Approximate Frequent Pattern Mining

Philip S. Yu[1], Xifeng Yan[1], Jiawei Han[2], Hong Cheng[2], Feida Zhu[2]

[1]IBM T.J.Watson Research Center

[2]University of Illinois at Urbana-Champaign

Frequent Pattern Mining

- Frequent pattern mining has been studied for over a decade with tons of algorithms developed
  - Apriori (SIGMOD′93, VLDB′94, ...)
  - FPgrowth (SIGMOD′00), EClat, LCM, ...
- Extended to sequential pattern mining, graph mining, ...
  - GSP, PrefixSpan, CloSpan, gSpan, ...
- Applications: Dozens of interesting applications explored
  - Association and correlation analysis
  - Classification (CBA, CMAR, ..., discrim. feature analysis)
  - Clustering (e.g., micro-array analysis)
  - Indexing (e.g. g-Index)

The Problem of Frequent Itemset Mining

- First proposed by Agrawal et al. in 1993 [AIS93].

Transaction-id	Items bought
10	A, B, C
20	A
30	A, B, C, D
40	C, D
50	A, B
60	A, C, D
70	B, C, D

Table 1. A sample transaction database D

- Itemset X = {x1, …, xk}
- Given a minimum support s, discover all itemsets X, s.t. sup(X) >= s
- sup(X) is the percentage of transactions containing X
- If s=40%, X={A,B} is a frequent itemset since sup(X)=3/7 > 40%

A Binary Matrix Representation

- We can also use a binary matrix to represent a transaction database.
  - Row: Transactions
  - Column: Items
  - Entry: Presence/absence of an item in a transaction

	A	B	C	D
10	1	1	1	0
20	1	0	0	0
30	1	1	1	1
40	0	0	1	1
50	1	1	0	0
60	1	0	1	1
70	0	1	1	1

Table 2. Binary representation of D

A Noisy Data Model

- A noise free data model
  - Assumption made by all the above algorithms
- A noisy data model
  - Real world data is subject to random noise and measurement error. For example:
    - Promotions
    - Special events
    - Out-of-stock items or overstocked items
    - Measurement imprecision
  - The true frequent itemsets could be distorted by such noise.
  - The exact itemset mining algorithms will discover multiple fragmented itemsets, but miss the true ones.

Itemsets With and Without Noise

Exact mining algorithms get fragmented itemsets!

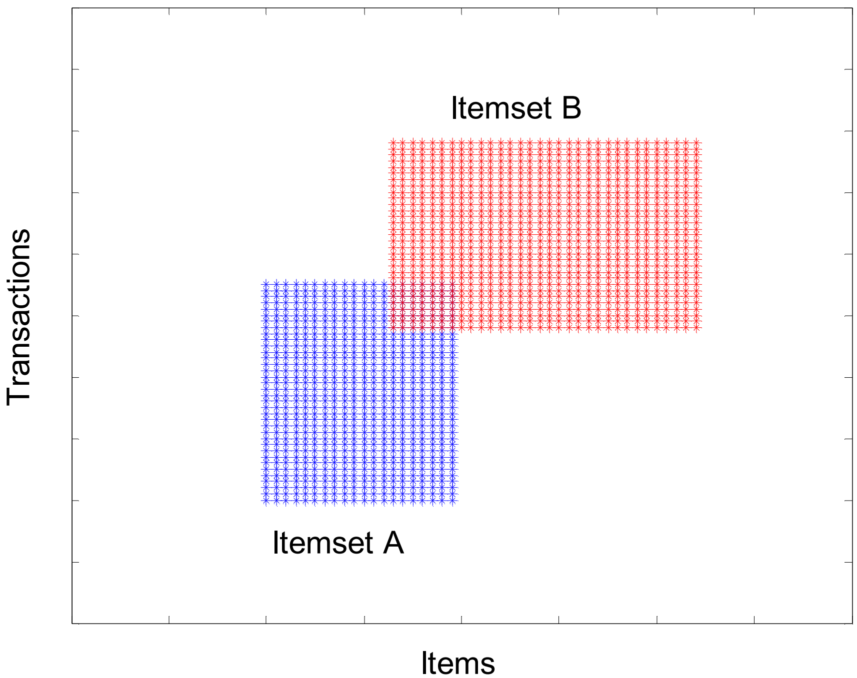


Figure1(a). Itemset without noise

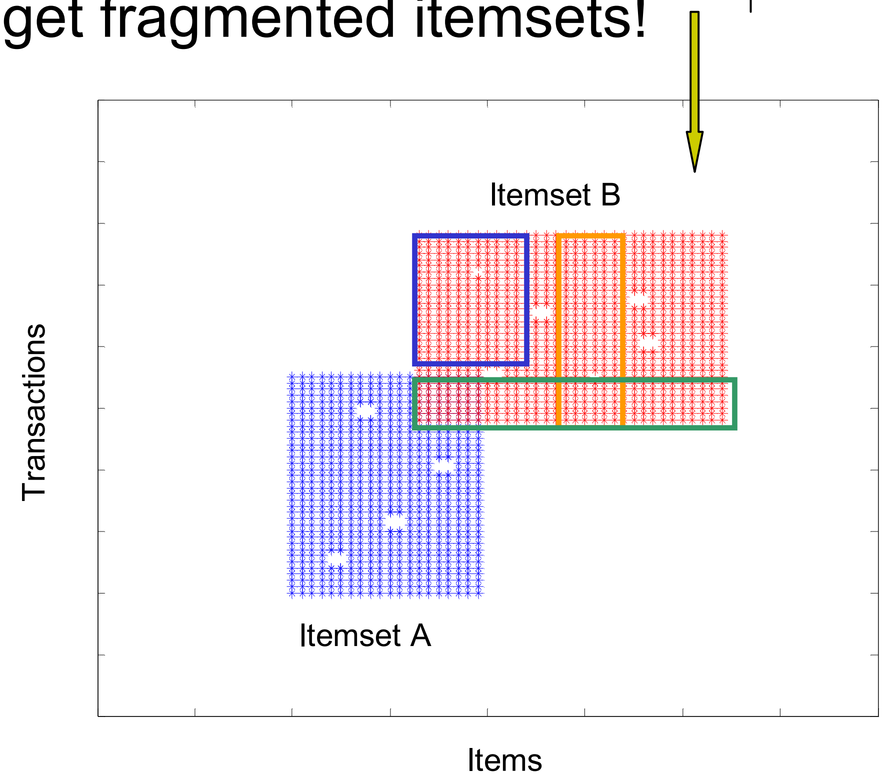


Figure 1(b). Itemset with noise

Alternative Models

- Existence of core patterns
  - I.E., even under noise, the original pattern can still appear with high probability
- Only summary patterns can be derived
  - Summary pattern may not even appear in the database

The Core Pattern Approach

- Core Pattern Definition
  - An itemset x is a core pattern if its exact support in the noisy database satisfies

$$\sup(x) \geq \alpha \cdot \min\sup, 0 \leq \alpha \leq 1$$

- If an approximate itemset is interesting, it is with high probability that it is a core pattern in the noisy database. Therefore, we could discover the approximate itemsets from only the core patterns.
- Besides the core pattern constraint, we use the constraints of minimum support, ε_r, and ε_c, as in [LPS+06].

Approximate Itemset Example

- Let $\varepsilon_r = 0.25$ and $\varepsilon_c = 0.25$
- For <ABCD>, its exact support = 1;
- By allowing a fraction of $\varepsilon_r = 0.25$ noise in a row, transaction 10, 30, 60, 70 all approximately support <ABCD>;
- For each item in <ABCD>, in the transaction set {10, 30, 60, 70}, a fraction of $\varepsilon_c = 0.25$ 0s is allowed.

	A	B	C	D
10	1	1	1	0
20	1	0	0	0
30	1	1	1	1
40	0	0	1	1
50	1	1	0	0
60	1	0	1	1
70	0	1	1	1

The Approximate Frequent Itemset Mining Approach

- Intuition
  - Discover approximate itemsets by allowing "holes" in the matrix representation.
- Constraints
  - Minimum support s: the percentage of transactions containing an itemset
  - Row error rate ε_r : the percentage of 0s (item) allowed in each transaction
  - Column error rate ε_c: the percentage of 0s allowed in transaction set for each item

Algorithm Outlines

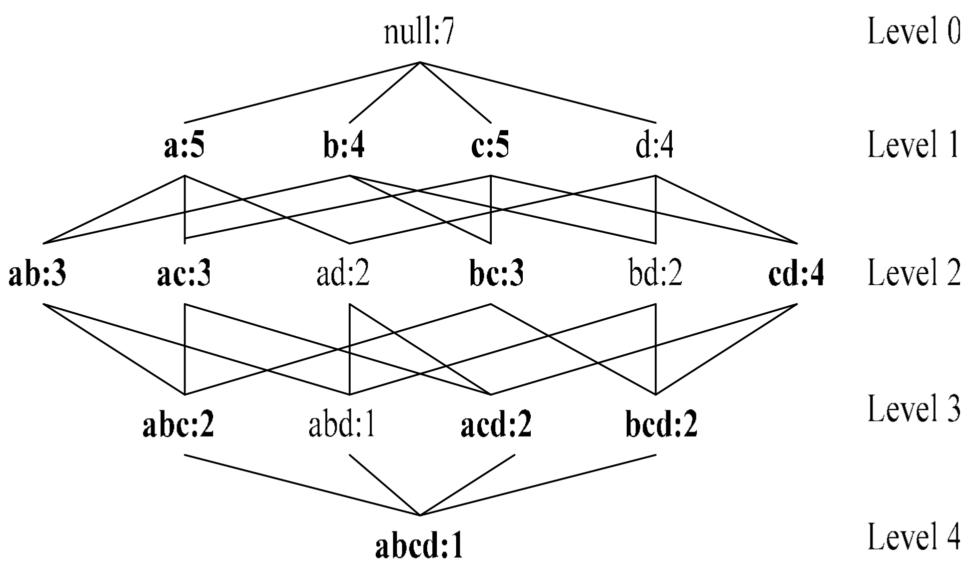
- Mine core patterns using

$$\min\sup' = \alpha \cdot \min\sup, 0 \le \alpha \le 1$$

- Build a lattice of the core patterns
- Traverse the lattice to compute the approximate itemsets

A Running Example

- Let the database be D, $\varepsilon_r = 0.5$, $\varepsilon_c = 0.5$, s=3, and $\alpha = \frac{1}{3}$



The Lattice of Core Patterns

	A	B	C	D
10	1	1	1	0
20	1	0	0	0
30	1	1	1	1
40	0	0	1	1
50	1	1	0	0
60	1	0	1	1
70	0	1	1	1

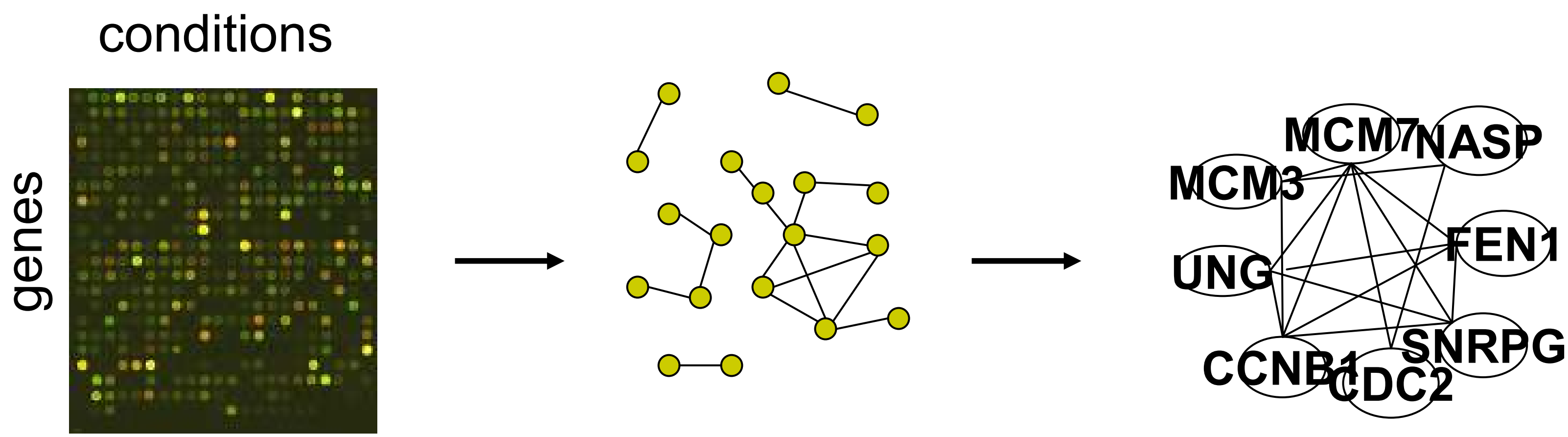
Database D

Microarray → Co-Expression Network

Microarray

Coexpression Network

Module

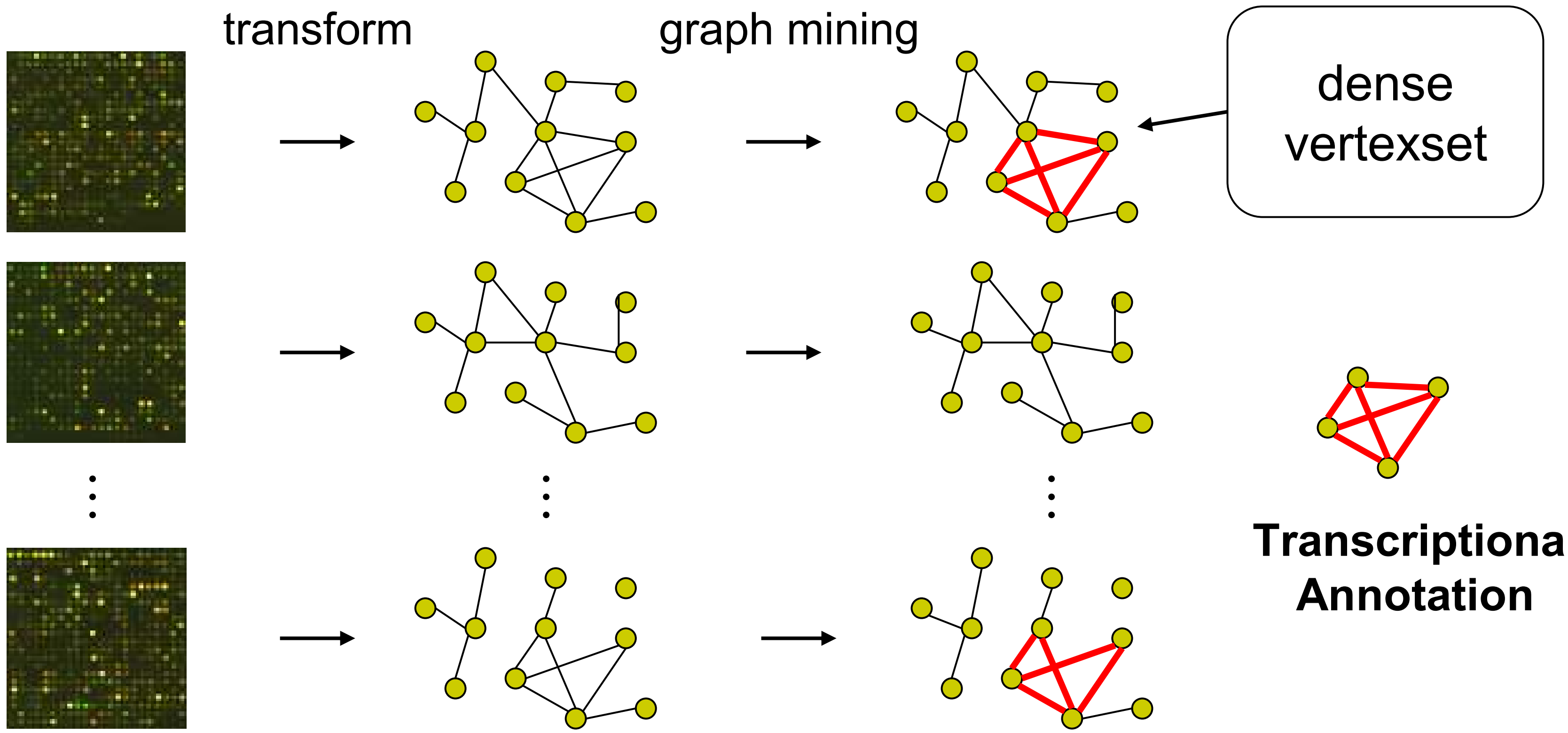


Two Issues:

- noise edges
- large scale

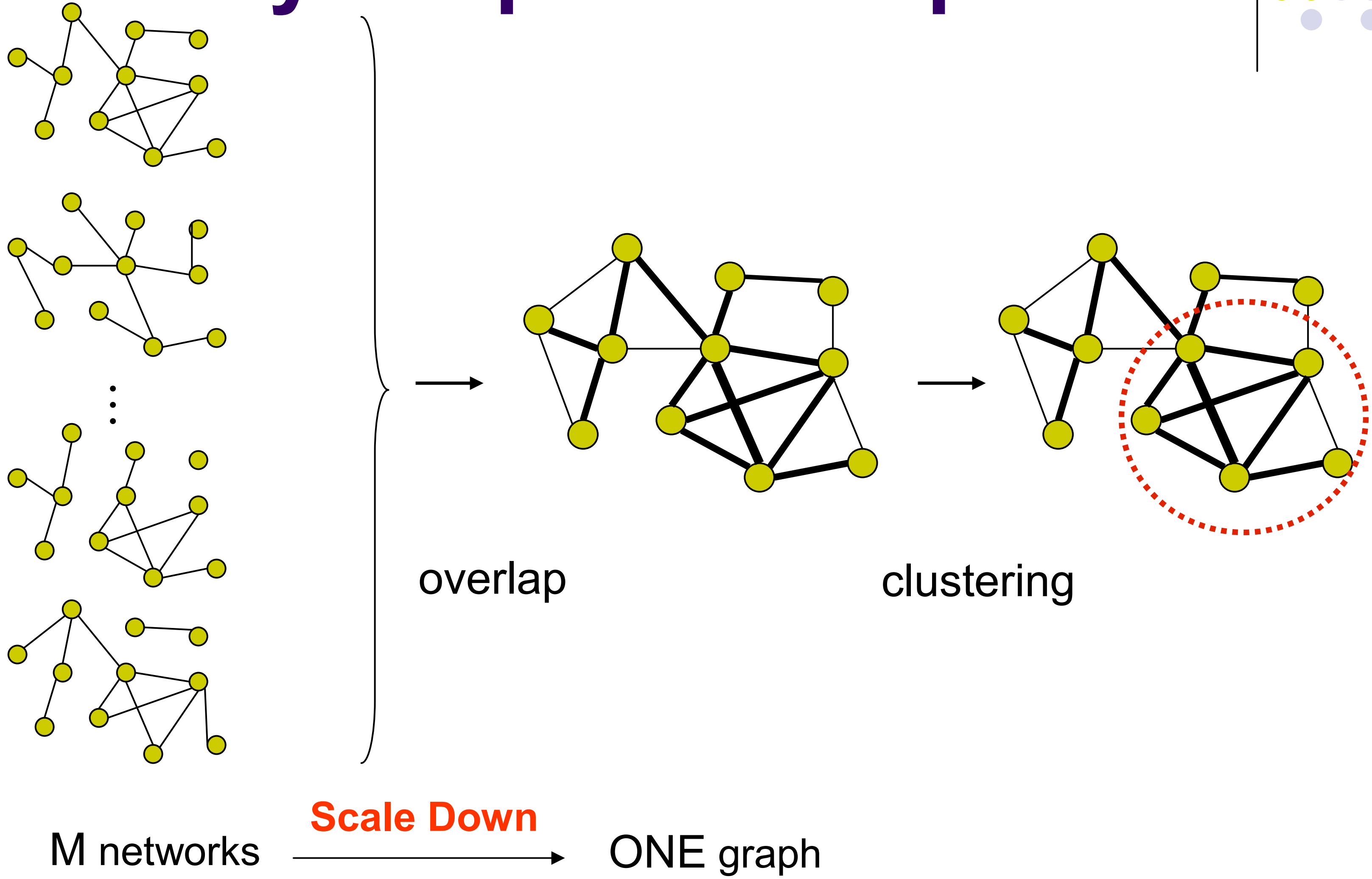
Mining Poor Quality Data

Patterns discovered in multiple graphs are more reliable and significant

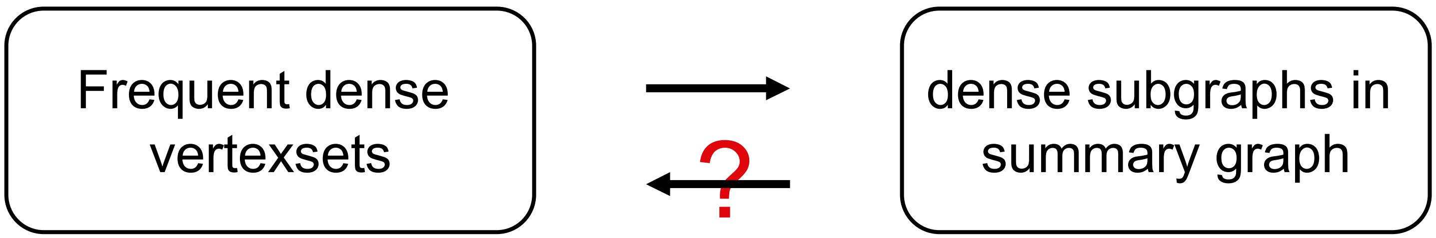


~9000 genes 105 x ~(9000 x 9000) = 8 billion edges

Summary Graph: Concept

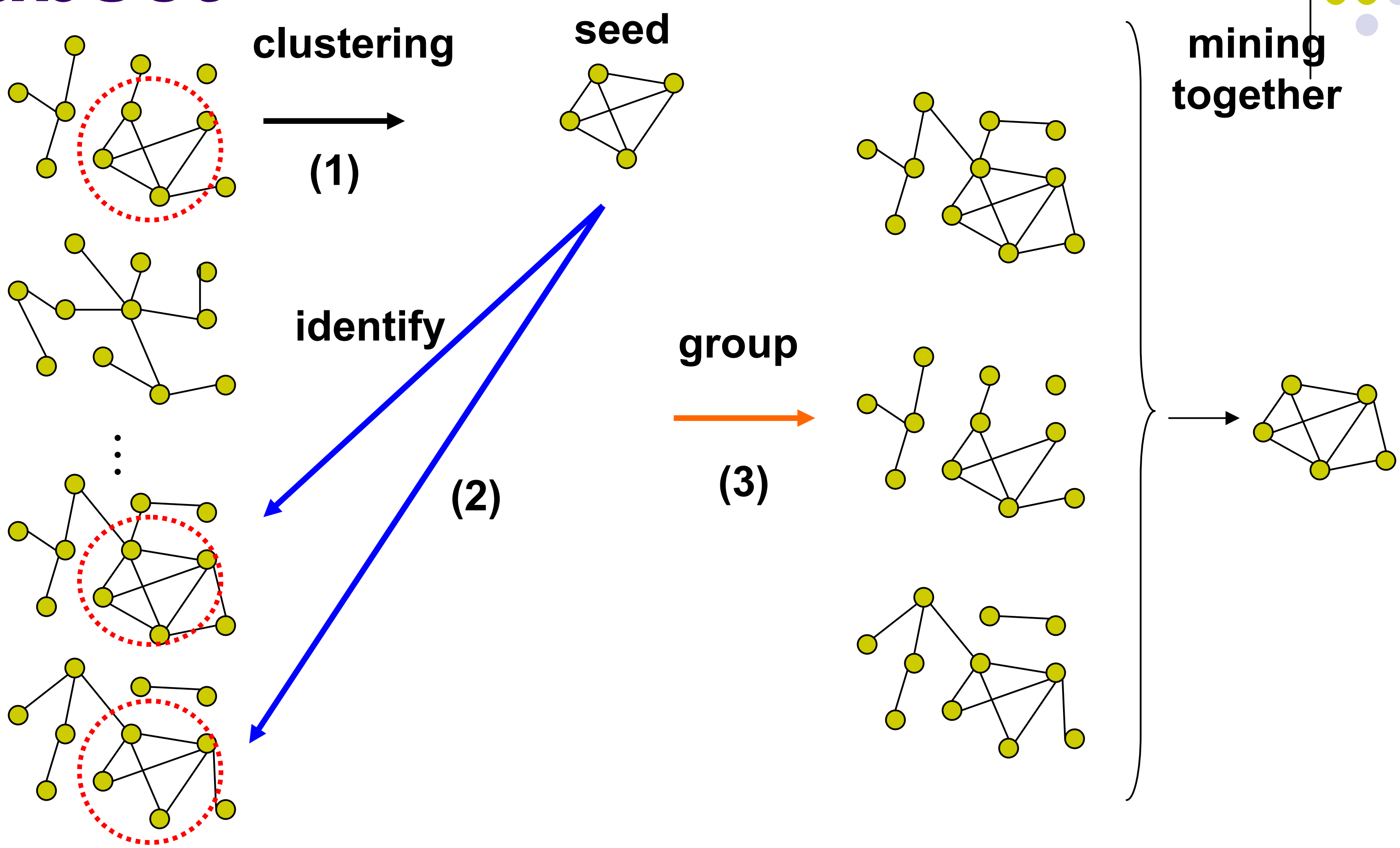


Summary Graph: Noise Edges



- Dense subgraphs are accidentally formed by noise edges
- They are false frequent dense vertexsets
- Noise edges will also interfere with true modules

Unsupervised Partition: Find a Subset



Frequent Approximate Substrinng

S_1 = ATCCGTACAGTTTCAGTTGCA

S_2 = ATCCGTACAGTTCAGTTGCA

S_3 = ATCTGCACAGGTCAGCAGCA

S_4 = ATCAGCACAGGTCAGGAGCA

ATCCGCACAGGTCAGT AGCA

Limitation on Mining Frequent Patterns: Mine Very Small Patterns!

- Can we mine large (i.e., colossal) patterns? — such as just size around 50 to 100? Unfortunately, not!
- Why not? — the curse of "downward closure" of frequent patterns
  - The "downward closure" property
    - Any sub-pattern of a frequent pattern is frequent.
  - Example. If $(a_1, a_2, ..., a_{100})$ is frequent, then $a_1, a_2, ..., a_{100}, (a_1, a_2), (a_1, a_3), ..., (a_1, a_{100}), (a_1, a_2, a_3), ...$ are all frequent! There are about 2^{100} such frequent itemsets!
  - No matter using breadth-first search (e.g., Apriori) or depth-first search (FPgrowth), we have to examine so many patterns
- Thus the downward closure property leads to explosion!

Do We Need Mining Colossal Patterns?

- From frequent patterns to closed patterns and maximal patterns
  - A frequent pattern is *closed* if and only if there exists no super-pattern that is both frequent and has the same support
  - A frequent pattern is *maximal* if and only if there exists no frequent super-pattern
- Closed/maximal patterns may partially alleviate the problem but not really solve it: We often need to mine scattered large patterns!
- Many real-world mining tasks needs mining colossal patterns
  - Micro-array analysis in bioinformatics (when support is low)
  - Biological sequence patterns
  - Biological/sociological/information graph pattern mining

Colossal Pattern Mining Philosophy

- *No hope for completeness*
  - If the mining of mid-sized patterns is explosive in size, there is no hope to find colossal patterns efficiently by insisting "complete set" mining philosophy
- *Jumping out of the swamp of the mid-sized results*
  - What we may develop is a philosophy that may jump out of the swamp of mid-sized results that are explosive in size and jump to reach colossal patterns
- *Striving for mining almost complete colossal patterns*
  - The key is to develop a mechanism that may quickly reach colossal patterns and discover most of them

Conclusions

- Most previous work focused on finding exact frequent patterns
- There exists a discrepancy between the exact model and some real world phenomenon due to
  - Noise, perturbation, etc
- Very long pattern mining can be another prohibiting problem
- Need to develop new methodologies to find approximate frequent patterns