AUGUST - 2023

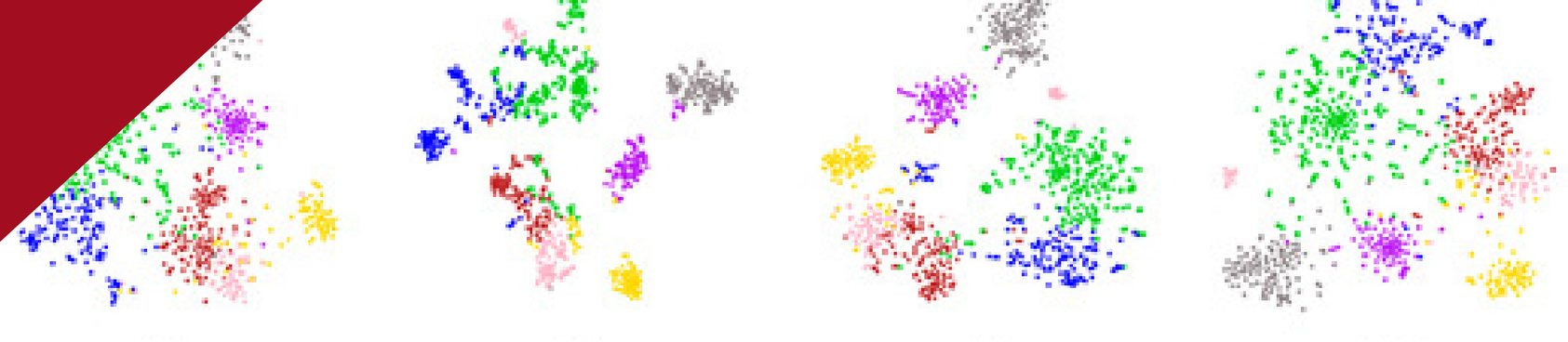
UNIVERSITY OF SCIENCE - VNU

FINAL PRESENTATION

SUBJECT: GRAPH DATA MINING - CSC17103

GROUP 04

CONTENT



Page	Content
02	Our Team
03	Problem outlined
06	Meaning of the topic
10	Methods to solve
35	Experiments
44	Conclusion

OUR TEAM

Huỳnh Thiện Nhân

18127164

Võ Văn Hoàng

20127028

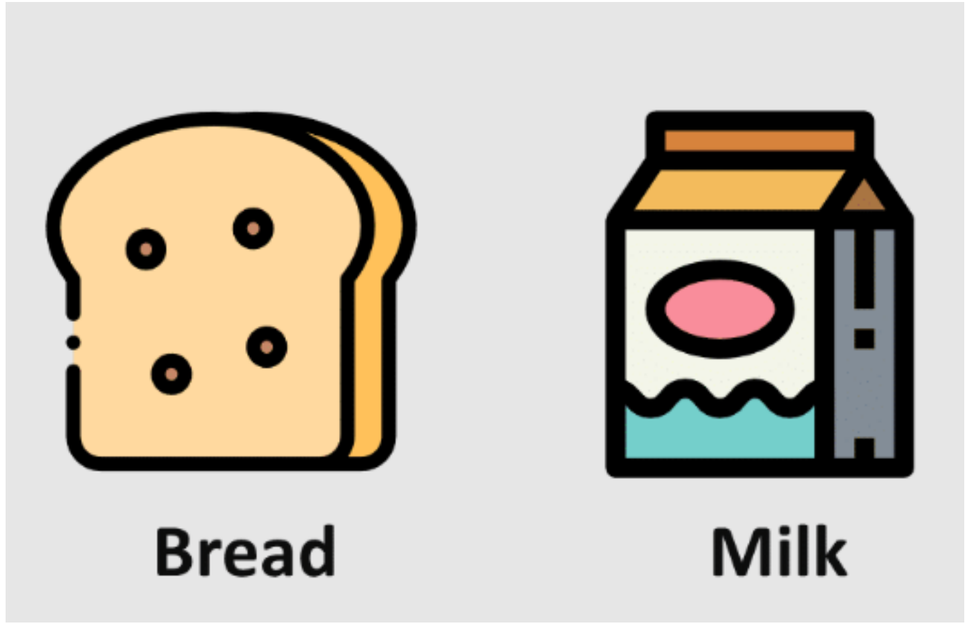
Ngô Văn Trung Nguyên

20127054

Nguyễn Trương Minh Khôi

20127214

PROBLEM OUTLINED



Quantity:

Better in Bread and Milk

Revenue

Better in Computer and monitor

- **Frequent pattern mining methods have a major drawback: they always assume the frequency is the only parameter of interest to the user.**
- **Frequency alone may not always be able to capture the users's interest.**

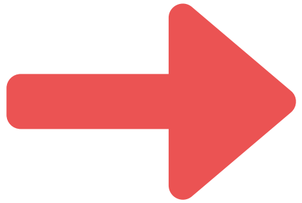
Mining High Utility Itemset in transactional and sequential databases

Internal utility: Quantity of the item

A part of utility of an item

External utility: Quality of the item

A part of utility of an item



- **Internal utility:** the number of items bought.
- **External utility:** revenue generated from an item.

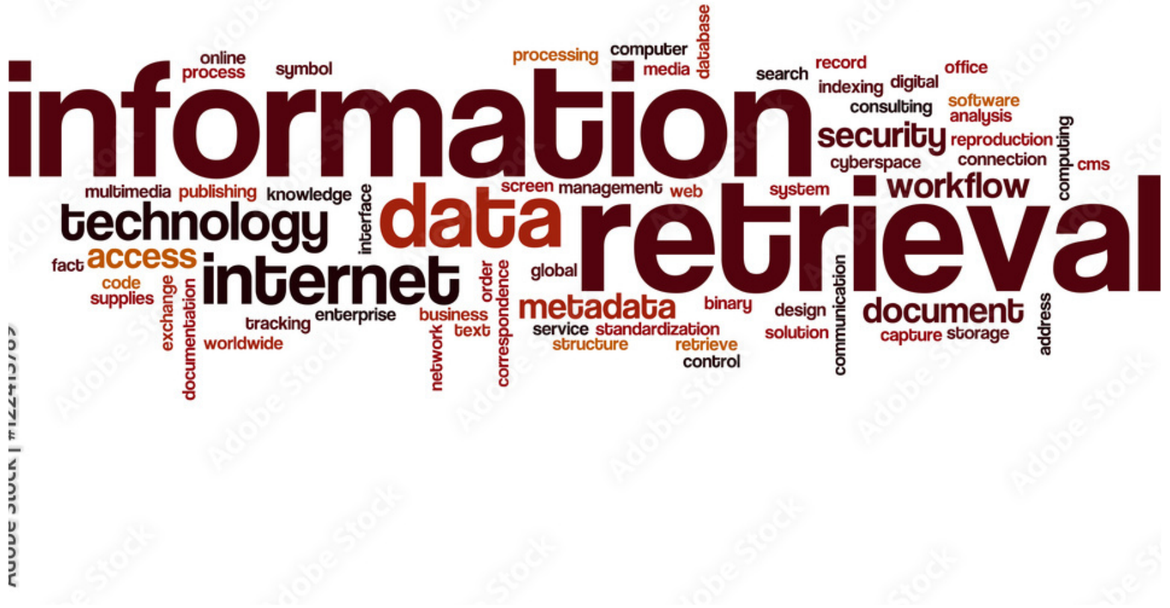
High utility itemset mining discovers itemsets that generate more revenue.

No COMPLETE framework proposed for utility-based subgraph mining.

MEANING OF THE TOPIC

Some practical applications

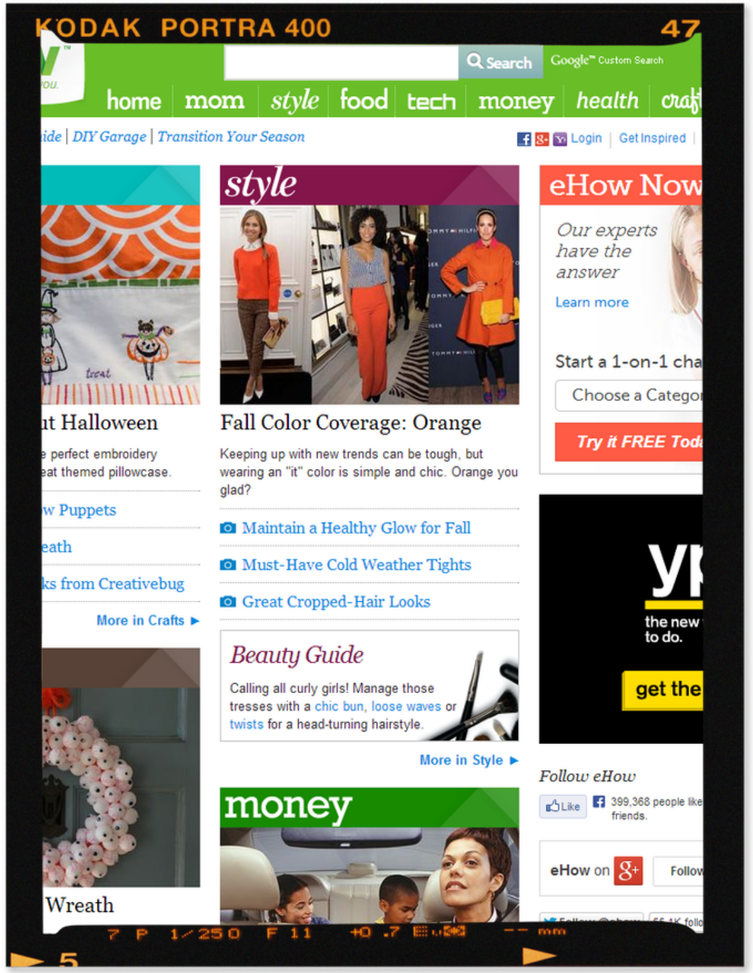
- Documents are displayed as graphs to maintain the complicated relationships between words. Each word in a document is represented by a **node** in the associated graph, and an **edge** between two nodes is added if the matching phrases appear in the same window of a certain size.
- **External utility** can be assigned to each edge by the terms' importance (tf-idf, node centrality) in the document
- **Internal utility** is the number of common occurrences of the terms within the same window.
- Mining high utility subgraphs from such a database will find complex associations between words **preserving their importance in the documents**, which can be used for tasks like classification, clustering,...



Information retrieval of utility-based graph mining

Some practical applications

- In this case, a graph that has web pages as nodes may be used to **describe a user's actions**. An edge connecting two nodes shows how one website may link to another website through an advertising. The kind of advertisement can be used to **label** the nodes and edges.
- The **internal utility** here is referral counts.
- The **external utility** that measures advertising income for that edge's particular kind.
- For this application, high utility subgraphs represent the portions of the advertisement network from which higher revenue is generated.



Context of online web page advertisements

METHOD TO SOLVE

- A complete framework suitable for utility-based sub graph mining.
- A complete algorithm - called UGMINE, for mining high utility subgraphs from transactional labeled graph databases.
- A tighter pruning technique - named RMU-Pruning - to eliminate false candidates from the search space and improve runtime performance

DEFINITIONS

Term	Definition
$u_e(e)$	The utility of an edge e
$u_g(g)$	The utility of a quantitative labeled subgraph g
$u_G(g', G)$	The utility of a labeled subgraph g' in a quantitative labeled graph G
$u_D(g', D)$	The utility of a labeled subgraph g' in a quantitative labeled graph Database D
ϕ(g', G)	Subgraph isomorphisms from labeled subgraph g' to quantitative labeled graph G
δ	User defined utility threshold

Proposed utility-based graph mining framework

$$u_e(e) = p(e) \times q(l(u), l(v), l(e))$$

$$u_g(g) = \sum_{e \subset E_g} u_e(e)$$

$$(u, v) \in E'_g \iff (\phi(u), \phi(v)) \in E_g;$$
$$\forall u \in V'_g,\ l(u) = l(\phi(u)); \text{ and}$$
$$\forall (u, v) \in E'_g,\ l(u, v) = l(\phi(u), \phi(v)).$$

$$u_G(g', G) = \max_{g \in \phi(g', G)} u_g(g)$$

$$u_D(g', D) = \sum_{G \in D} u_G(g', G)$$

$$minutil = \sum_{G \in D} u_g(G) \times \delta$$

EXAMPLE

- A quantitative labeled graph database D containing two quantitative labeled graphs G_1 and G_2. Each vertex in the graphs is assigned with a label and an identification number. Each edge is assigned with a label and a number representing internal utility.

EXAMPLE

l(u), l(v), l(e)	Giá trị tiện ích bên ngoài (external unitility)
a, a, p	4
a, a, q	1
a, b, p	2
a, b, q	3

- In graph G_1, the internal utility of edge (0,1) is 3. The edge label is q, and the vertices labels are a and b.
- From table above, we find that external utility of such an edge is 3.

  → The utility of edge (0, 1) in graph G_1 is: 3 x 3 = 9

  → The utility of edge (0, 2) in graph G_1 is: 4 x 4 = 16

  → The utility of edge (1, 2) in graph G_1 is: 5 x 2 = 10

Therefore, the utility of graph G_1 is:

$\mathbf{u_e(0, 1) + u_e(0, 2) + u_e(1, 2)} = 9 + 16 + 10 = \mathbf{35}$

EXAMPLE

- Similarly, with graph G_2:

→ The utility of edge (1, 0) in graph G_2 is: 2 x 4 = 8

→ The utility of edge (1, 2) in graph G_2 is: 2 x 3 = 6

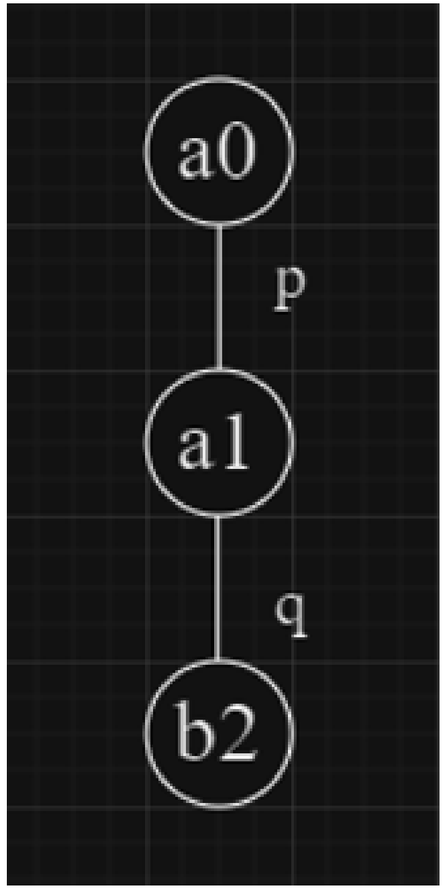
→ The utility of edge (1, 3) in graph G_2 is: 2 x 1 = 2

Therefore, the utility of graph G_2 is: $\mathbf{u_e(1, 0)} + \mathbf{u_e(1, 2)} + \mathbf{u_e(1, 3)} = 8 + 6 + 2 = \mathbf{16}$

So, The utility of the database D is $\mathbf{u_g(G1)} + \mathbf{u_g(G2)} = \mathbf{35 + 16 = 51}$

EXAMPLE

- If the minimum utility threshold $\boldsymbol{\delta = 0.35}$ then minutil = 51 x 0.35 = 17.85



- Let the labeled subgraph presented besides be g'.

✓ A subgraph isomorphism $\boldsymbol{\phi 1}$ from g' to a subgraphg $\in$ G_1 holds where: $\boldsymbol{\phi 1(0) = 2,\ \phi 1(1) = 0,\ \phi 1(2) = 1}$

- $\boldsymbol{u_G(g', G1) = u_g(g) = u_e(0, 1) + u_e(0, 2)} = 3\times3 + 4\times4 = \boldsymbol{25}$

✓ Similarly

- $\boldsymbol{u_G(g', G2) = u_g(g) = u_e(1, 0) + u_e(1, 2)} = 2\times4 + 2\times3 = \boldsymbol{14}$

- So, $\boldsymbol{u_D(g', D) = u_G(g', G1) + u_G(g', G2)} = 25 + 14 = \boldsymbol{39}$

As $u_D(g', D) = 39 > \text{minutil} = 17.85$, g' is a high utility subgraph in Database D.

Proposed algorithm

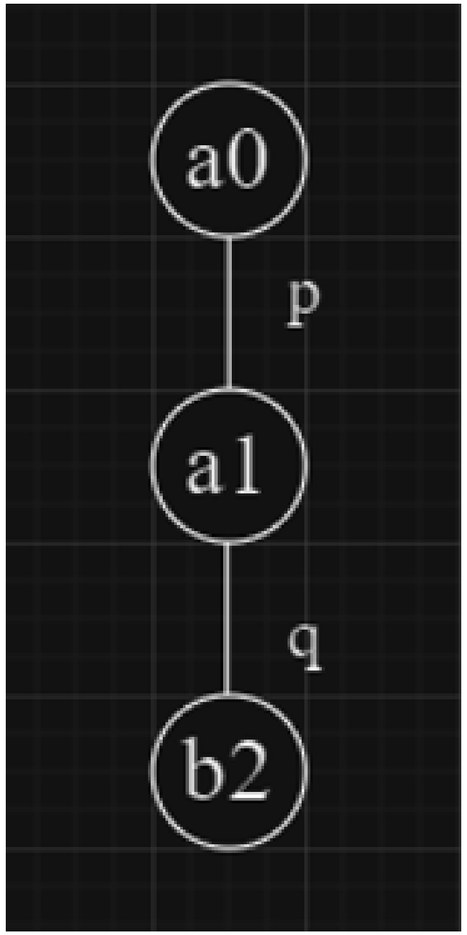
The graph weighted utility (GWU) of a labeled subgraph g' in a quantitative labeled graph G

$$GWU(g', G) = \begin{cases} u_g(G) & \text{if } \exists g \in \phi(g', G) \\ 0 & \text{otherwise.} \end{cases}$$

The graph weighted utility (GWU) of a labeled subgraph g' in a quantitative labeled graph database D

$$GWU(g', D) = \sum_{G \in D} GWU(g', G)$$

Proposed algorithm



For example, the GWU value of the labeled subgraph g' in the database D is: $GWU(g', D) = 35 + 16 = 51$

For all canonical codes in DFS code tree, UGMINE calculates the utility of the labeled subgraph associated with the code as defined:

$$u_D(g', D) = \sum_{G \in D} u_G(g', G)$$

If the utility value is greater than or equal to minut il, then the labeled subgraph is a high utility subgraph. In UGMINE, the authors prune codes in the DFS code tree that are associated with non-high GWU subgraphs.

RMU Pruning

- Establish a tighter upper bound on the utility of a candidate subgraph and all of the supergraphs extended from it.
- An edge adjacent to a vertex—which is not on the rightmost path of a candidate subgraph—will not be added in any of its extensions.

RMU Pruning

Definition 1: Let $g' = (V_g', E_g', l_g')$ be a candidate labeled subgraph in a DFS code tree with the rightmost path R_g'. Let G be a quantitative labeled graph and ϕ be an isomorphism from g to g'. Define

- $LM(g', G, g) = \{(u, v) : (u, v) \in E_G, \ \phi^{-1}(u, v) \notin E_G', \ \phi^{-1}(u) \in V_g' - R_g' \ or \ \phi^{-1}(v) \in V_g' - R_g'\}$.

Definition 2: $$RMU(g', G) = \max_{g \in \phi(g', G)} \{u_g(G) - \sum_{e \in LM(g', G, g)} u_e(e)\}$$

Definition 3: $$RMU(g', D) = \sum_{G \in D} RMU(g', G)$$

Definition 4:

Given a quantitative labeled graph database and a minimum utility value minutil, a labeled subgraph g' is a high RMU subgraph if and only if RMU(g', D) $\geqslant$ minutil.

RMU Pruning

Theorem 1: The high RMU subgraph is antimonotonic.

- Given any two labeled graphs g1’, g2’ in a DFS tree and any quantitative labeled graph database D such that g1’ is a subgraph of g2'.
- We can prove that, if RMU(g1', D) < minutil, then RMU(g2', D) < minutil.

RMU Pruning

Theorem 2: The UGMINE-RMU algorithm is complete.
A "complete algorithm" is an algorithm that is guaranteed to produce a correct and valid result for a given problem or task. It means that the algorithm will always terminate and provide a solution if one exists.

- If g' is a non-high RMU subgraph, then

$$u_D(g', D) \leqslant RMU(g', D) < minutil$$

- In other words, a non-high RMU subgraph cannot be a high utility subgraph

RMU Pruning

Algorithm 1 UGMINE.

```
1:  procedure UGMINE(C, D, minutil)    ▷ Initially C = ∅
2:      E ← RightMostPathExtensions(C, D)
3:      for e ∈ E do
4:          C′ ← C ∪ e
5:          if IsCanonical(C′) is False then
6:              Continue
7:          g = Graph(C') \\Graph associated with code C'
8:          if u_D(g, D) ≥ minutil then
9:              g is a high utility subgraph
10:         if RMU(g, D) ≥ minutil then ▷ GWU(g, D)
    in case of GWU pruning
11:             UGMINE(C′, D, minutil)
```

RMU Pruning

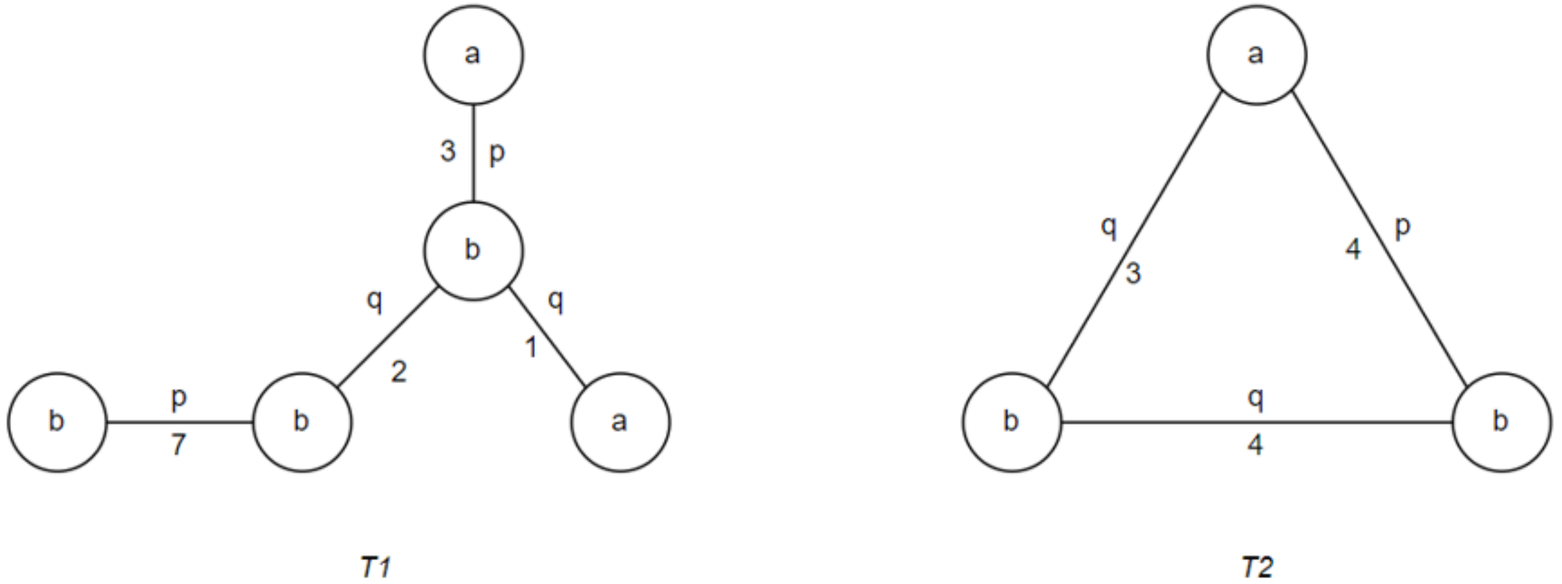
Algorithm 2 Canonality checking for a DFS code.

```
1:  procedure IsCanonical(C)
2:      g' ← Graph(C)
3:      C' ← ∅
4:      for t ∈ C do
5:          m = min(RightMostPathExtensions(C, g'))
6:          if m < t then
7:              return False
8:          C' ← C' ∪ t
9:      return True
```

Algorithm 3 Right most path extensions.

```
1:  procedure RightMostPathExtensions(C, D)
2:      E = ∅
3:      for G ∈ D do
4:          if C = ∅ then
5:              for (u, v) ∈ G do
6:                  E ← E ∪ (0, 1, l(u), l(v), l(u, v))
7:          else
8:              R ← Vertices on right most path in C
9:              u_r ← Right most vertex in C
10:             g' ← Graph(C)
11:             for φ ∈ SubgraphIsomorphisms(C, G) do
12:                 for v ∈ R do
13:                     if (u_r, v) ∉ g' and (u_r, v) ∈ G then
14:                         E ← E ∪ (u_r, v, l(u_r), l(v), l(u_r, v))
15:                 for u ∈ R do
16:                     if v ∉ g' and (u, v) ∈ G then
17:                         E ← E ∪ (u, u_r+1, l(u), l(v), l(u, v))
18:     return Set of all distinct tuples in E
```

SIMULATION



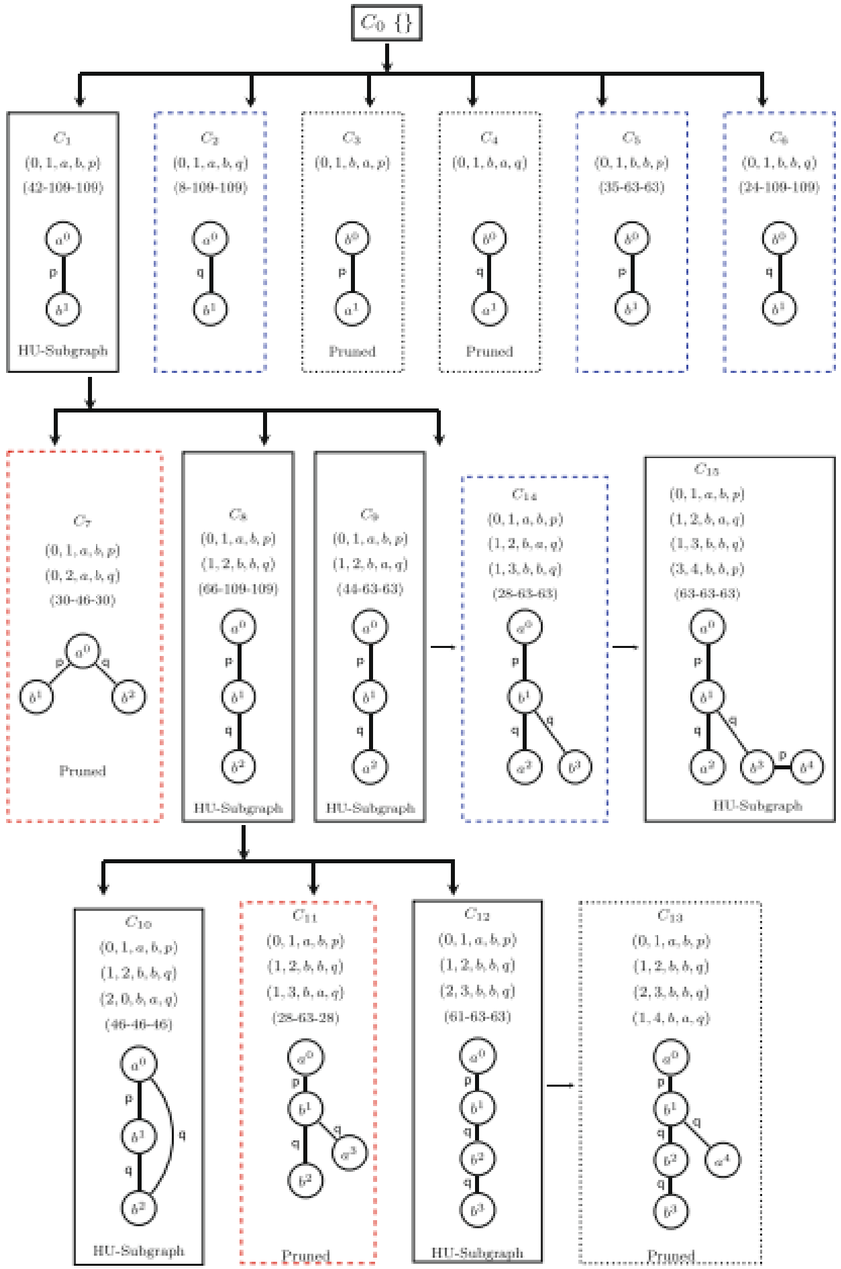
Hình a: *một cơ sở dữ liệu đồ thị được gán nhãn định lượng D*

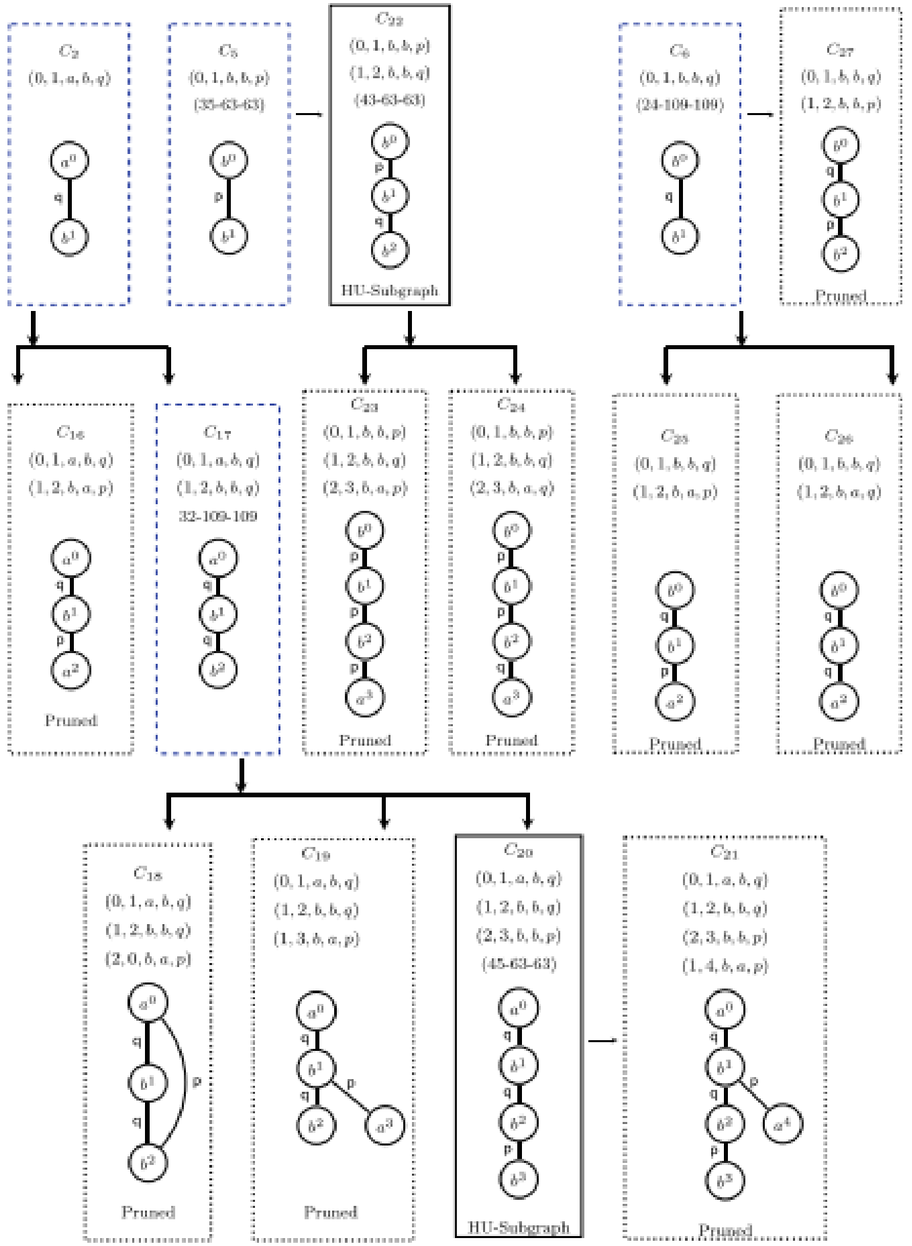
Nhãn điểm 1	Nhãn điểm 2	Nhãn cạnh	Độ tiện dụng ngoài
a	b	p	6
a	b	q	2
b	b	p	5
b	b	q	4

Bảng độ tiện dụng ngoài

- $U(T_1) = U((a - b - p)) + U((b - b - q) + U((a - b - q)) + U((b - b - p)) = (3 \times 6) + (2 \times 4) + (1 \times 2) + (7 \times 5) = \mathbf{63}$
- $U(T_2) = U((a - b - p)) + U((b - b - q) + U((a - b - q) = (4 \times 6) + (4 \times 4) + (3 \times 2) = \mathbf{46}$
- $U(D) = U(T_1) + U(T_2) = 63 + 46 = \mathbf{109}$

SIMULATION

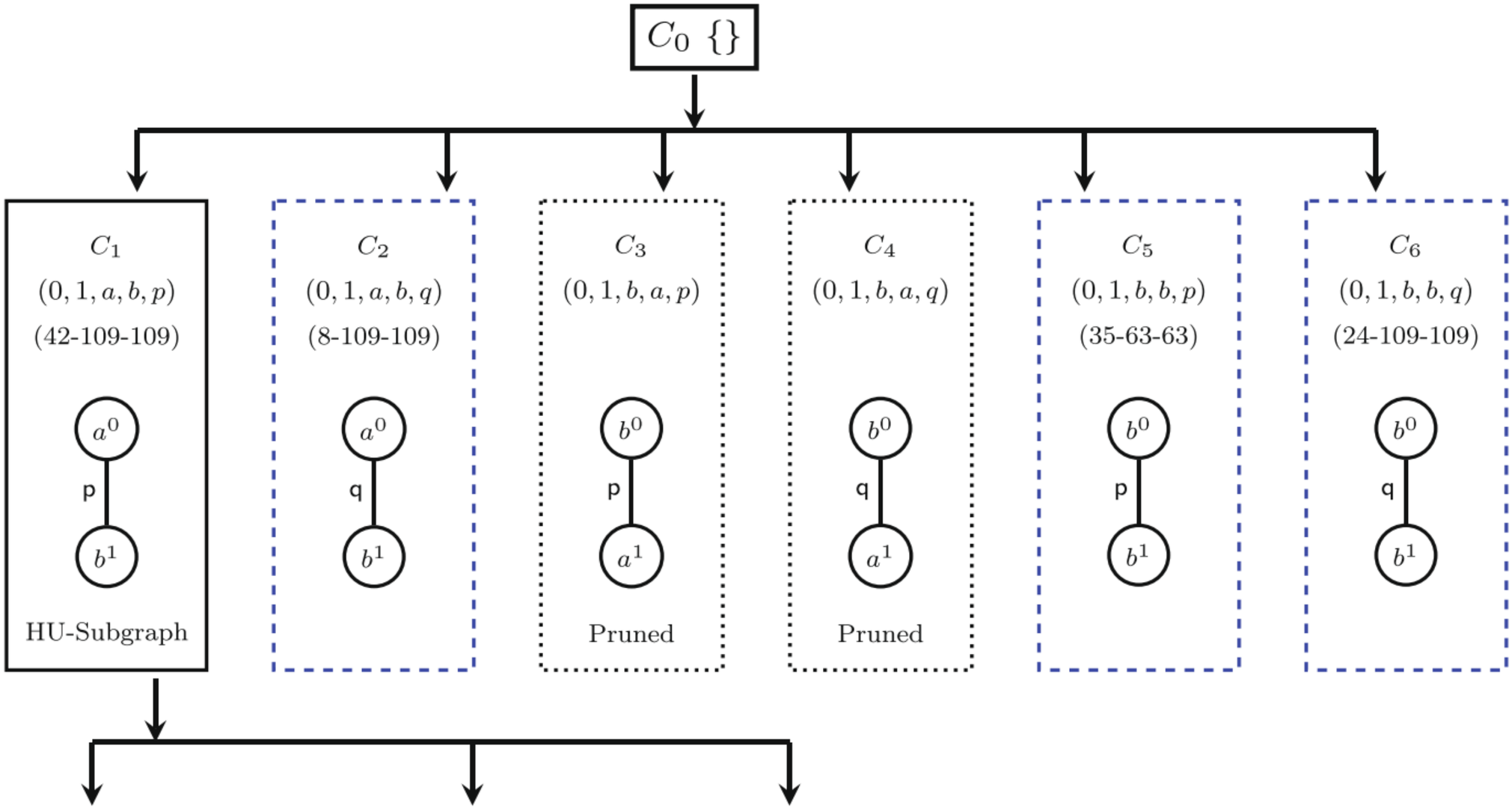

C_0 {}
C_1 ⟨0, 1, a, b, p⟩ ⟨42-109-109⟩ HU-Subgraph
C_2 ⟨0, 1, a, b, q⟩ ⟨8-109-109⟩
C_3 ⟨0, 1, b, a, p⟩ Pruned
C_4 ⟨0, 1, b, a, q⟩ Pruned
C_5 ⟨0, 1, b, b, p⟩ ⟨35-63-63⟩
C_6 ⟨0, 1, b, b, q⟩ ⟨24-109-109⟩
C_7 ⟨0, 1, a, b, p⟩ ⟨0, 2, a, b, q⟩ ⟨30-46-30⟩ Pruned
C_8 ⟨0, 1, a, b, p⟩ ⟨1, 2, b, b, q⟩ ⟨66-109-109⟩ HU-Subgraph
C_9 ⟨0, 1, a, b, p⟩ ⟨1, 2, b, a, q⟩ ⟨44-63-63⟩ HU-Subgraph
C_{14} ⟨0, 1, a, b, p⟩ ⟨1, 2, b, a, q⟩ ⟨1, 3, b, b, q⟩ ⟨28-63-63⟩
C_{15} ⟨0, 1, a, b, p⟩ ⟨1, 2, b, a, q⟩ ⟨1, 3, b, b, q⟩ ⟨3, 4, b, b, p⟩ ⟨63-63-63⟩ HU-Subgraph
C_{10} ⟨0, 1, a, b, p⟩ ⟨1, 2, b, b, q⟩ ⟨2, 0, b, a, q⟩ ⟨46-46-46⟩ HU-Subgraph
C_{11} ⟨0, 1, a, b, p⟩ ⟨1, 2, b, b, q⟩ ⟨1, 3, b, a, q⟩ ⟨28-63-28⟩ Pruned
C_{12} ⟨0, 1, a, b, p⟩ ⟨1, 2, b, b, q⟩ ⟨2, 3, b, b, q⟩ ⟨61-63-63⟩ HU-Subgraph
C_{13} ⟨0, 1, a, b, p⟩ ⟨1, 2, b, b, q⟩ ⟨2, 3, b, b, p⟩ ⟨1, 4, b, a, q⟩ Pruned


C_2 ⟨0, 1, a, b, q⟩
C_5 ⟨0, 1, b, b, p⟩ ⟨35-63-63⟩
C_{22} ⟨0, 1, b, b, p⟩ ⟨1, 2, b, b, q⟩ ⟨43-63-63⟩ HU-Subgraph
C_6 ⟨0, 1, b, b, q⟩ ⟨24-109-109⟩
C_{27} ⟨0, 1, b, b, q⟩ ⟨1, 2, b, b, p⟩ Pruned
C_{16} ⟨0, 1, a, b, q⟩ ⟨1, 2, b, a, p⟩ Pruned
C_{17} ⟨0, 1, a, b, q⟩ ⟨1, 2, b, b, q⟩ 32-109-109
C_{23} ⟨0, 1, b, b, p⟩ ⟨1, 2, b, b, q⟩ ⟨2, 3, b, a, p⟩ Pruned
C_{24} ⟨0, 1, b, b, p⟩ ⟨1, 2, b, b, q⟩ ⟨2, 3, b, a, q⟩ Pruned
C_{25} ⟨0, 1, b, b, q⟩ ⟨1, 2, b, a, p⟩ Pruned
C_{26} ⟨0, 1, b, b, q⟩ ⟨1, 2, b, a, q⟩ Pruned
C_{18} ⟨0, 1, a, b, q⟩ ⟨1, 2, b, b, q⟩ ⟨2, 0, b, a, p⟩ Pruned
C_{19} ⟨0, 1, a, b, q⟩ ⟨1, 2, b, b, q⟩ ⟨1, 3, b, a, p⟩ Pruned
C_{20} ⟨0, 1, a, b, q⟩ ⟨1, 2, b, b, q⟩ ⟨2, 3, b, b, p⟩ ⟨45-63-63⟩ HU-Subgraph
C_{21} ⟨0, 1, a, b, q⟩ ⟨1, 2, b, b, q⟩ ⟨2, 3, b, b, p⟩ ⟨1, 4, b, a, p⟩ Pruned

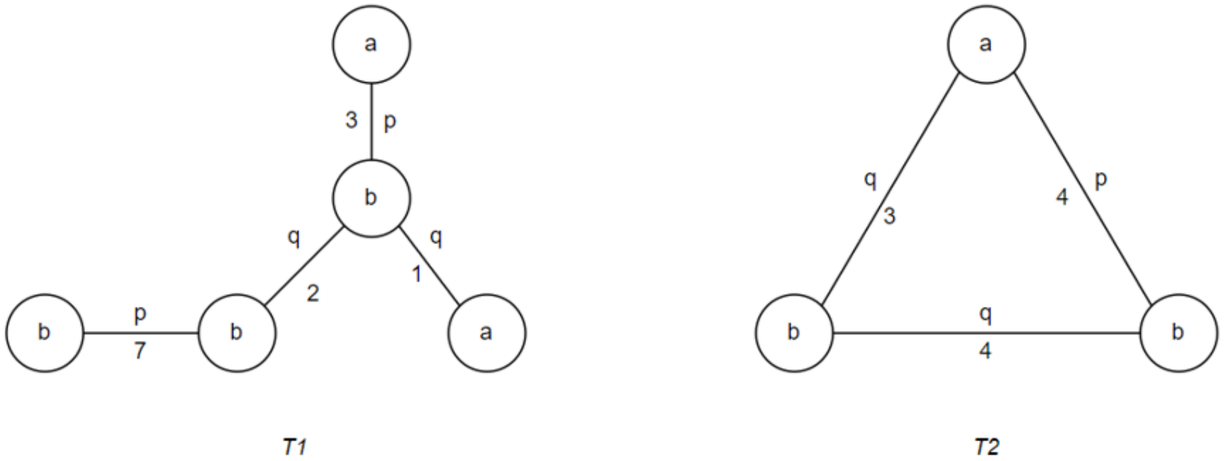
SIMULATION

- The minimum utility threshold: **35%**.
- minutil = 0.35 * 109 = **38.15**

SIMULATION


C_0 {}
C_1
$(0, 1, a, b, p)$
(42-109-109)
a^0
p
b^1
HU-Subgraph
C_2
$(0, 1, a, b, q)$
(8-109-109)
a^0
q
b^1
C_3
$(0, 1, b, a, p)$
b^0
p
a^1
Pruned
C_4
$(0, 1, b, a, q)$
b^0
q
a^1
Pruned
C_5
$(0, 1, b, b, p)$
(35-63-63)
b^0
p
b^1
C_6
$(0, 1, b, b, q)$
(24-109-109)
b^0
q
b^1

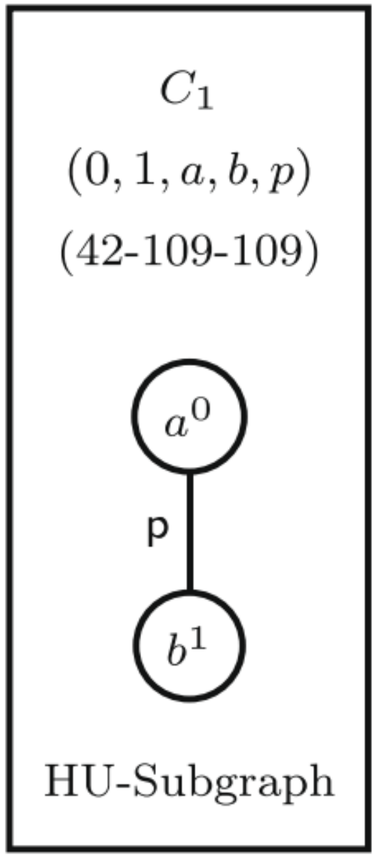
SIMULATION



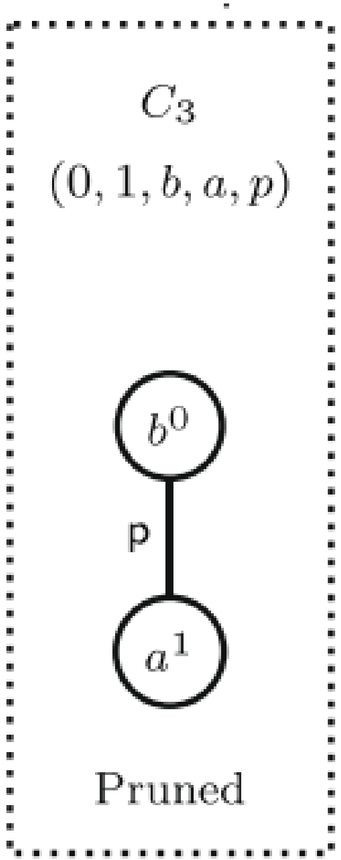
Hình a: *một cơ sở dữ liệu đồ thị được gán nhãn định lượng D*

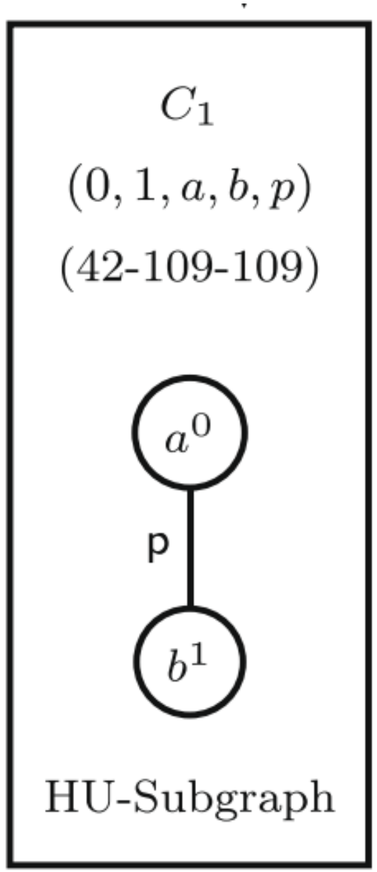
Nhãn điểm 1	Nhãn điểm 2	Nhãn cạnh	Độ tiện dụng ngoài
a	b	p	6
a	b	q	2
b	b	p	5
b	b	q	4

Bảng độ tiện dụng ngoài



- C1 subgraph utility:
  (4+3) × 6 = 7 × 6 = **42**
- C1 GWU value:
  63 + 46 = **109**
- C1 RMU value: **109**

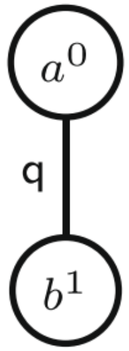

C_3
$(0, 1, b, a, p)$
b^0
p
a^1
Pruned


C_1
$(0, 1, a, b, p)$
(42-109-109)
a^0
p
b^1
HU-Subgraph

C_2

$(0, 1, a, b, q)$

(8-109-109)

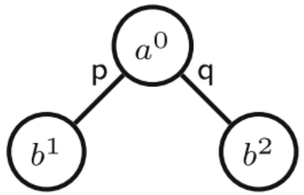


C_7

$(0, 1, a, b, p)$

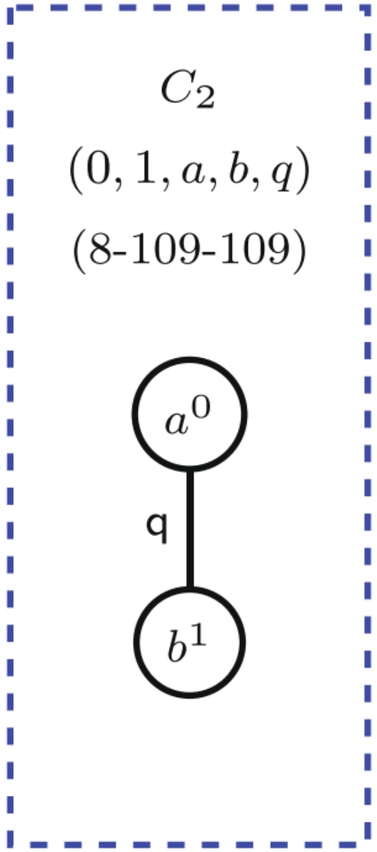
$(0, 2, a, b, q)$

(30-46-30)

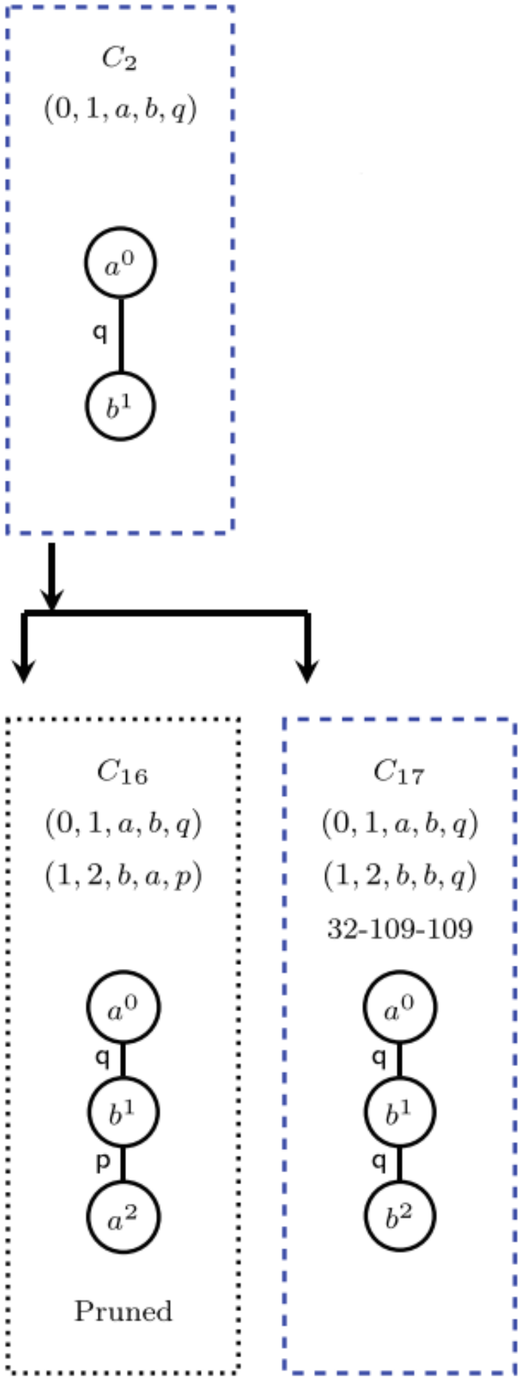


Pruned

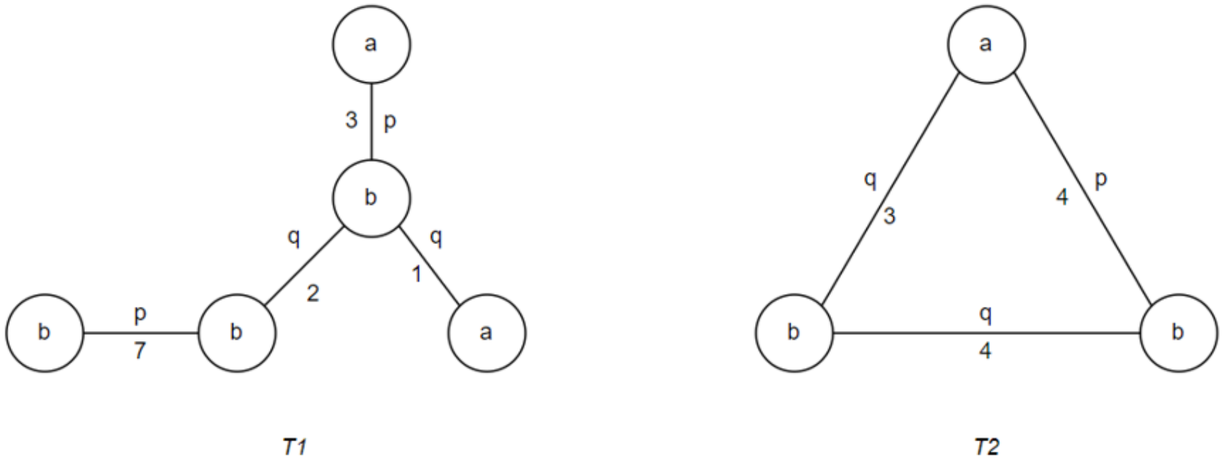
SIMULATION



- C2 subgraph utility: **8** < minutil = 38.15
- C2 GWU value: **109**
- C2 RMU value: **109**


C_2
$(0, 1, a, b, q)$
a^0
q
b^1
C_{16}
$(0, 1, a, b, q)$
$(1, 2, b, a, p)$
a^0
q
b^1
p
a^2
Pruned
C_{17}
$(0, 1, a, b, q)$
$(1, 2, b, b, q)$
32-109-109
a^0
q
b^1
q
b^2

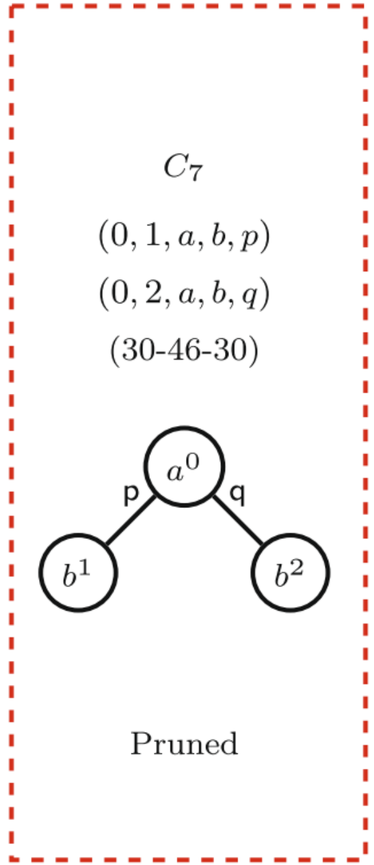
SIMULATION



Hình a: *một cơ sở dữ liệu đồ thị được gán nhãn định lượng D*

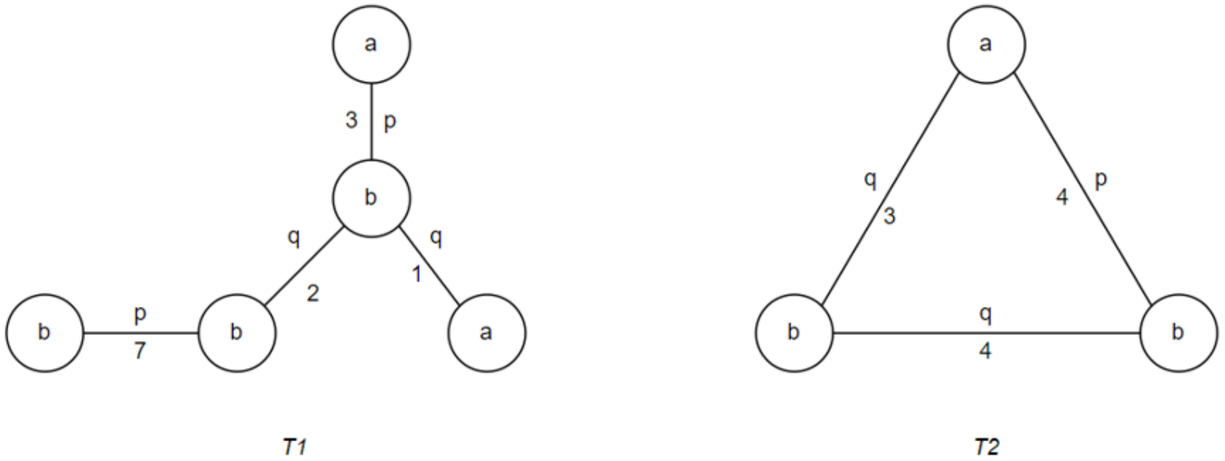
Nhãn điểm 1	Nhãn điểm 2	Nhãn cạnh	Độ tiện dụng ngoài
a	b	p	6
a	b	q	2
b	b	p	5
b	b	q	4

Bảng độ tiện dụng ngoài



- C7 GWU value: **46**
- (b-b-q) utility: 4×4 = **16**
- C7 RMU value:
  46-16 = **30** < minutil = 38.15

SIMULATION



Hình a: *một cơ sở dữ liệu đồ thị được gán nhãn định lượng D*

Nhãn điểm 1	Nhãn điểm 2	Nhãn cạnh	Độ tiện dụng ngoài
a	b	p	6
a	b	q	2
b	b	p	5
b	b	q	4

Bảng độ tiện dụng ngoài

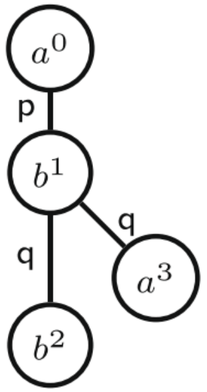
C_{11}

$(0, 1, a, b, p)$

$(1, 2, b, b, q)$

$(1, 3, b, a, q)$

(28-63-28)

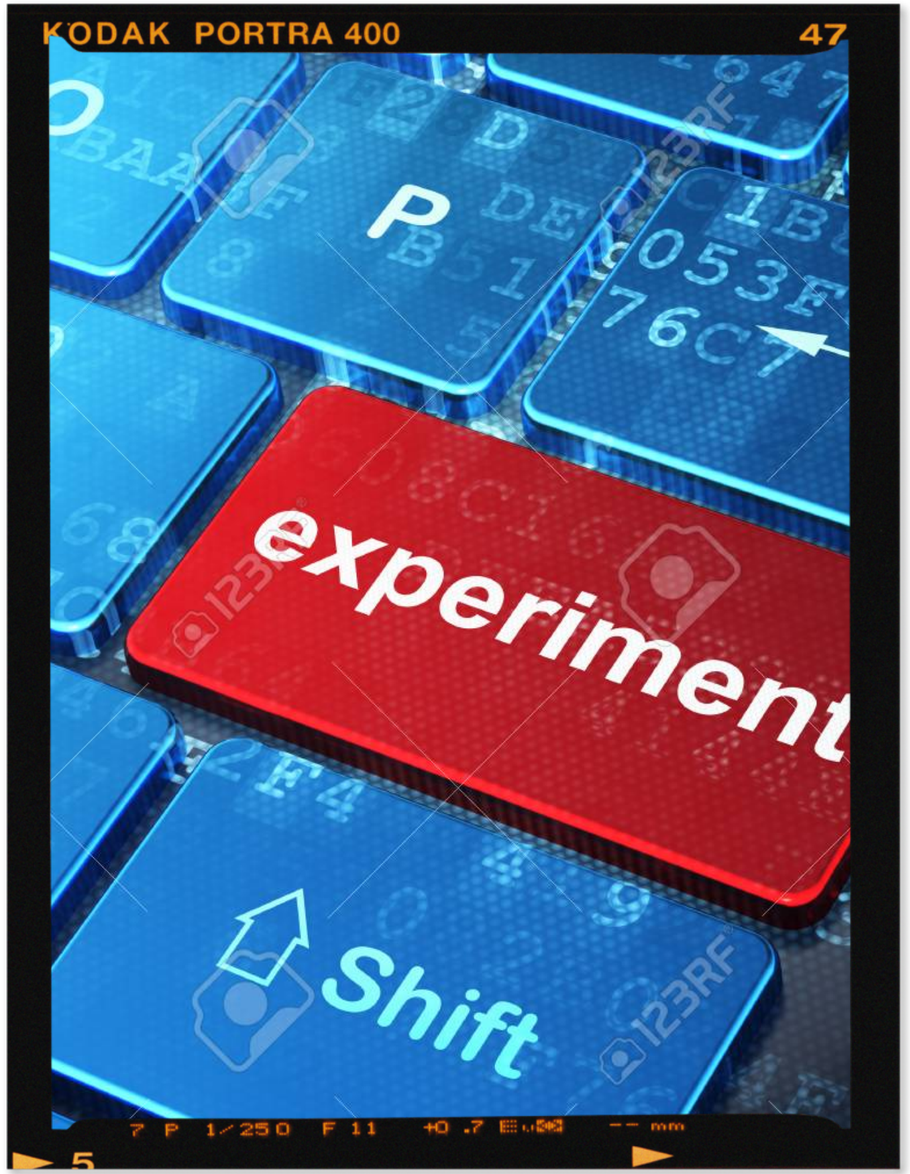


Pruned

- C11 GWU value: **63**
- (b-b-p) utility: 7 × 5 = **35**
- C11 RMU value:
  63 − 35 = **28** < minutil = 38.15

EXPERIMENTS

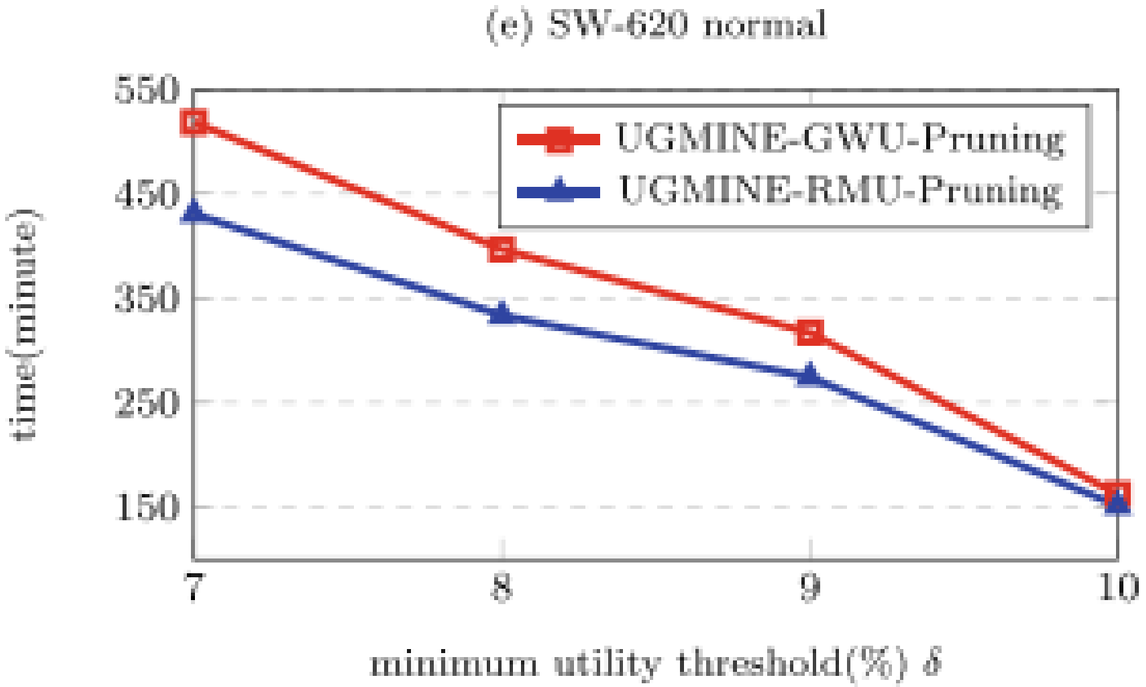
Compare the performance analysis for both **normal** and **log-normal** distributions of **internal** and **external** utility show that the **UGMINE-RMU pruning** technique outperforms the baseline approach **UGMINE-GWU**

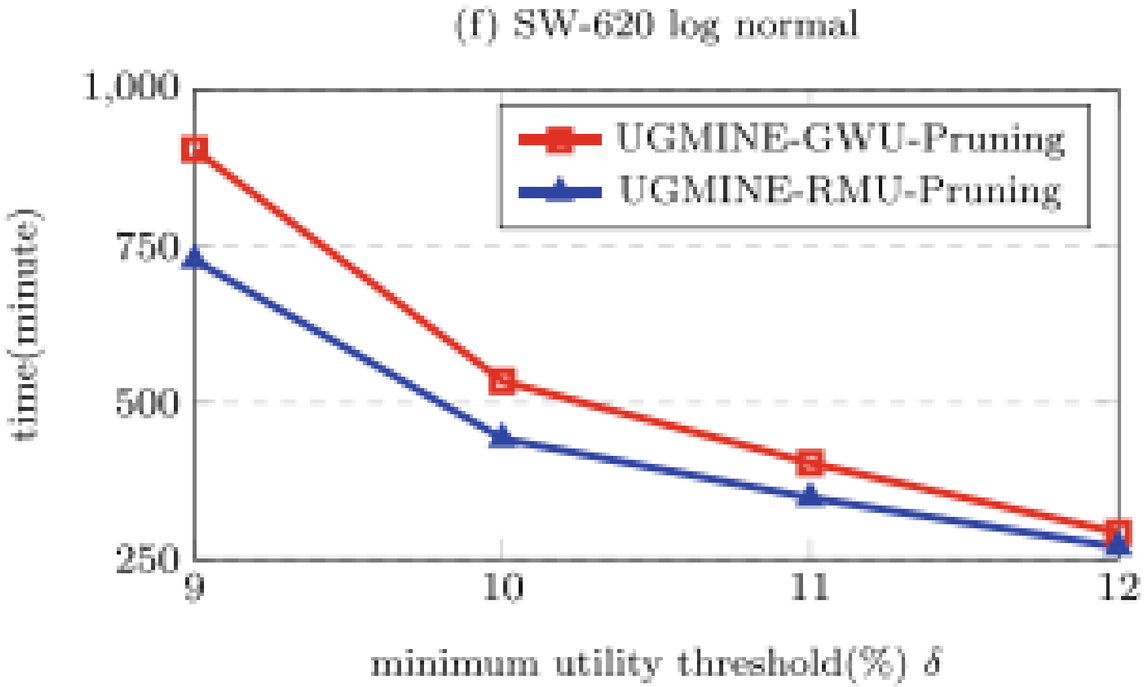


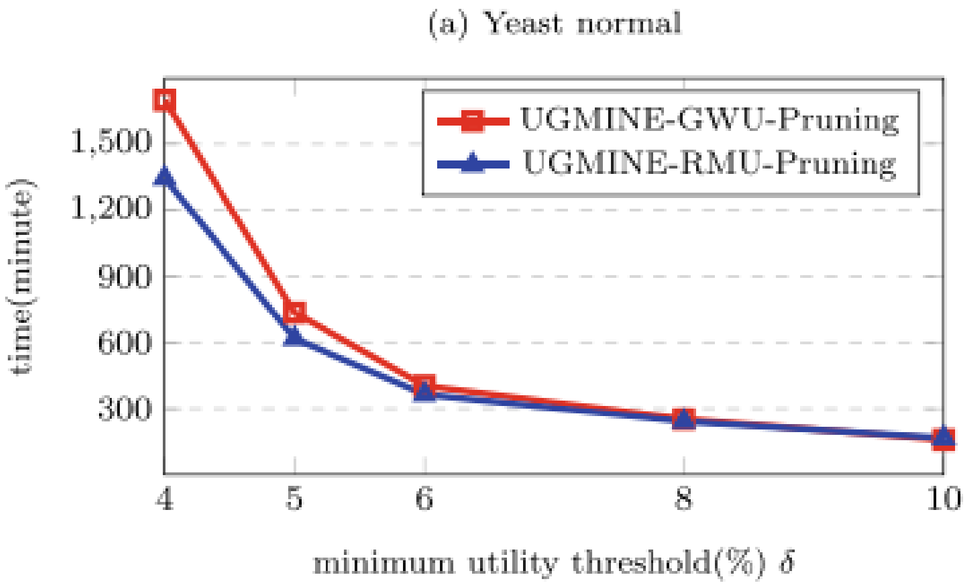
- Experiments using a PC with an Intel Core i7-6700K CPU at 4.00 GHz and 16 GB RAM
- Use runtime, search space reduction, and memory usage as the performance evaluation metrics

δ	UGMINE-GWU pruning			UGMINE-RMU pruning		
	Runtime (minute)	Candidates	HU Sub-Graphs	Runtime (minute)	Candidates	HU Sub-Graphs
(a) Normal Distribution						
7%	518.71	5417	65	430.80	1820	65
8%	396.89	4105	205	333.57	1620	205
9%	317.13	3248	254	274.35	1320	254
10%	162.21	1569	310	151.03	1120	310
(b) Log-normal Distribution						
9%	904.45	547381	136	727.63	417865	136
10%	534.05	369515	91	442.18	287463	91
11%	403.79	257307	65	347.78	211362	65
12%	292.28	193673	44	269.34	162129	44

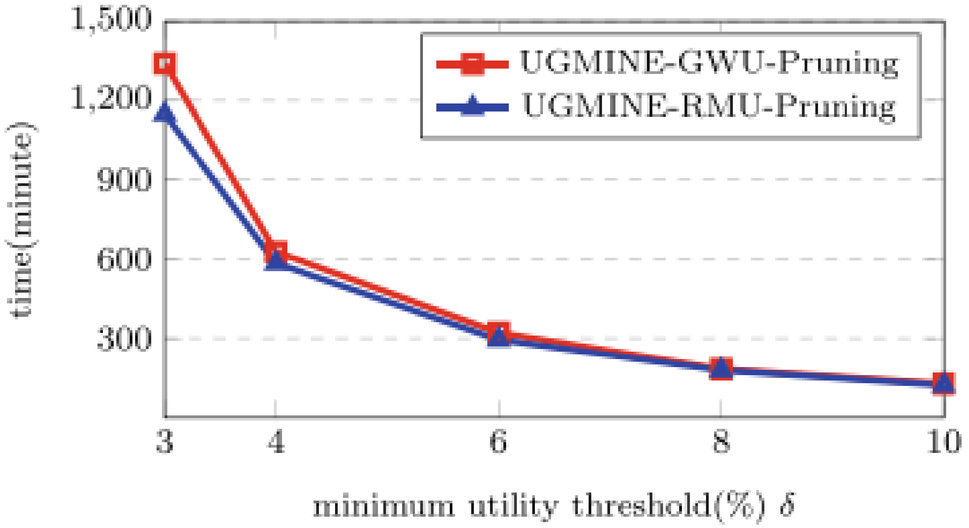
Runtime and search space reduction statistics: SW-620


(e) SW-620 normal
550
450
350
250
150
time(minute)
UGMINE-GWU-Pruning
UGMINE-RMU-Pruning
7
8
9
10
minimum utility threshold(%) δ


(f) SW-620 log normal
1,000
750
500
250
time(minute)
UGMINE-GWU-Pruning
UGMINE-RMU-Pruning
9
10
11
12
minimum utility threshold(%) δ



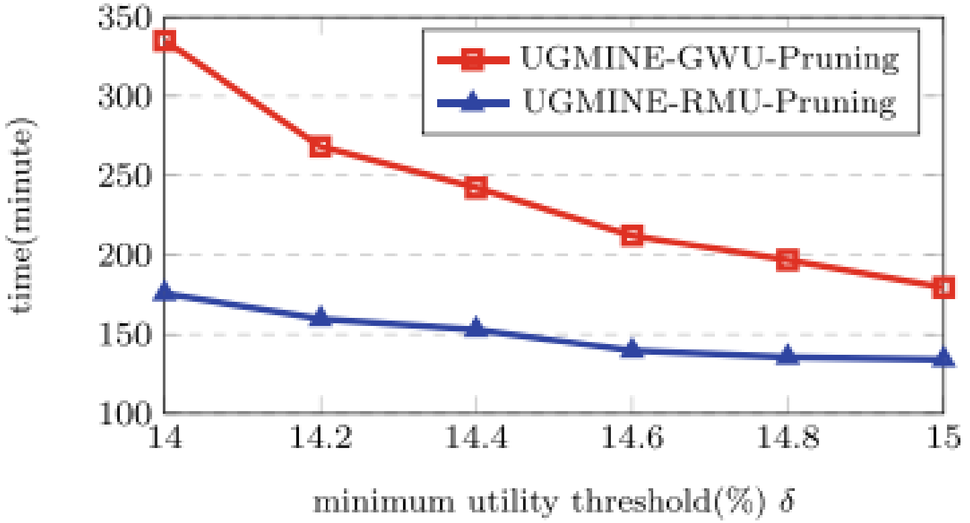
(a) Yeast normal



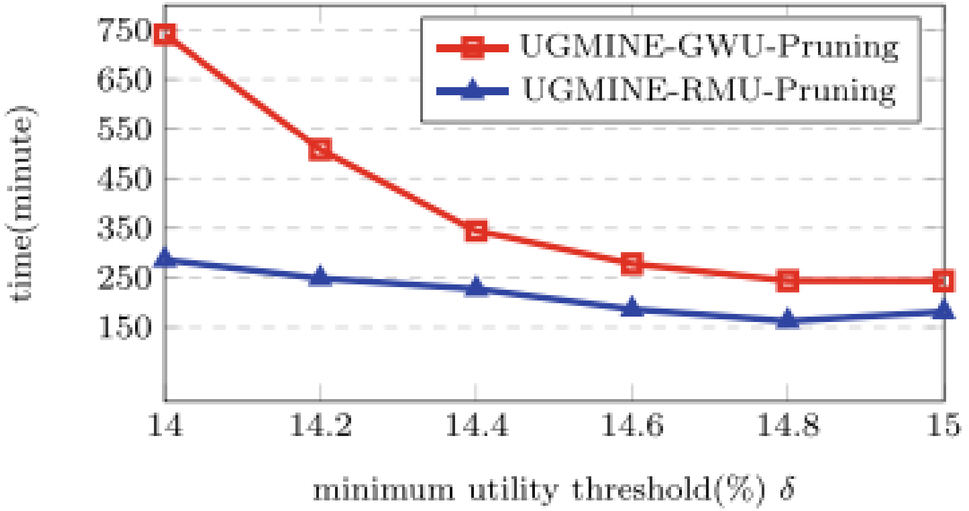
(b) Yeast log normal

(c) p388 normal

(d) p388 log normal

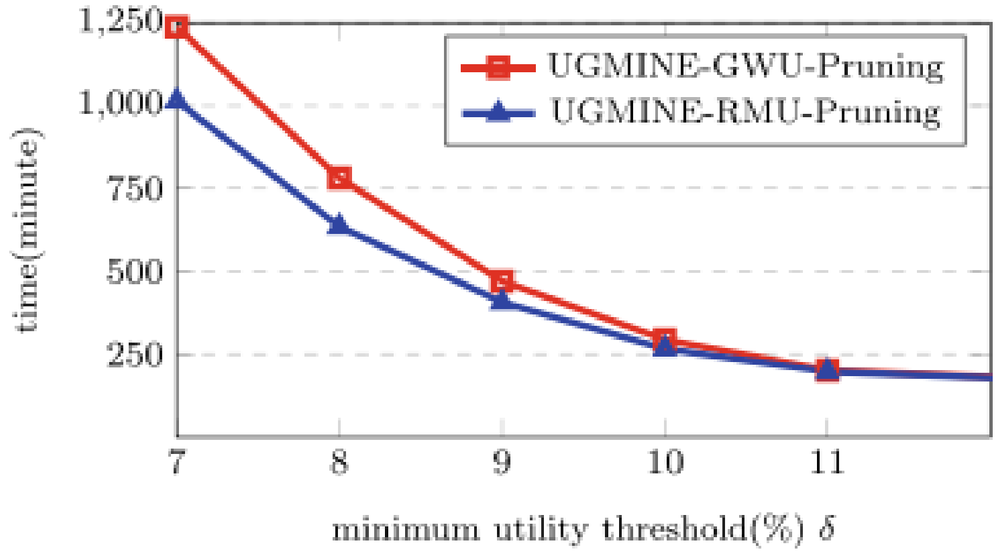


(e) SW-620 normal

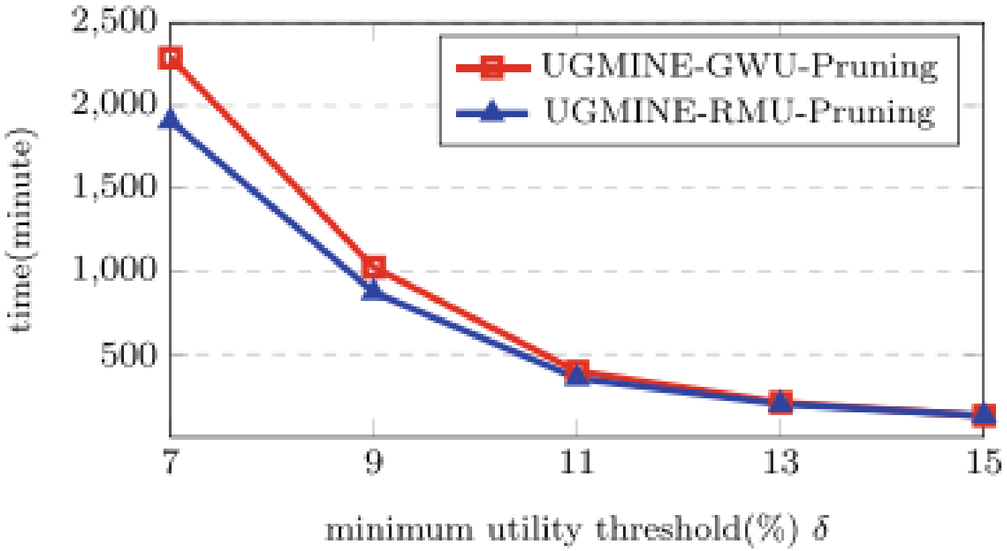


(f) SW-620 log normal

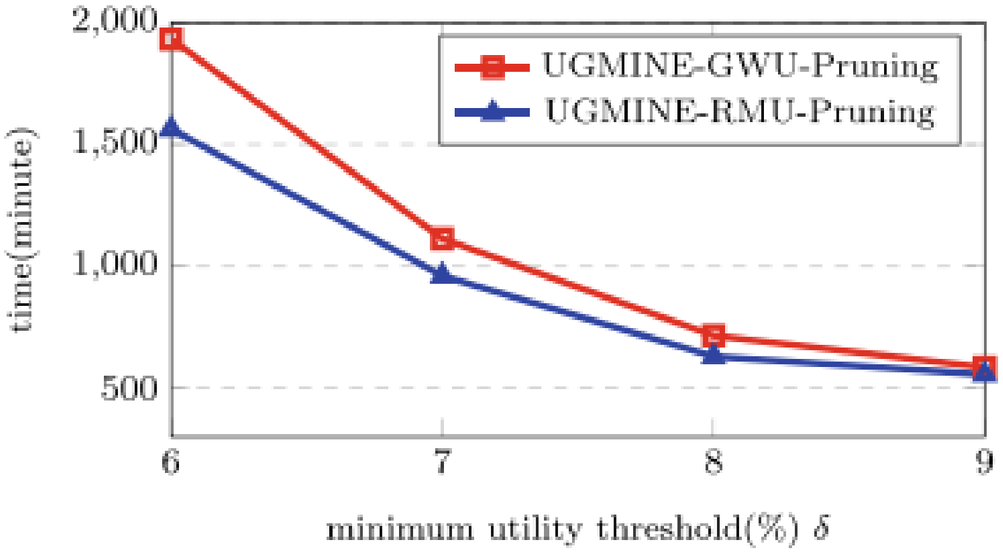
(g) MOLT-4 normal



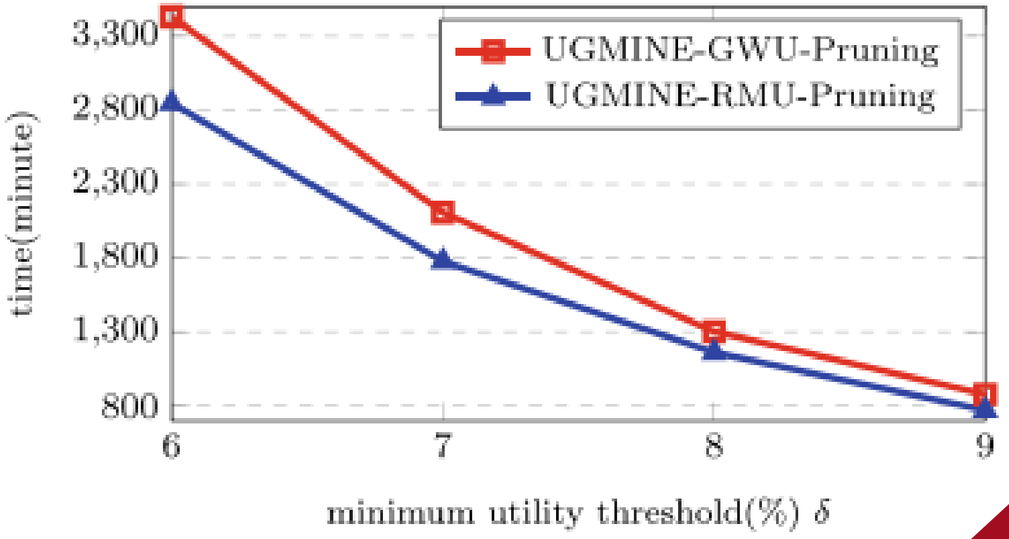
(h) MOLT-4 log normal



(i) NCI1 normal



(j) NCI1 log normal



- The low memory usage of UGMINE-RMU is explained by the low number of generated false candidates.
- We also observe that the memory usage is relatively higher for lower utility thresholds as the utility threshold decreases, the search space explodes, which leads to longer runtimes and higher memory costs.
- The performance gap is wider in the normal distribution than in the log-normal distribution, both in terms of runtime and search space reduction.


conclusion

- In this Paper, authors introduced a complete framework for utility-based graph mining.
- They also proposed an algorithm named UGMINE for extracting high utility subgraph patterns combined with efficient pruning techniques.
- The framework and methods proposed are expected to be helpful for solving problems in areas such as: analyzing web page access log networks, chemical structure databases, social networks, and anywhere using graph databases.
- In the future, they are exploring some tighter pruning techniques in order to support larger datasets.
- They are also modifying the framework based on the application such that summation, minimum or other complex utility functions used as a replacement of maximum.
- Finally, they are examining negative utility or multi-edged negative utility.

JUDGEMENT

	TASK 1	Rate	TASK 2	Rate
18127164	Graph Patterns	100%	Slide & Report	100%
20127028	Graph Classification	100%	Slide & Report	100%
20127054	Link Prediction	100%	Slide & Report	100%
20127214	Frequent Subgraph Mining	100%	Slide & Report	100%
All IDs	Community Detection	100%		


THANK
YOU