# CHARM: An Efficient Algorithm for Closed Itemset Mining

Authors: Mohammed J. Zaki and Ching-Jui Hsiao
Presenter: Junfeng Wu



# Outline

- Introductions
- Itemset-Tidset tree
- CHARM algorithm
- Performance study
- Conclusion
- Comments



# Introductions

When we are mining association rules in a database, a huge number of frequent patterns (itemsets) will be generated.

- Database: {(1,2,3,4),(1,2,3,4,5,6)}
- Minimum support = 50%
- 63 frequent itemsets ({(1),(2),(3),(4),(5),(6),(1,2),(1,3),…,(1,2,3,4,5,6)})



# Introductions

Closed frequent itemsets are non-redundant representations of all frequent itemsets.

Mining association rules on closed frequent itemsets is a much easier task.

In the previous database, the number of closed frequent itemsets is only 2, (1,2,3,4) and (1,2,3,4,5,6).

# Closed frequent itemsets

- A frequent itemset *X* is closed if and only if there is no itemset *Y* such that
  - *Y* subsumes *X*
  - every transaction that contains *X* also contains *Y*

Database: {(1,2,3,4),(1,2,3,4,5,6)}
Itemset (1,2) is not a closed itemset.
Itemset (1,2,3,4) is a closed itemset.



# Example Database

**DISTINCT DATABASE ITEMS**

| Jane Austen | Agatha Christie | Sir Arthur Conan Doyle | Mark Twain | P.G. Wodehouse |
|---|---|---|---|---|
| A | C | D | T | W |

**DATABASE**

| *Transaction* | *Items* |
|---|---|
| 1 | A,C,T,W |
| 2 | C,D,W |
| 3 | A,C,T,W |
| 4 | A,C,D,W |
| 5 | A,C,D,T,W |
| 6 | C,D,T |

**ALL FREQUENT ITEMSETS**
**MINIMUM SUPPORT = 50%**

| *Support* | *Itemsets* |
|---|---|
| 100%(6) | C |
| 83%(5) | W,CW |
| 67%(4) | A,D,T,AC,AW,CD,CT,ACW |
| 50%(3) | AT,DW,TW,ACT,ATW,CDW,CTW,ACTW |



# Horizontal/Vertical format database

- Horizontal format database
  - Each record is a set of items.
  - Each record is assigned a distinct number named transaction id.
- Vertical format database
  - Each record is a set of transaction id about an item.
  - This item occurs in these transactions.



# Vertical format database

| A | C | D | T | W |
|---|---|---|---|---|
| 1 | 1 | 2 | 1 | 1 |
| 3 | 2 | 4 | 3 | 2 |
| 4 | 3 | 5 | 5 | 3 |
| 5 | 4 | 6 | 6 | 4 |
|  | 5 |  |  | 5 |
|  | 6 |  |  |  |

# Notations

Given an itemset *X*, *t(X)* is the set of all tids that contains *X*.

*For example: t(ACW) = 1345*

Given a tidset *Y*, *i(Y)* is the set of all common items to all the tids in *Y*.

*For example: i(12) = CW*

Given an itemset *X*, *c(X)* is the smallest closed set that contains *X*.

*For example: c(A)=c(C)=C(W)=ACW*

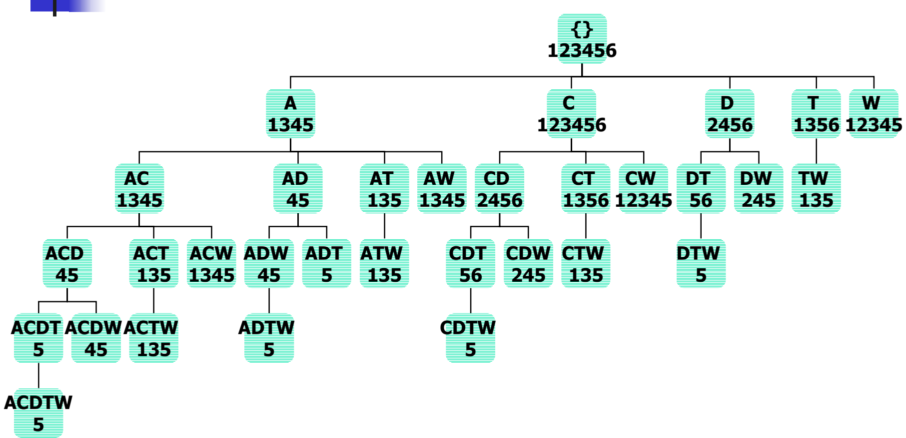


# Itemset-Tidset Search Tree (IT-tree)

- Each node in the IT-tree is an itemset-tidset pair, *X×t(X).*
  *For example: AT×135*
- All the children of node *X* share the same prefix *X* and belong to an equivalence class



# Example of IT-tree






# Theorem 1

- Let $X_i \times t(X_i)$ and $X_j \times t(X_j)$ be any two members of a class $[p]$, with $X_i \leq_f X_j$ , where $f$ is a total order. The following four properties hold:
  - 1. If $t(X_i) = t(X_j)$, then $c(X_i) = c(X_j) = c(X_i \cup X_j)$
  - 2. If $t(X_i) \subset t(X_j)$, then $c(X_i) \neq c(X_j)$ , but $c(X_i) = c(X_i \cup X_j)$
  - 3. If $t(X_i) \supset t(X_j)$, then $c(X_i) \neq c(X_j)$ , but $c(X_j) = c(X_i \cup X_j)$
  - 4. If $t(X_i) \neq t(X_j)$, then $c(X_i) \neq c(X_j) \neq c(X_i \cup X_j)$

# CHARM algorithm

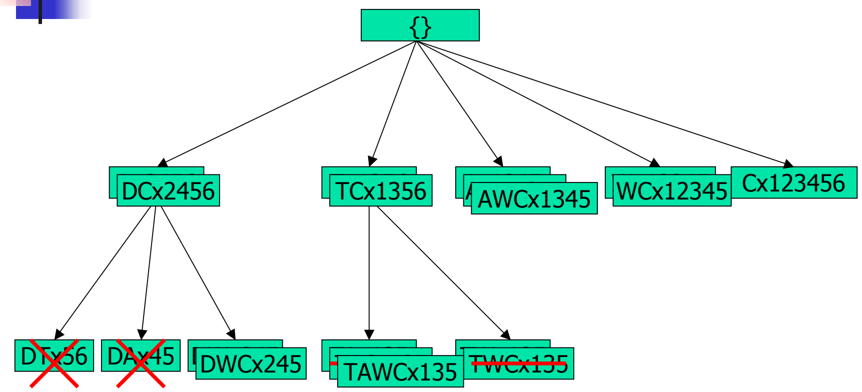
```
CHARM (D, min_sup):
    [P] = {X_i × t(X_i) : X_i ∈ I ∧ σ(X_i) ≥ min_sup}
    CHARM-EXTEND ([P], C = ∅)
    return C //all closed sets

CHARM-EXTEND ([P], C):
    for each X_i × t(X_i) in [P]
        [P_i] = ∅ and X = X_i
        for each X_j × t(X_j) in [P], with X_j ≥_f X_i
            X = X ∪ X_j and Y = t(X_i) ∩ t(X_j)
            CHARM-PROPERTY([P], [P_i])
        if ([P_i] ≠ ∅) then CHARM-EXTEND ([P_i], C)
        delete [P_i]
        C = C ∪ X //if X is not subsumed

CHARM-PROPERTY ([P], [P_i]):
    if (σ(X) ≥ minsup) then
        if t(X_i) = t(X_j) then //Property 1
            Remove X_j from [P]
            Replace all X_i with X
        else if t(X_i) ⊂ t(X_j) then //Property 2
            Replace all X_i with X
        else if t(X_i) ⊃ t(X_j) then //Property 3
            Remove X_j from [P]
            Add X × Y to [P_i] //use ordering f
        else if t(X_i) ≠ t(X_j) then //Property 4
            Add X × Y to [P_i] //use ordering f
```



# How does CHARM work?






# Subsumption Checking

Before add a set X to the current set of closed set, we need check if X is subsumed by some closed sets.

- Comparing X with all closed set is expensive.

Solution: using hash function to retrieve relevant closed sets



# Hash function

$$h(X) = \sum_{T \in t(X)} T$$

The sum of the tids in the tidset of an itemset

- Assumption: itemsets with the same hash key have different supports.

# Complexity issues

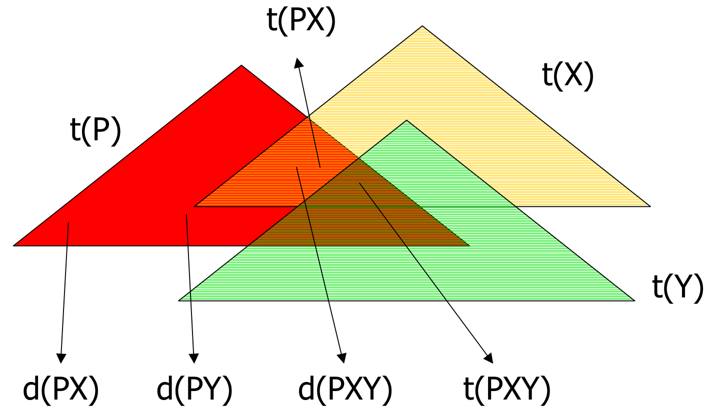
Comparing two itemset's tidsets becomes a time consuming task when tidset gets very large.

Keeping all tids of itemsets in memory needs lots of space.

Solution: using diffsets



# Diffsets






# Diffset and Tidset

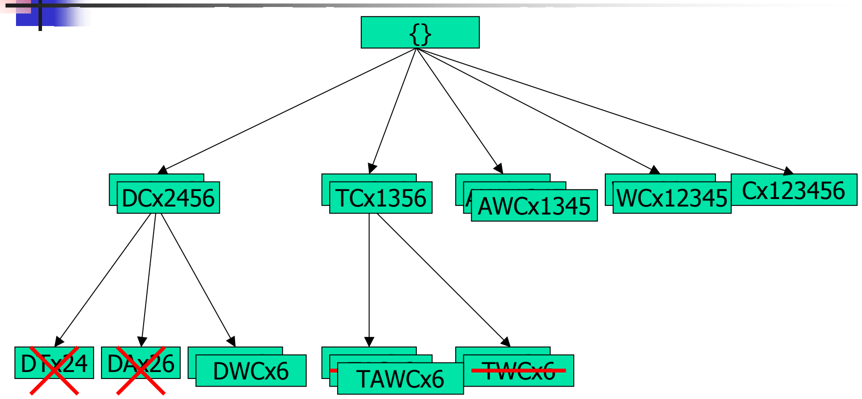
Let *$m(X_i)$* and *$m(X_j)$* denote the number of mismatches in the diffsets *$d(X_i)$* and *$d(X_j)$*

For example: $X_i$=D, $X_j$=T, then $d(X_i)$=2456, $d(X_j)$=1356, $m(X_i)$=|(13)|=2, $m(X_j)$=|(24)|=2

$$
\begin{array}{llllllll}
m(X_i)=0 & and & m(X_j)=0, & then & d(X_i)=d(X_j) & or & t(X_i)=t(X_j) \\
m(X_i)>0 & and & m(X_j)=0, & then & d(X_i)\supset d(X_j) & or & t(X_i)\subset t(X_j) \\
m(X_i)=0 & and & m(X_j)>0, & then & d(X_i)\subset d(X_j) & or & t(X_i)\supset t(X_j) \\
m(X_i)>0 & and & m(X_j)>0, & then & d(X_i)\neq d(X_j) & or & t(X_i)\neq t(X_j)
\end{array}
$$



# CHARM using *diffsets*



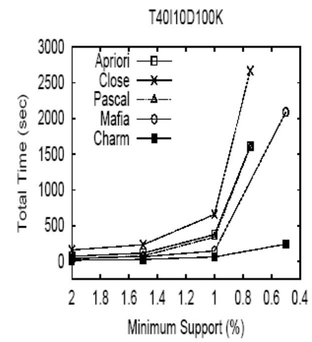
# Performance study

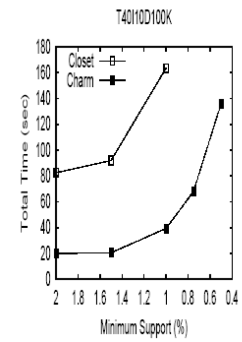
- Datasets

| Database | # Items | Avg. Length | Std. Dev. | # Records |
|---|---|---|---|---|
| chess | 76 | 37 | 0 | 3,196 |
| connect | 130 | 43 | 0 | 67,557 |
| mushroom | 120 | 23 | 0 | 8,124 |
| pumsb* | 7117 | 50 | 2 | 49,046 |
| pumsb | 7117 | 74 | 0 | 49,046 |
| gazelle | 498 | 2.5 | 4.9 | 59,601 |
| T10I4D100K | 1000 | 10 | 3.7 | 100,000 |
| T40I10D100K | 1000 | 40 | 8.5 | 100,000 |



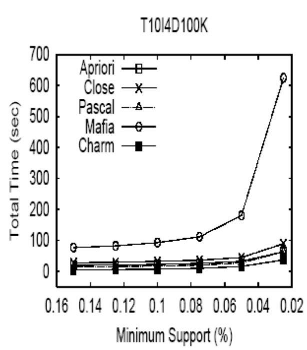
# Performance study

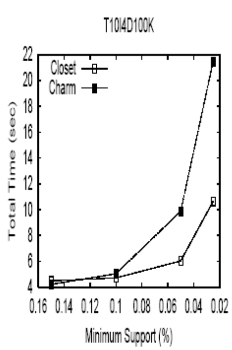




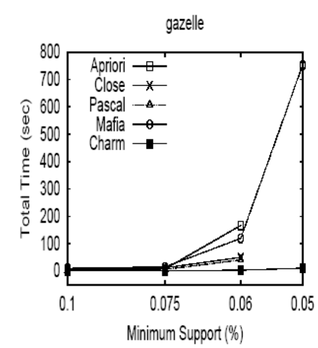

# Performance study

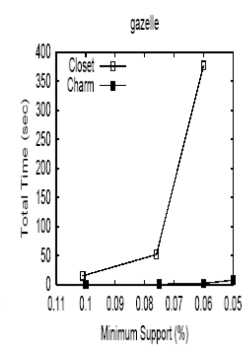




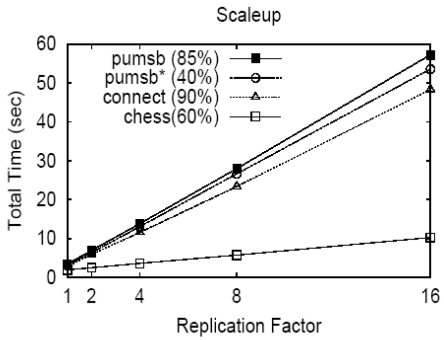

# Performance study

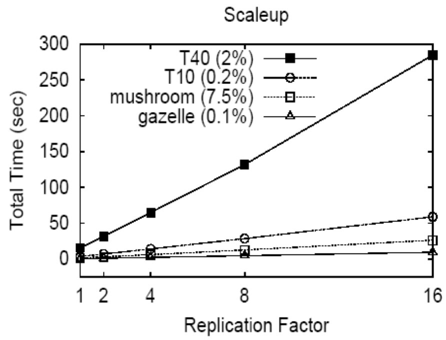


# Scalability

Linear increasing in the running time with increasing number of transactions at a giving support.






# Memory usage

The memory usage is 50 times smaller by using diffsets than using tidsets.

Memory usage (using diffsets)

| DB | 50% | 20% | DB | 0.1% | 0.05% | DB | 1% | 0.5% |
|---|---|---|---|---|---|---|---|---|
| connect | 0.68MB | 1.17MB | gazelle | 0.13MB | 1.24MB | T40I10D100K | 0.39MB | 0.52MB |



# Conclusion

- Advantage of CHARM
  - Faster than other algorithm at low support threshold
  - Faster than other algorithm on a database with very long closed patterns
- Disadvantage of CHARM
  - Slower than Closet when most of closed sets are 2-itemset



# Comments

- Strength
  - The ideas in the paper are intuitive.
  - The authors first introduced an efficient data structure (IT-tree) for closed itemset mining.
  - The authors demonstrated the algorithm on various datasets.
  - The experimental studies are convincing.
- Weakness
  - The algorithm requires the conversion of database from horizontal format to vertical format.
- Follow-up
  - Closet+ (Wang et al, 2003) beats CHARM one year later.

# THANK YOU!

Questions or comments?