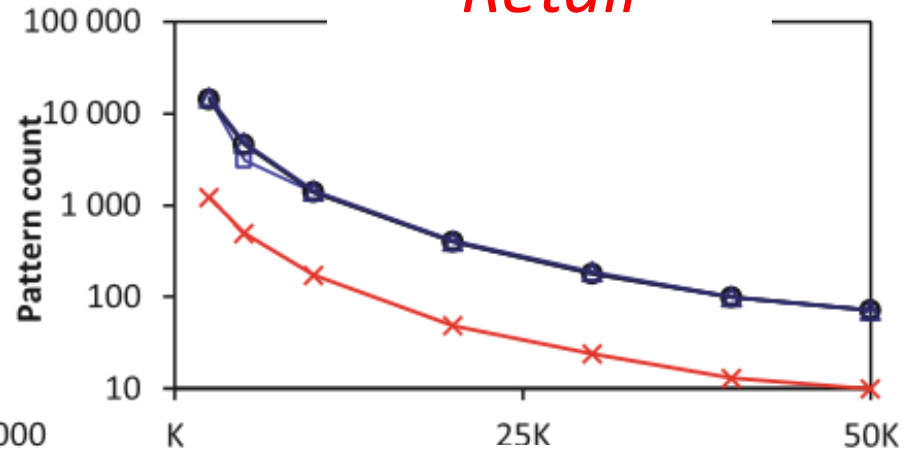
**MinFHM** is up to 800 times faster

# Number of patterns

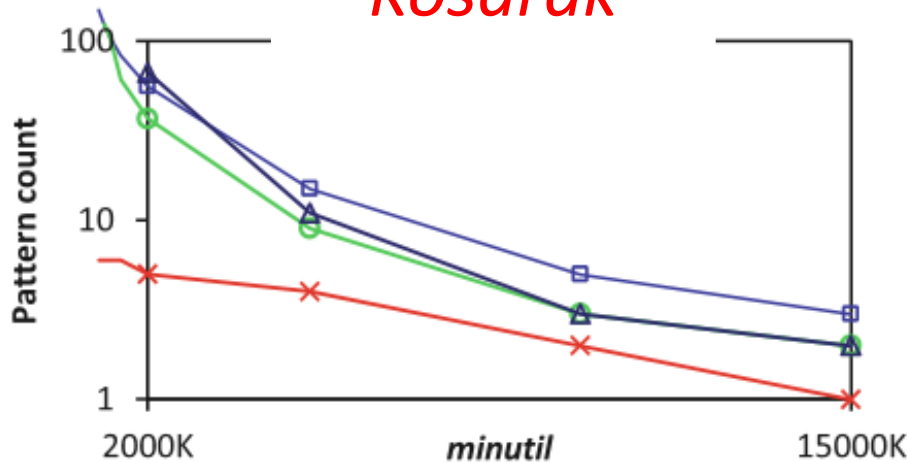
*Mushroom*



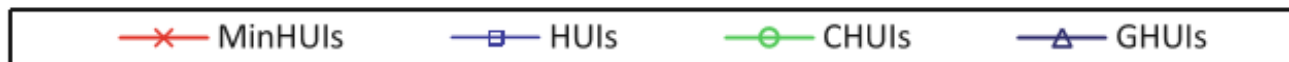
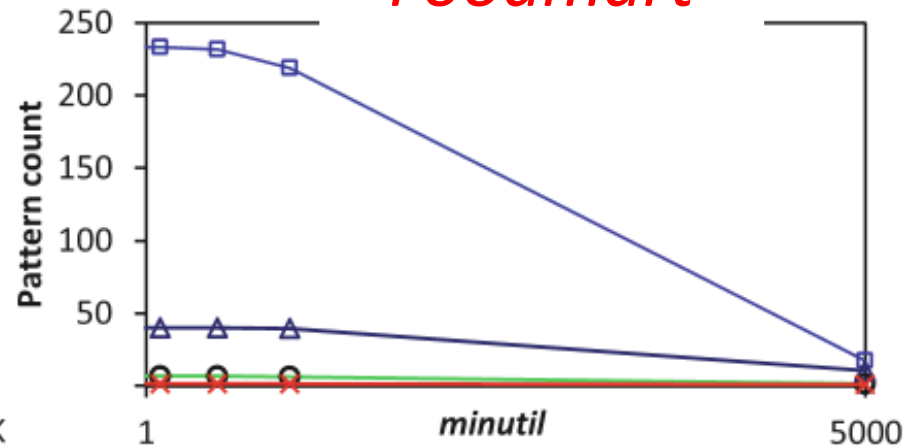
*Retail*



*Kosarak*



*Foodmart*



**MinFHM** is up to 900,000 times less patterns



# Conclusion

- Contribution:
  - New task : mining **minimal high utility itemsets**
  - Properties and an algorithm: **MinFHM**
- Experimental results:
  - up to **800 times faster**
  - Very compact: up to **900,000 times less patterns** than other concise representations of HUIs
- 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.philippe-fournier-viger.com/spmf/>

# Thank you. Questions?



**SPMF**

Open source Java data mining software, 120 algorithms  
<http://www.philippe-fournier-viger.com/spmf/>