MINING TOP-K CROSS-LEVEL HIGH UTILITY ITEMSETS

AIP490_G8

TEAM MEMBER

Nguyễn Đức Chính

HE150974

Nguyễn Tuấn Trường

HE150138

Nguyễn Khắc Tuệ

HE150066

OUTLINE

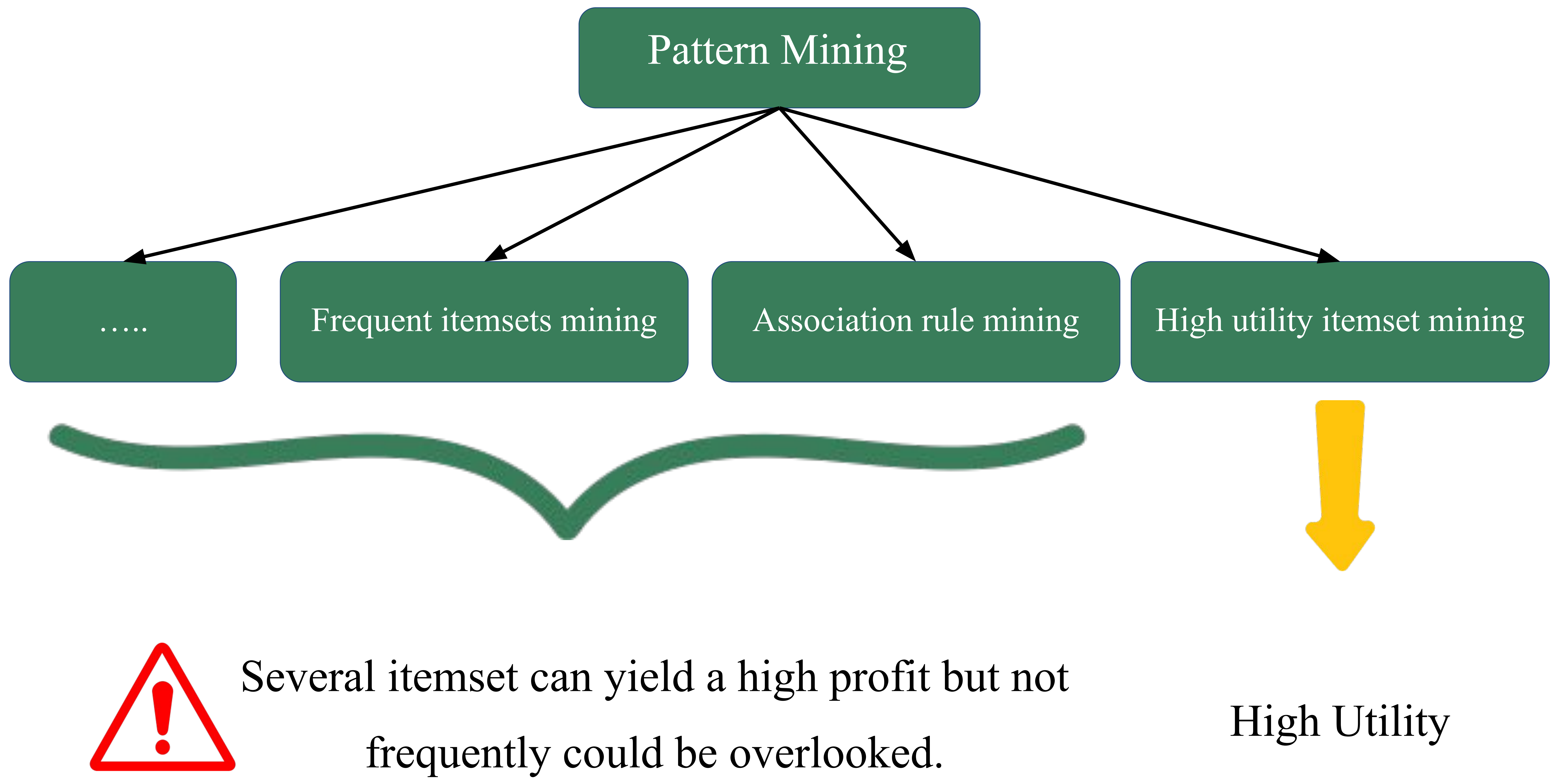
1. Problem and Motivation

Profit

- Profit is the ultimate goal in business.
- The way sales campaigns, advertising, and product displays are operated greatly impact revenue.

Need a way to support operations team to increase profits

2. Related Works

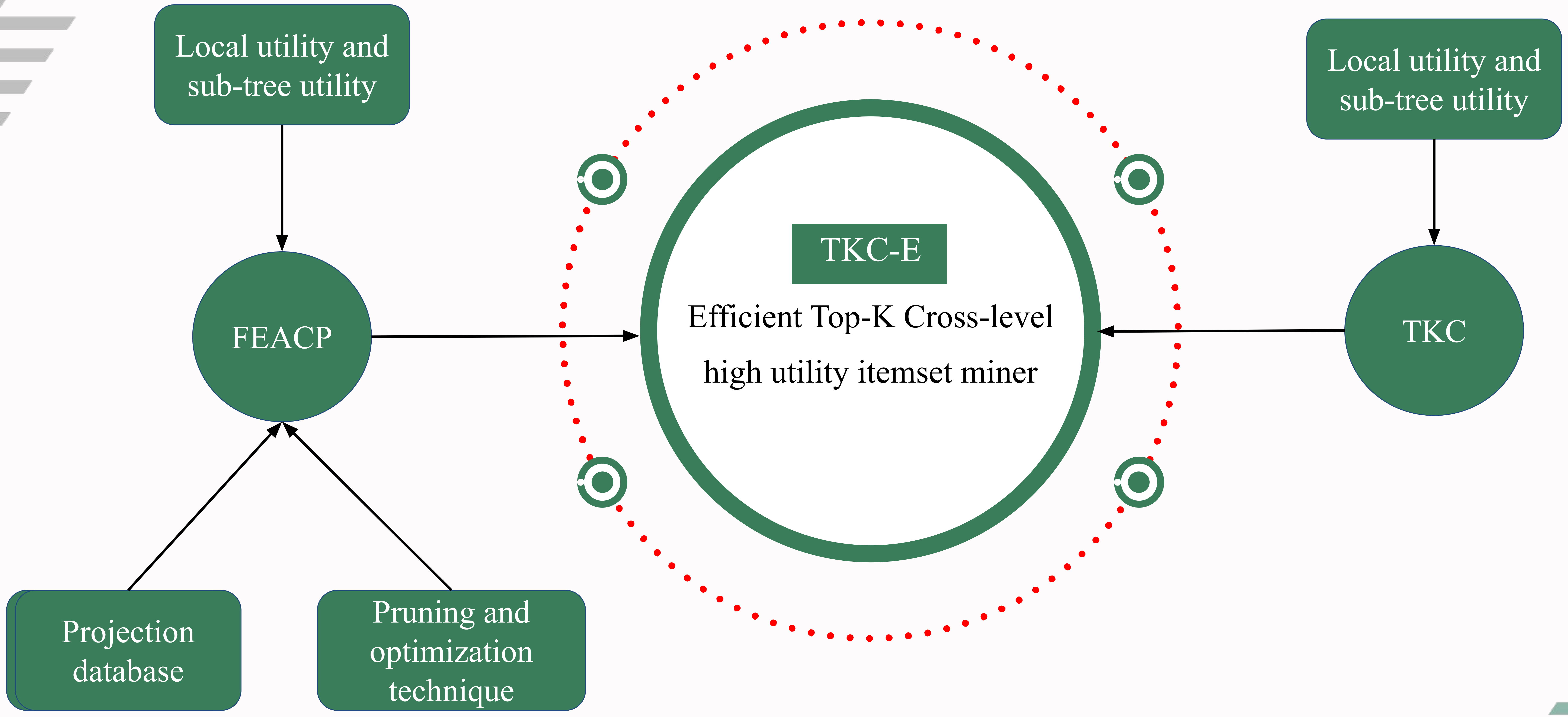


Pattern Mining
…..
Frequent itemsets mining
Association rule mining
High utility itemset mining
Several itemset can yield a high profit but not frequently could be overlooked.
High Utility

Summary table of algorithms

Author	Year of publication	Algorithm abbreviation	Algorithm name	Type
Tseng et al.	2010	UP-Growth	Utility Pattern-Growth	Mining High utility itemsets
Liu & Qu	2012	HUI-Miner	High Utility Itemset Miner	Mining High utility itemsets
Fournier-Viger et al.	2014	FHM	Frequent High Utility Itemset Mining	Mining High utility itemsets
Zida et al.	2016	EFIM	Efficient high-utility Itemset Mining	Mining High utility itemsets
Luca Cagliero et al.	2017	ML-HUI Miner	Multiple-Level High-Utility Itemset Miner	Mining Multiple-Level High utility itemsets
N.T. Tung et al.	2021	MLHMiner	Multiple-Level HMiner	Mining Multiple-Level High utility itemsets
Fournier-Viger et al.	2020	CLH-Miner	Cross-level high utility itemset mining	Mining Cross-Level High utility itemsets
N.T. Tung et al.	2021	FEACP	Fast and Efficient Algorithm for Cross-level high-utility Pattern mining	Mining Cross-Level High utility itemsets
Cheng-Wei Wu et al.	2012/2016	TKU	Top-K High Utility Itemset Miner	Mining Top-k HUIs
Vincent S. Tseng et al.	2016	TKO	Top-K High Utility Itemset Miner in One Phase	Mining Top-k HUIs
Mourad Nouioua et al.	2020	TKC	Top-K Cross-level high utility itemset miner	Mining Cross-level Top-k HUIs

3. Objective


Local utility and sub-tree utility
FEACP
Projection database
Pruning and optimization technique
TKC-E
Efficient Top-K Cross-level high utility itemset miner
Local utility and sub-tree utility
TKC

4. Methods

4.1. Problem definition

Table 1. A transaction database

TID	Transaction
T1	(a, 1),(c, 1),(d, 1)
T2	(a, 2),(c, 6),(e, 2),(g, 5)
T3	(a, 1),(b, 2),(c, 1),(d, 6),(e, 1),(f, 5)
T4	(b, 4),(c, 3),(d, 3),(e, 1)
T5	(b, 2),(c, 2),(e, 1),(g, 2)
T6	(a, 2),(c, 6),(e, 2)
T7	(c, 1),(d, 2),(e, 1)

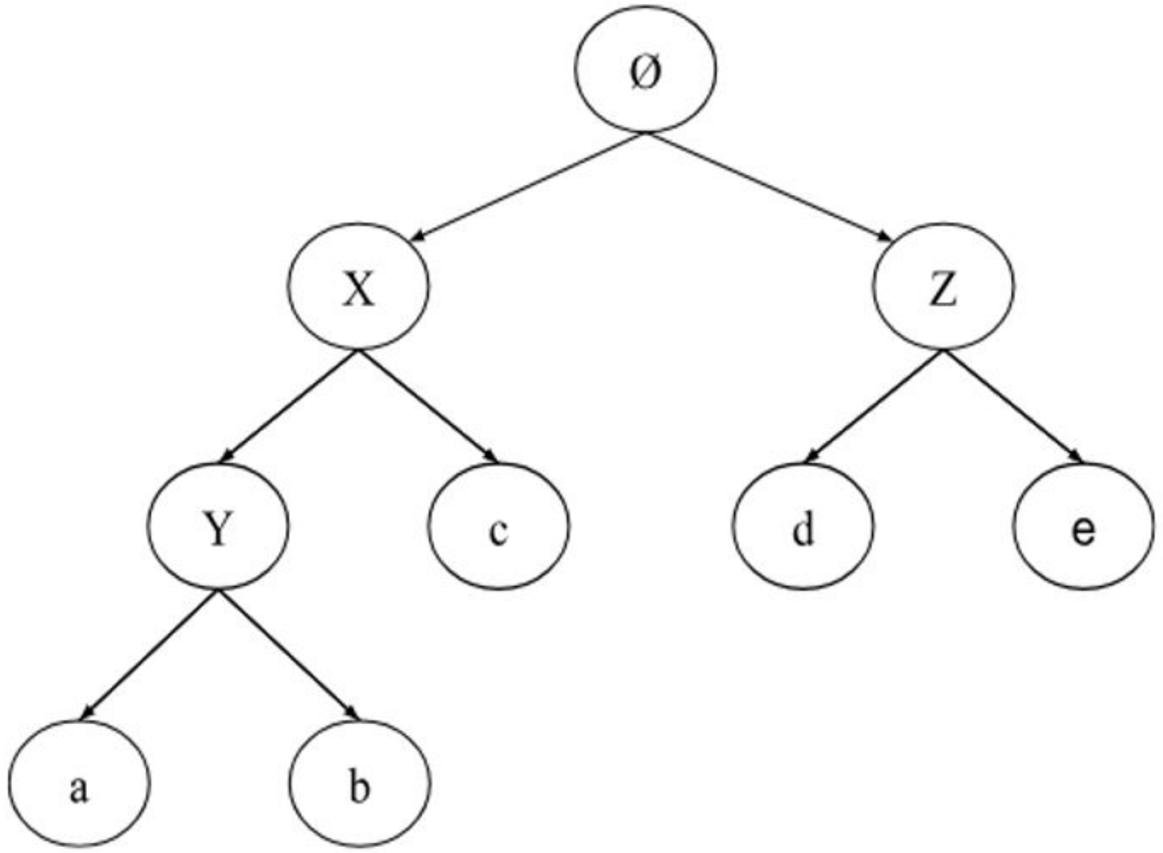


Fig. 1. A taxonomy of items.

Table 2. External utility values.

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

4.1. Problem definition

Utility of an item/itemsets:

$$u(i,T_c) = p(i) \times q(i, T_c)$$

$$u(P,T_c) = \sum_{i \in P} u(i,T_c).$$

$$u(P) = \sum_{T_c \in g(P)} u(P,T_c)$$

Example 1: The utility of a in is u(a,)= 1 × 5 = 5. The utility of {a, c} in is u({a, c}, T_2) = u(a, T_2) + u(c, T_2)= 2 × 5 + 6 × 1 = 16. The utility of {b, c} in the database D is u({b, c})= u({b, c}, T_3) + u({b, c}, T4) + u({b, c}, T_5)= 5 + 11 + 6= 22.

4.1. Problem definition

Utility of a generalized item/itemsets:

$$u(g, T_c) = \sum_{i \in Leaf(g,\tau)} p(i) \times q(i, T_c)$$

$$u(GP, T_c) = \sum_{d \in GP} u(d, T_c)$$

$$u(GP, T_{c)} = \sum_{d \in GP} u(d, T_c)$$

Example 2: In the taxonomy of Fig. 1, Z is a generalized item and $u(Z,T_4) = u(d, T_4) + u(e, T_4) = 3 \times 2 + 1 \times 3 = 9$. The utility of the generalized itemset {Z, b} in T_4 is $u(\{Z, b\}, T_4) = u(Z, T_4) + u(b, T_4) = (6 + 3) + 8 = 17$. The utility of the generalized itemset {Z, b} in the database is $u(\{Z, b\}) = u(\{Z, b\}, T_3) + u(\{Z, b\}, T_4) + u(\{Z, b\}, T_5) = 17 + 19 + 7 = 43$.

4.1. Problem definition

Transaction Weighted Utilization – TWU:

$$TU(T_c) = \sum_{i \in T_c} u(i, T_c)$$

$$TWU\ (P) = \sum_{T_c \in g(P)} TU(T_c)$$

$$TWU(GP) = \sum_{T_c \in g(i \in Leaf(GP,\tau))} TU(T_c)$$

Example 3: the TU values of transactions T_1 to T_7 for Table 2 are: 8, 27, 30, 20, 11, 22 and 8, respectively. TWU ({a}) = TU(T_1) + TU(T_2) + TU(T_3) + TU(T_6) = 8 + 27 + 30 + 22 = 87, TWU ({Y}) = TU(T_1) + TU(T_2) + TU(T_3) + TU(T_4) + TU(T_5) + TU(T_6) = 8 + 27 + 30 + 20 + 11 + 22 = 118.

4.1. Problem definition

Total order ($\succ$): Two distinct items a, b $\in$ AI are ordered as a $\succ$ b if level(a) < level(b), or if level(a) = level(b) $\wedge$ TWU(a) > TWU(b).

Extension:

$$E(P) = \{i \mid i \in AI \wedge i \succ w \;\; and \;\; \forall\, w \in P,\, i \notin Desc(w, \tau)\}$$

Example 4: If the total order is X $\succ$ Z $\succ$ c $\succ$ Y $\succ$ e $\succ$ d $\succ$ a $\succ$ b $\succ$ g $\succ$ f,

E(X) = {Z, c, Y, e, d, a, b, g, f}, E{c,e}= {d, a, b, g, f}.

4.1. Problem definition

Remaining utility:

$$re(P, T_c) = \sum_{i \in T_c \wedge i \in E(P)} u(i, T_c)$$

Example 5: Consider the running example, if the total order is X > Z > c > Y > e > d > a > b > g > f, re(X, T_2)= 6 +5 = 11, re({c, e}, T_4) = 8+6 = 14.

Local utility:

$$lu\ (P, i) = \sum_{T_c \in g(P \cup \{i\})} [u(P, T_c) + re(P, T_c)]$$

$$lu\ (P, z) = \sum_{T_c \in g(P \cup j \in Leaf(z, \tau))} [u(P, T_c) + re(P, T_c)]$$

Example 6: Consider the running example and P = {c}. We have that lu(P, a) = 8 + 27 + 30+22 = 87, lu(P, d) =8 + 30 + 20 + 8= 66 and lu(P, e) =115.

4.1. Problem definition

Sub-tree utility:

$$su\ (P,i) = \sum\nolimits_{T_c \in g(P \cup \{i\})} [u(P,T_c) + u(i,T_c) + \sum\nolimits_{j \in T_c \wedge j \in E(P \cup \{i\})} u(j,T_c)]$$

$$su\ (P,z) = \sum\nolimits_{T_c \in g(P \cup j \in Leaf(z,\tau))} [u(P,T_c) + \sum\nolimits_{j \in Leaf(z,\tau)} u(j,T_c) + \sum\nolimits_{j \in T_c \wedge j \in E(P \cup \{z\})} u(j,T_c)]$$

Example 7: Consider the running example and P = {c}. We have that su(P, a) = 8 + 21 + 27 + 16 = 72, su(P, d) = 6 + 22 + 17 + 5 = 50 and su(P, e) = 115.

4.1. Problem definition

Primary and secondary items:

$$Primary(P) = \{z \mid z \in E(P) \wedge su(P, z) \geq \mu\}.$$

$$Secondary(P) = \{z \mid z \in E(P) \wedge lu(p, z) \geq \mu\}$$

Projected database:

$$(T_c)_P = \{k \mid k \in T \wedge k \in E(P)\}$$

$$D_P = \{(T_c)_P \mid T_c \in D \wedge (T_c)_P \neq \emptyset)\}$$

Example 11: for the database *D* given in Table 2 and P ={d}, the database D_P can be constructed by the following transactions: $(T_1)_P$={a}, $(T_3)_P$={a,b,e,f}, $(T_4)_P$={b}.

4.2. TKC-E Algorithm

Algorithm 1: The TKC-E algorithm

input: *D*: a transaction database, τ: a taxonomy, *k*: the number of patterns to be found.
output: the top-k cross-level HUIs.

1. Initializes $\mu = 0$, $P = \{\emptyset\}$ a priority queue *Q* with the top-k cross-level HUIs from AI;
2. Read τ and *D* and use a utility-bin array to calculate to compute $lu(P, z)$ of each (generalized) item $z \in AI$;
3. Secondary(*P*) = $\{z \mid z \in AI \wedge lu(P, z) \geq \mu\}$;
4. Compute $\prec$, the total order on items from Level and *TWU* values on Secondary(*P*);
5. Scan D to store each generalized item g $\in$ Secondary(*P*) in each transaction, discard every item i $\notin$ Secondary(*P*) from transactions,sort items in each transaction, delete empty transactions, and then build and store the utility-list of each generalized item;
6. Compute the sub-tree utility $su(P, z)$ of each item $z \in$ Secondary(*P*);
7. Primary(*P*) = $\{z \mid z \in AI \wedge su(P, z) \geq \mu\}$;
8. SEARCH (P, D, $Primary(P)$, $Secondary(P)$, k , μ ,Q);

4.2. TKC-E Algorithm

Algorithm 2: The SEARCH procedure

input: P: itemset, D_P: P-projected database, Primary(P): primary items of P, Secondary(P): secondary items of P, k: the number of patterns to find, μ: the
internal threshold, Q: the top-k patterns until now.
output: Q is updated with top-k CLHUIs that are transitive extensions of P.
FOR EACH item $z \in$ Primary(P) DO:

1. $N = P \cup \{z\}$, Secondary(P)' = $\{x \in$ Secondary(P) | $x \notin$ Desc(z, τ)$\}$;
2. Scan D_P to determine u(N), construct D_N, remove every item $\in$ Desc(z, τ) and remove empty transactions;
3. IF u(N) > μ THEN Insert z into Q;
4. IF Size of Q > k THEN:
   Raises to the k-th largest utility value in Q;
   Remove from Q all patterns with utility less than μ;
5. Scan D_N to compute su(N,w), lu(N,w) for every item $w \in$ Secondary(P)';
6. Primary(N) = $\{x \in$ Secondary(P)' $\wedge$ su(N, z) $\geq \mu\}$;
7. Secondary(N) = $\{x \in$ Secondary(P)' $\wedge$ lu(N, z) $\geq \mu\}$;
8. SEARCH (N, D_N, Primary(N) , Secondary(N) , k, μ , Q);

END

5. Experiment and Results

5.1. Experiments

- Configuration: Intel Core-i7 processor clocked at 4.5GHz, 16 GB of RAM and running on the Windows 11 operating system.

- Java programming language with version JDK 11.

- Evaluation parameter: Runtime and Memory usage.

- The Algorithm execute 5 times to get the average value.

5.2. Data

- Source: https://www.philippe-fournier viger.com/spmf/index.php?link=datasets.php.

- Description: Real-life customer transaction datasets with actual utility values.

- Suitability: It has been used in many scientific papers.

Raw data

- **Dataset file**:

Format: Item1 item2 item3… : TU : Util(item1) Util(item2) Util(item3)

- **Example**: Fruithut database

2010 2021 2032 : 897 : 199 399 299

2038 : 180 : 180

1031 2022 : 449 : 150 299

Raw data

- **Taxonomy file**:

Format: Item, category item belongs to

- **Example**: Fruithut database

1001,110	159,150
1002,150	110,100
1003,150	120,100
1004,150	130,100
1005,130	140,100
1007,120	150,100

Analyze data

Database	|D|	|I|	|GI|	MaxLevel	$\|T_{MAX}\|$	$\|T_{AVG}\|$	Density
Fruithut	181.970	1.265	43	4	36	3.58	Sparse

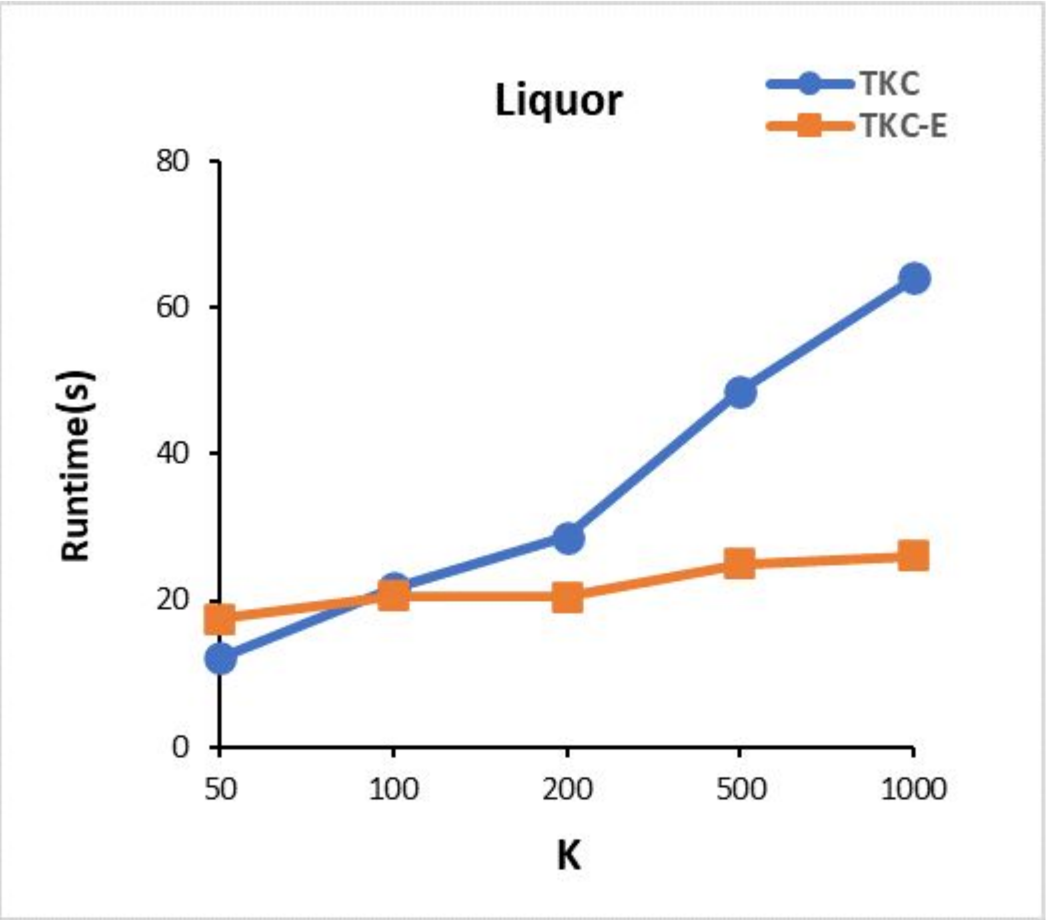
- |D|: transaction count of D
- |I|: number of distinct items
- |GI|:generalized item count
- MaxLevel: maximum level in each database
- |T max|: maximum transaction length
- |T avg|: average transaction length
- Density: density of the databases

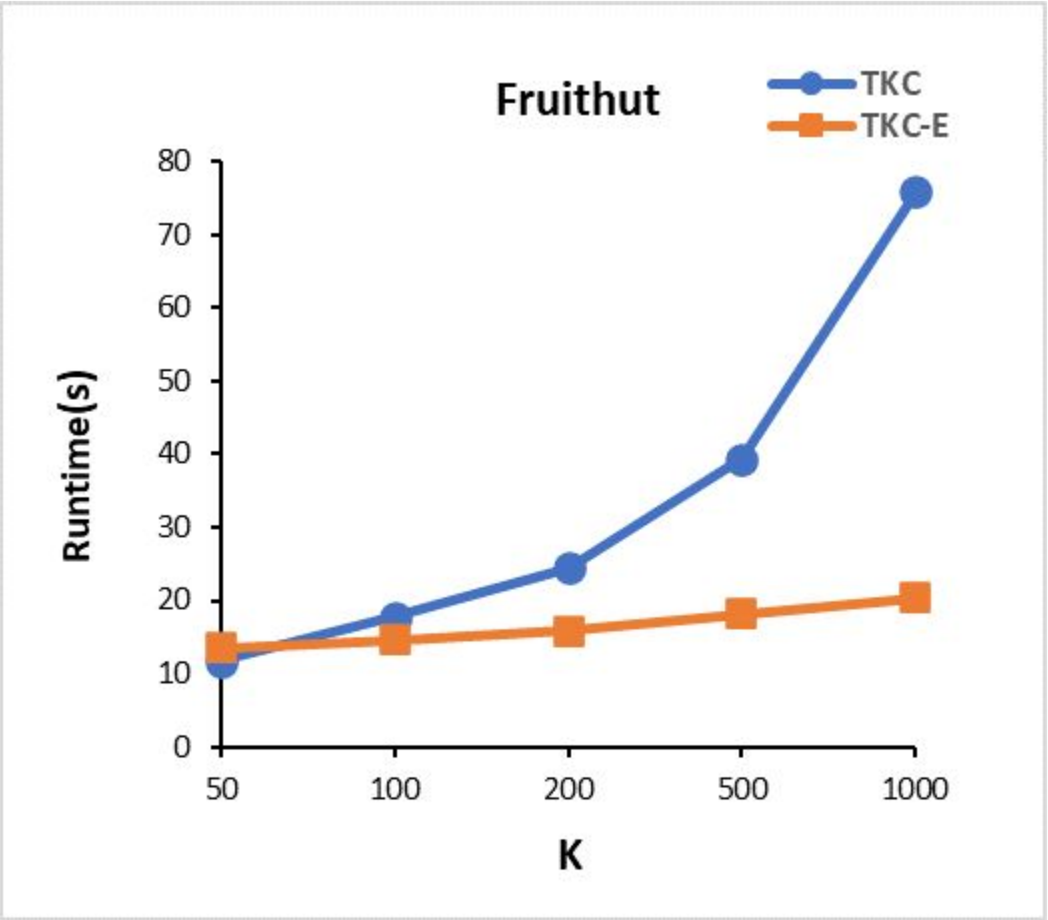
Analyze data

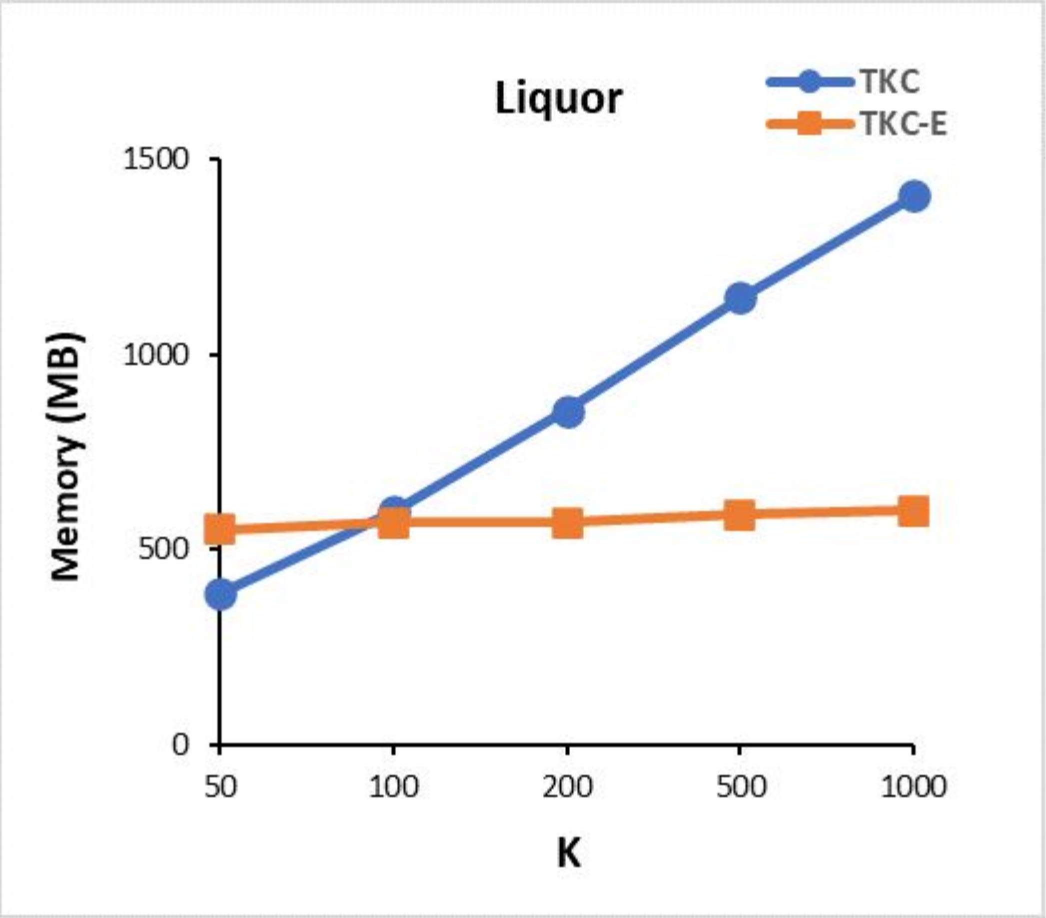
Database	**|D|**	**|I|**	**|GI|**	**MaxLevel**	**$\|T_{MAX}\|$**	**$\|T_{AVG}\|$**	**Density**
Liquor	9284	4026	78	7	11	7.87	Sparse
Fruithut	181.970	1.265	43	4	36	3.58	Sparse
Chess	3.196	75	30	3	37	37.00	Dense
Accident	10.000	468	216	6	51	33.80	Dense

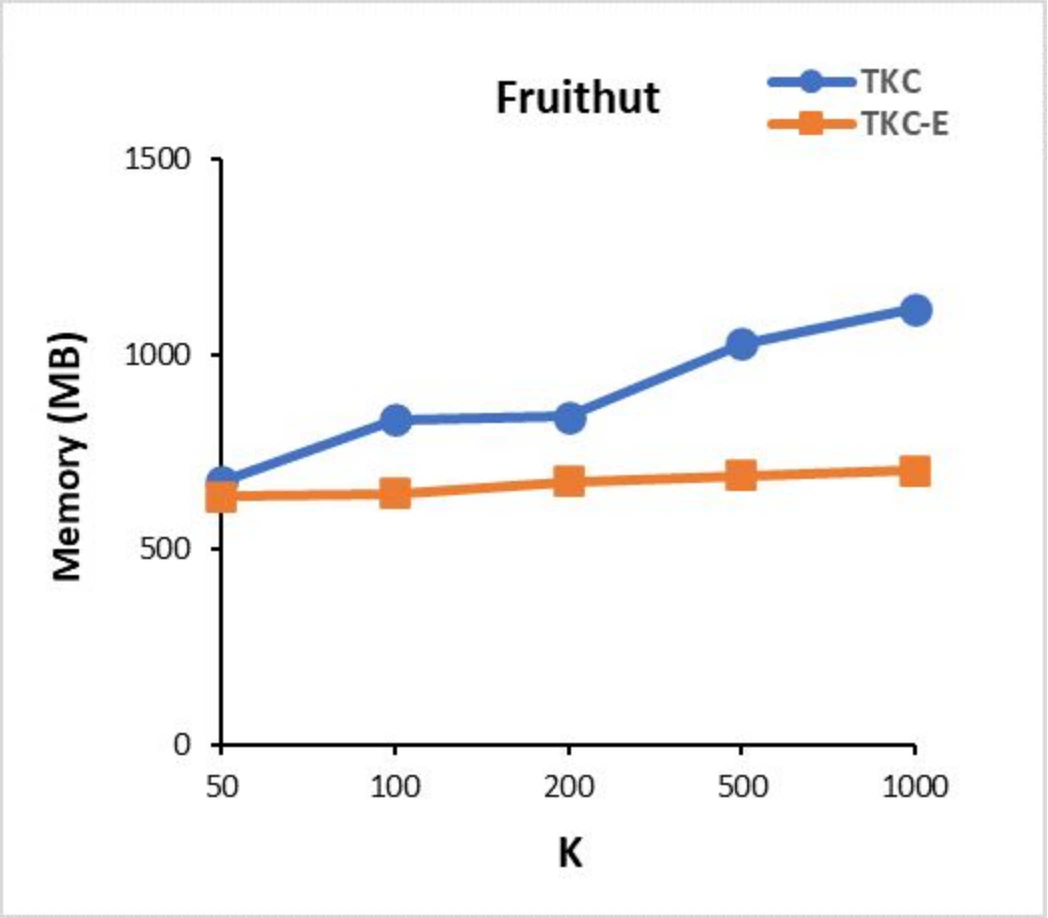
Database characteristics

Sparse Database

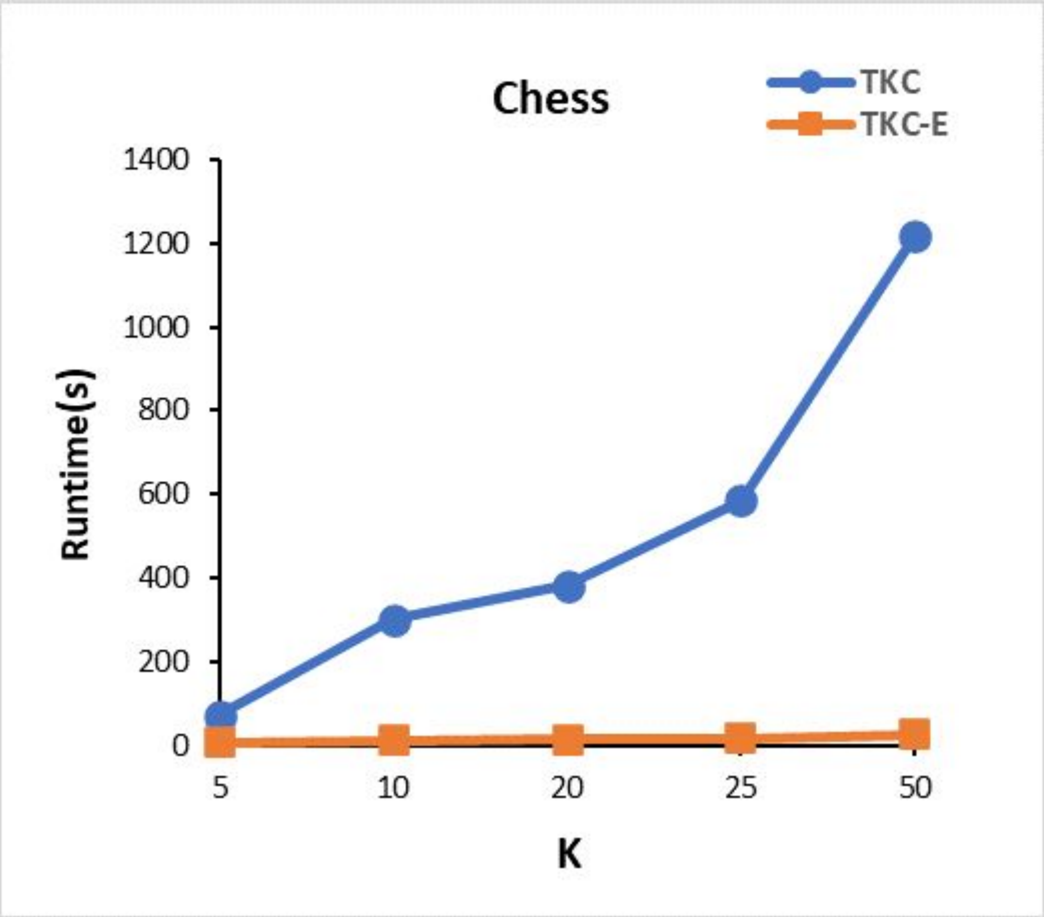


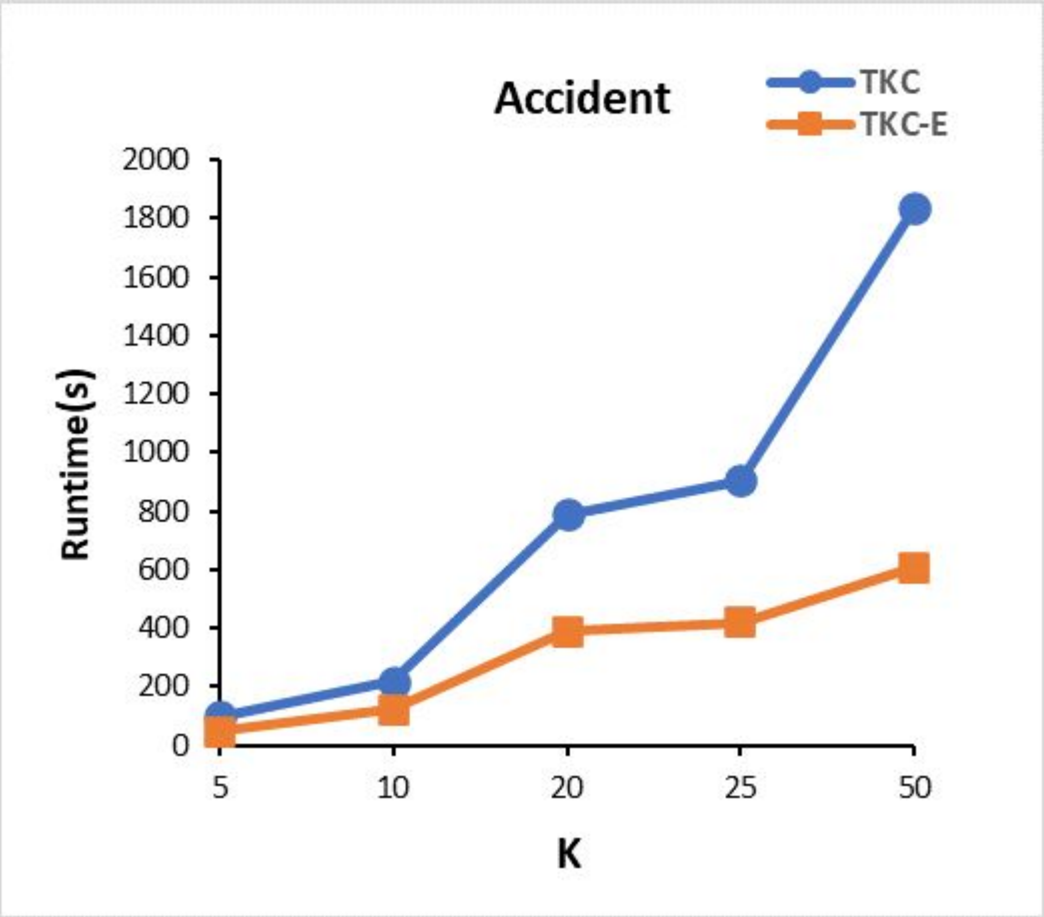


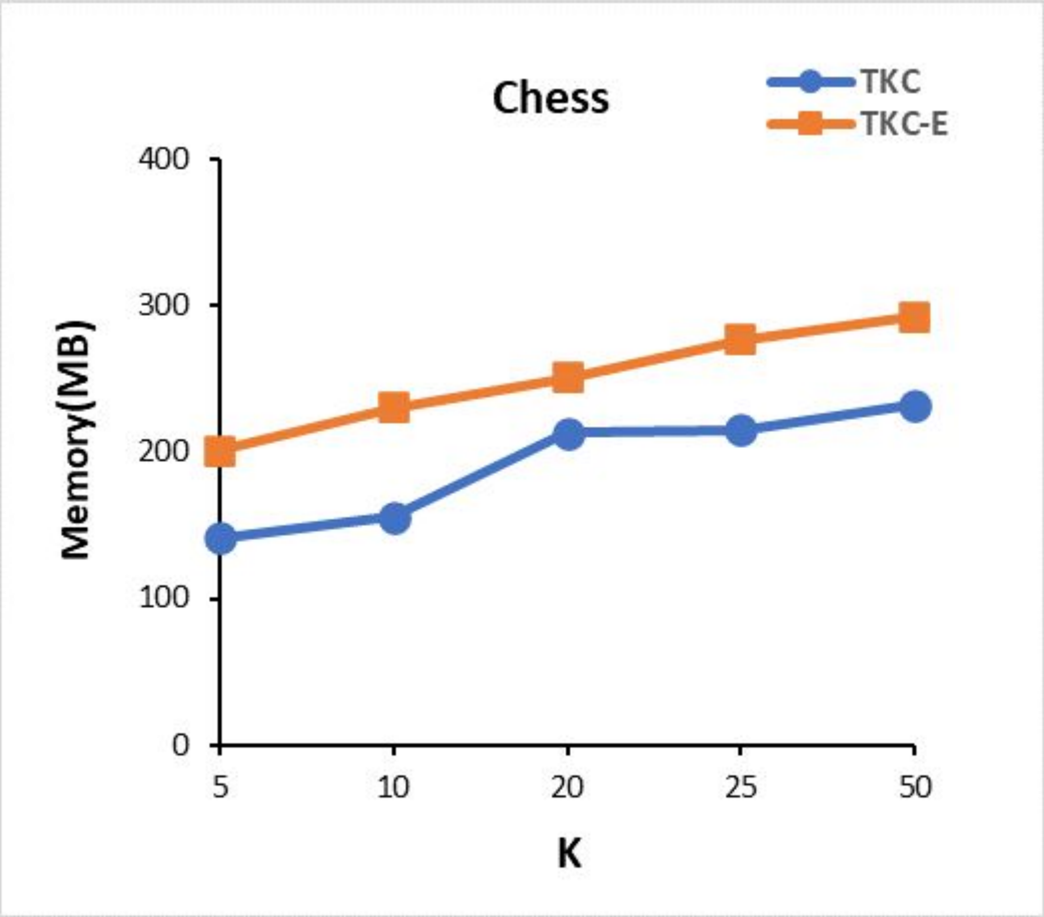


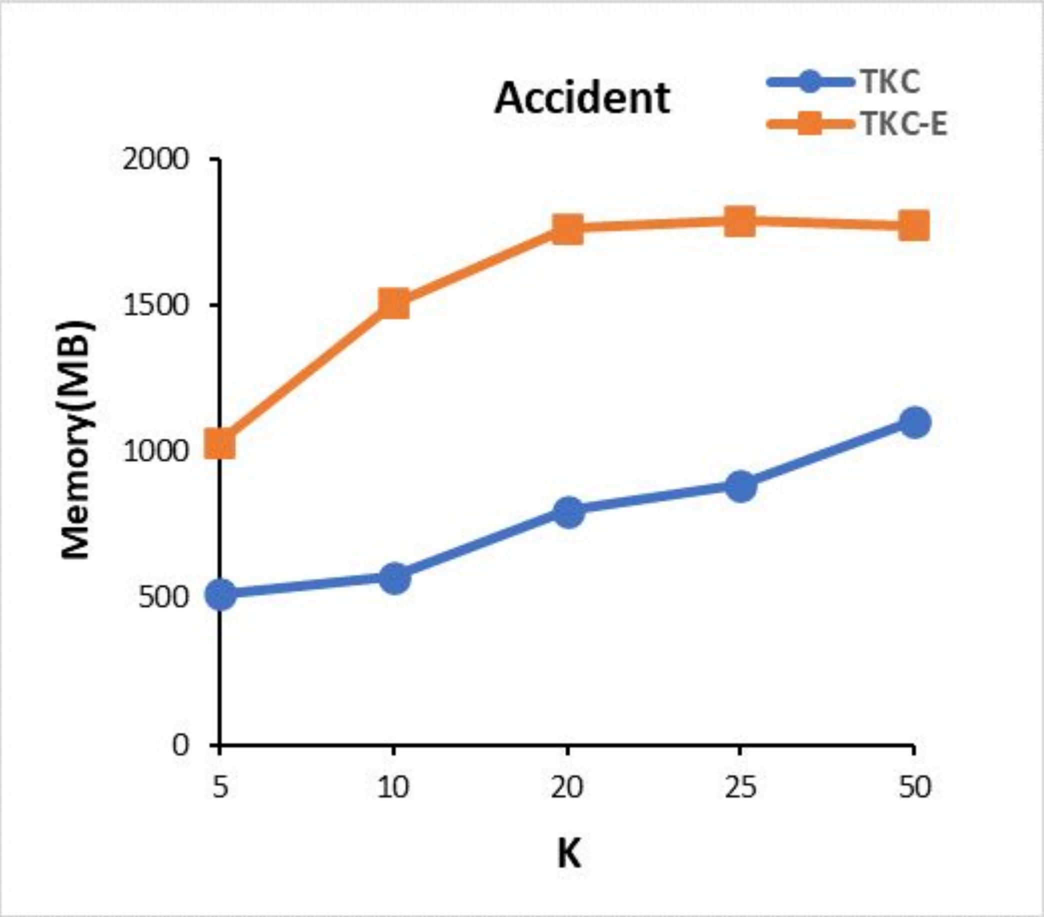


Dense Database









6.Future work

- Improve the memory usage of TKC-E for both sparse and dense datasets.
- Apply efficient pruning strategies
- Study parallel computing frameworks to reduce mining time, as well as enable computation with larger databases.

THANK YOU!