Theory, Practice, and an Application of Frequent Pattern Space Maintenance

Limsoon Wong

(Joint work with Mengling Feng, Thanh-Son Ngo, Jinyan Li, Guimei Liu)

NUS
National University of Singapore

What Data?

Market Basket Dataset

Tid	Transactions
1	milk, bread, chips
2	beer, chips
3	beer
4	milk, bread

Patterns: {milk, bread} : 2
{beer} : 2
{milk, beer} : 0

Generic Dataset

Tid	Transactions
1	a, b, c, d
2	b, d
3	a, c, d
4	a, c

{a} : 3
{a, b} : 1
{a, b, c, d} : 1

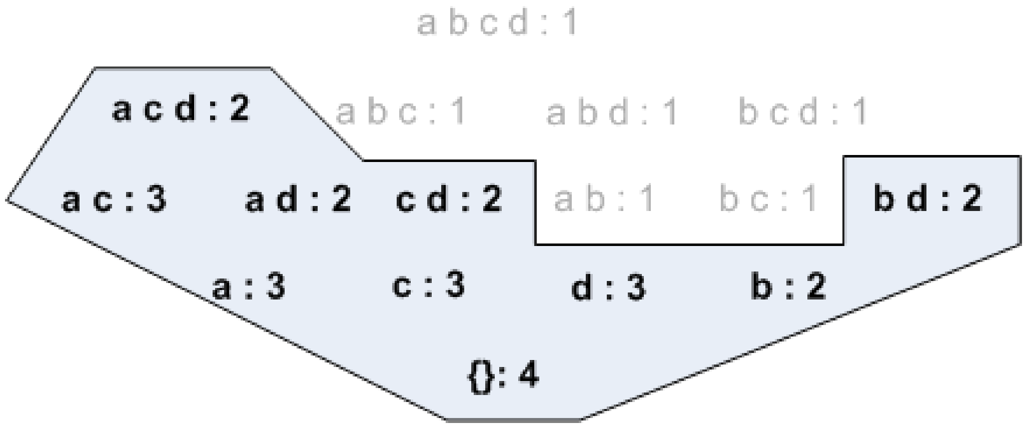
- **Transactional data**
  - Items, transactions, transaction ID, pattern, support of pattern

What Pattern?

Sample Dataset

$ms_{\%}$ = 50%, ms_a =2

Tid	Transactions
1	a, b, c, d
2	b, d
3	a, c, d
4	a, c

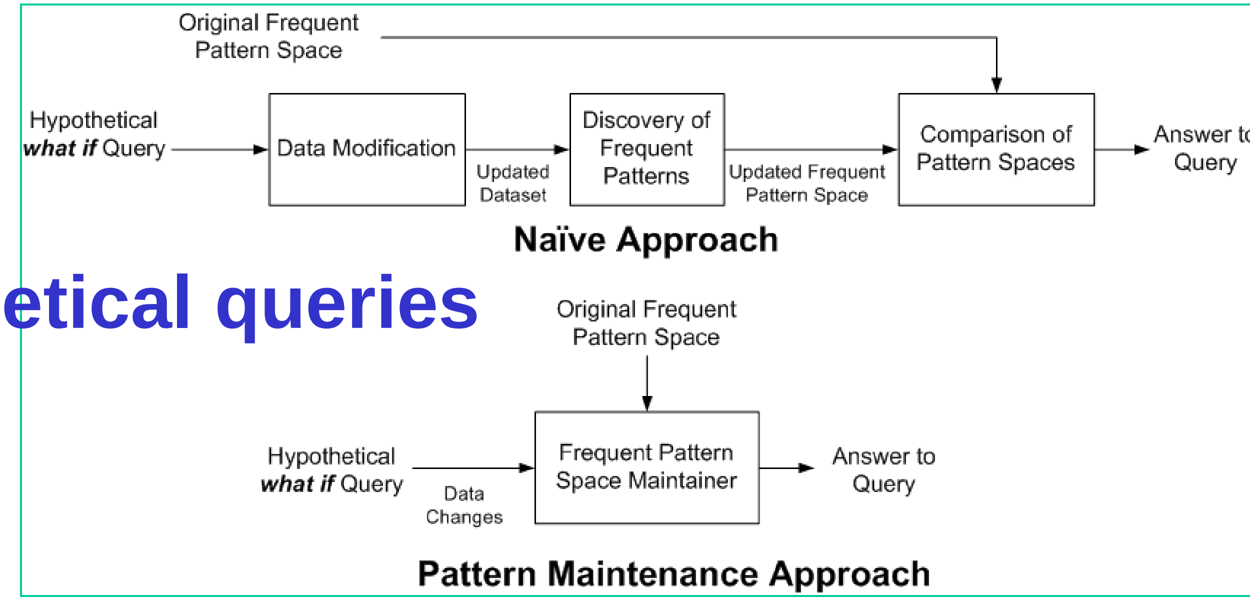


- **Freq patterns & space of freq patterns**
  - Minimum support threshold
  - ms_a or $ms_{\%}$ ($ms_a = \text{ceil}(ms_{\%} \times |D|)$)
  - Huge: 2^n ($2^{100} \approx 1.3$ E 30)

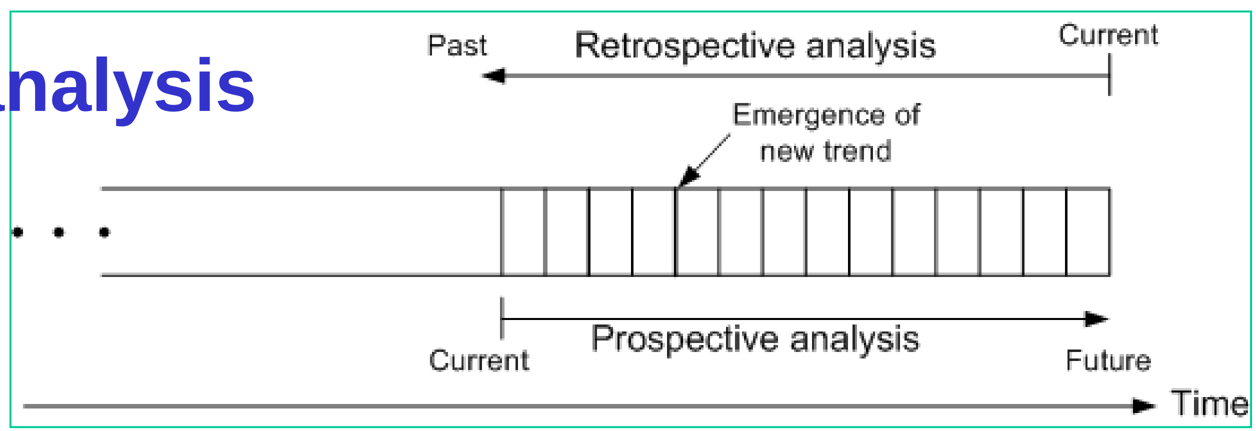
What Updates?

- **Incremental updates**
- **Decremental updates**
- **Support threshold adjustment**

Motivation



- **Hypothetical queries**



- **Trend analysis**

Challenges

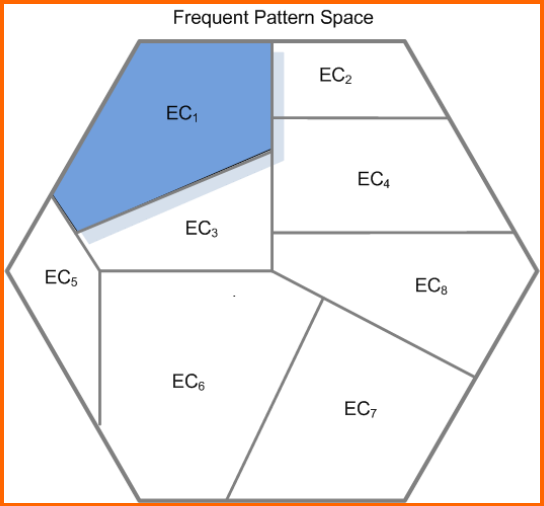
- **# of existing freq patterns is large**
  - Naïve maintenance: O(NFP $\times$ m)
    - **NFP, # of freq patterns (upper bound 2^n)**
    - **m, # of updated transactions**
- **# of "new" freq pattern candidates is large**
  - 2^n – NFP ($\approx 2^n$)

- **Existing approaches: Extension of certain pattern discovery algo / data structure they used**
- **What is missing?**
  - How freq pattern space evolves
  - A theoretical framework

Outline

- **Pattern space evolution**
- **TRUM: A decremental maintainer**
- **PSM: A complete maintainer**
- **Optimizing performance of PCL Classifier**

Pattern Space Evolution

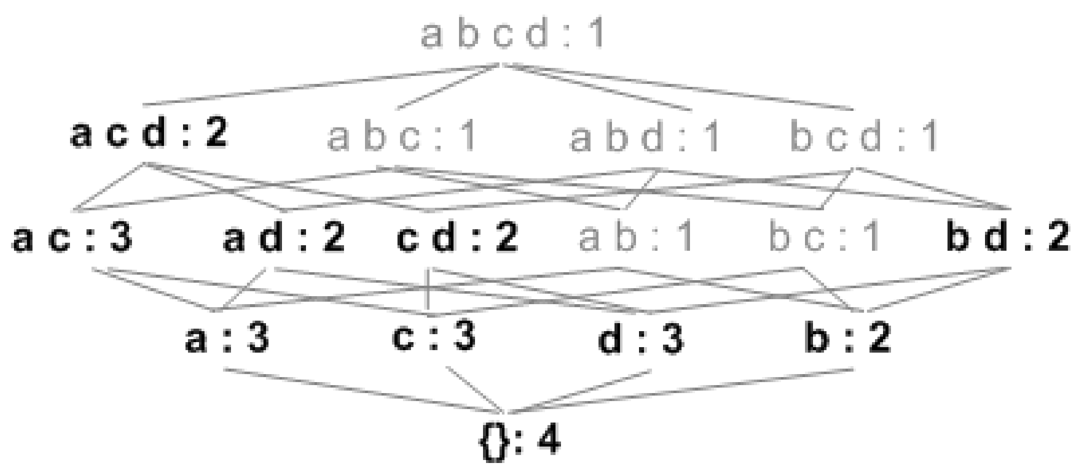


Basic Property of Pattern Space

Sample Dataset

$ms_{\%} = 50\%$, $ms_a = 2$

Tid	Transactions
1	a, b, c, d
2	b, d
3	a, c, d
4	a, c



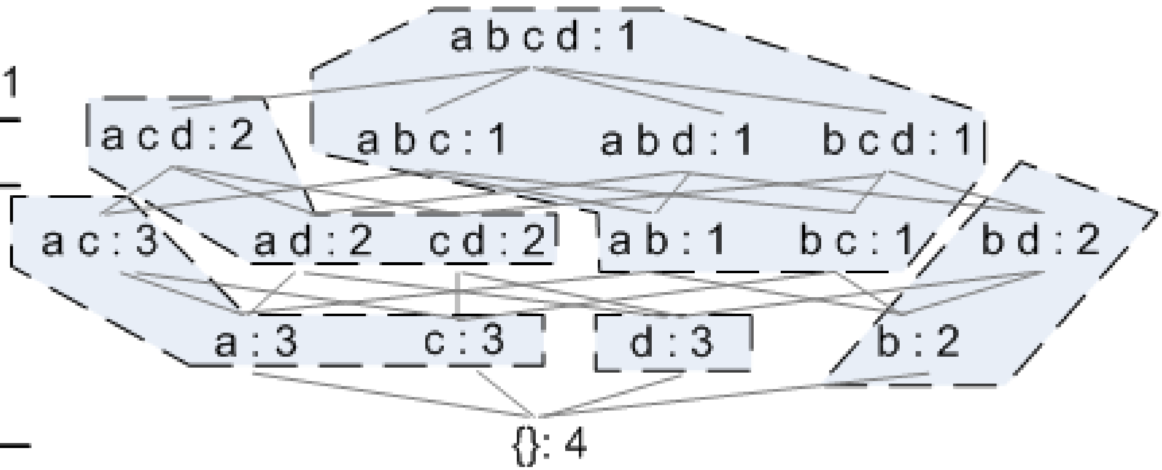
- **Anti-monotone property**
  - If P is freq, all subset of P is freq
  - If P is infreq, all superset of P is infreq

Decomposition into Equiv Classes

Sample Dataset

$ms_{\%} = 20\%$, $ms_a = 1$

Tid	Transactions
1	a, b, c, d
2	b, d
3	a, c, d
4	a, c

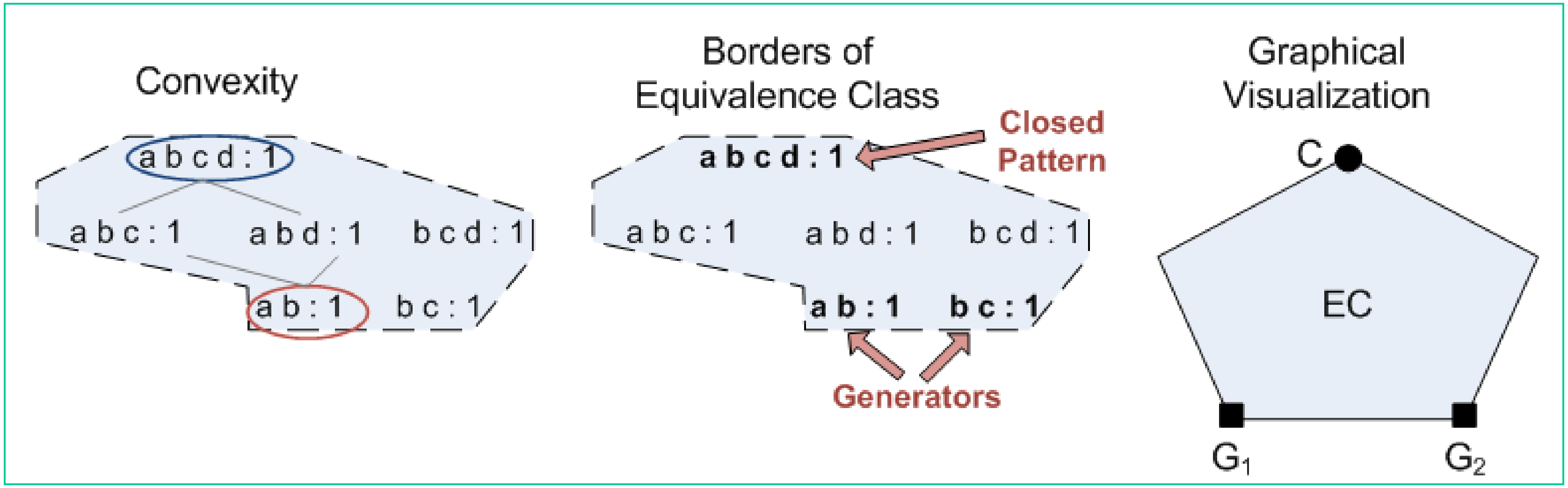


- **Equiv Class: A set (class) of patterns that appear in exactly the same transactions**

Equivalence Class

- **Equiv classes are convex**

$\Rightarrow$ **Can be compactly represented by borders**
  - A unique closed pattern (most specific pattern)
  - A set of generators (most general patterns)

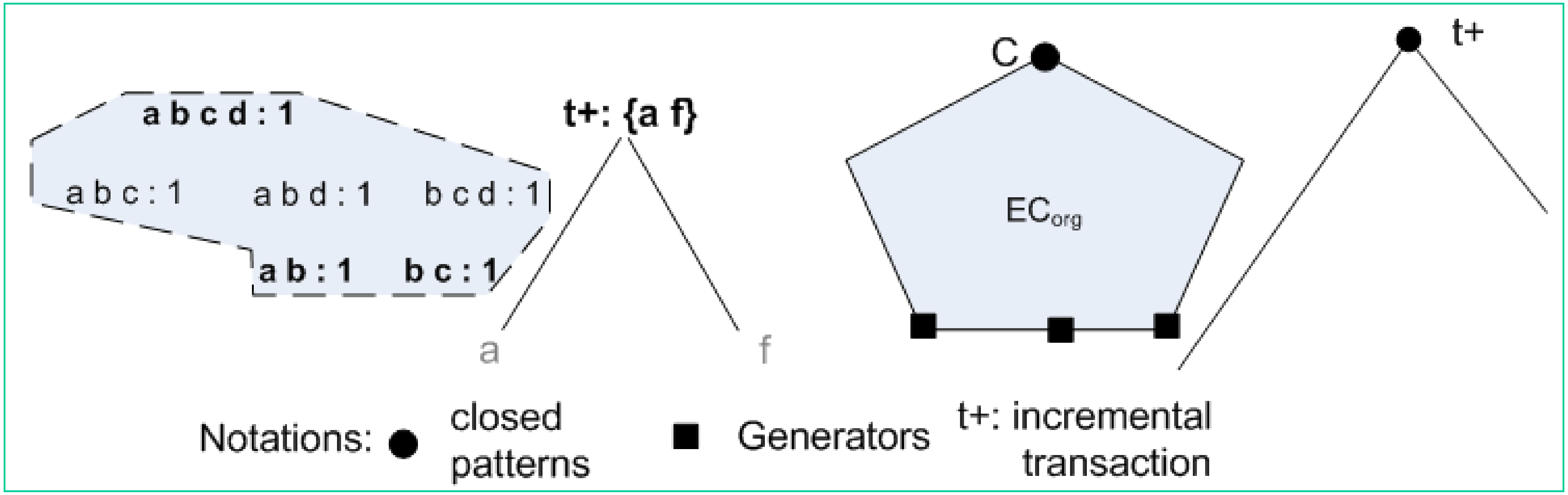


Pattern Space Evolution = Equiv Class Evolution

- **Pattern Space Maintenance**
  = Equiv Class Maintenance

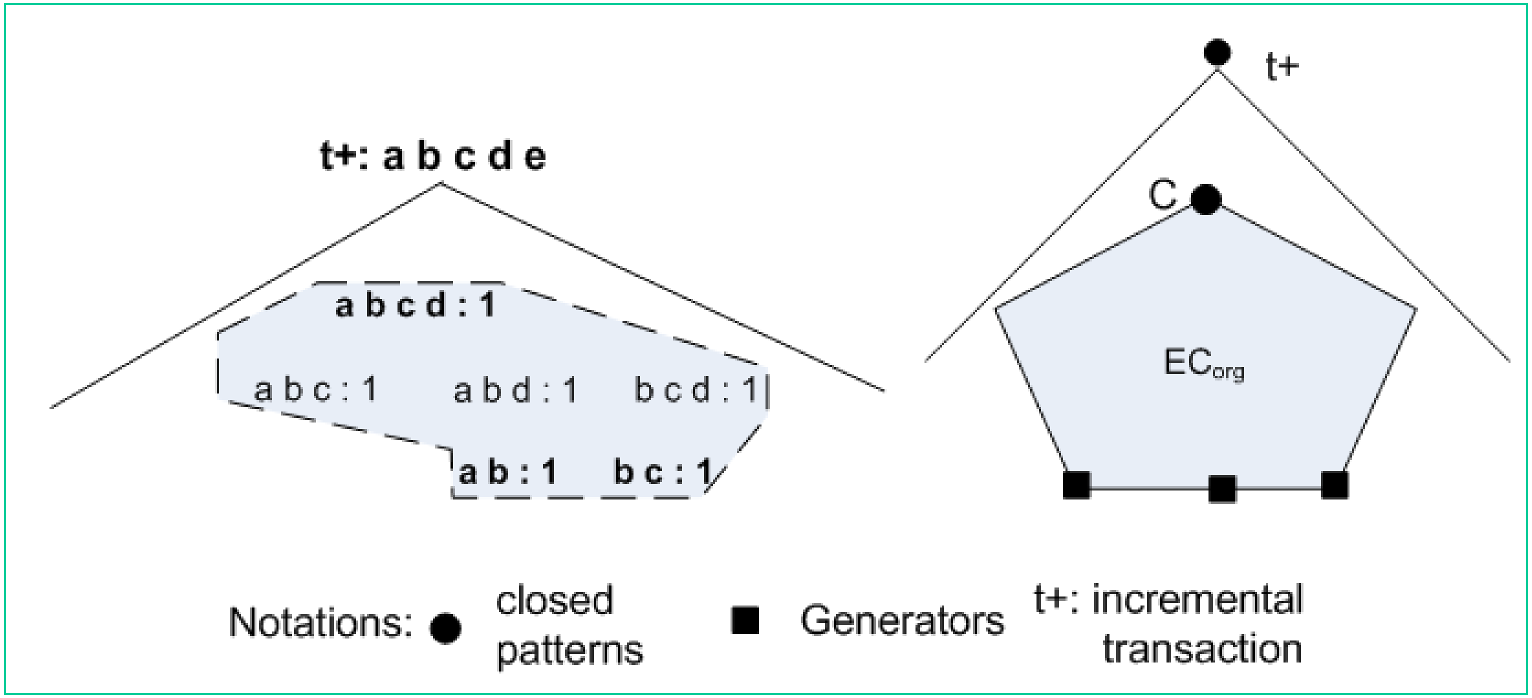
- **Equiv Class Maintenance**
  = Border Maintenance

Equiv Class Evolution Incremental Updates: Case 1



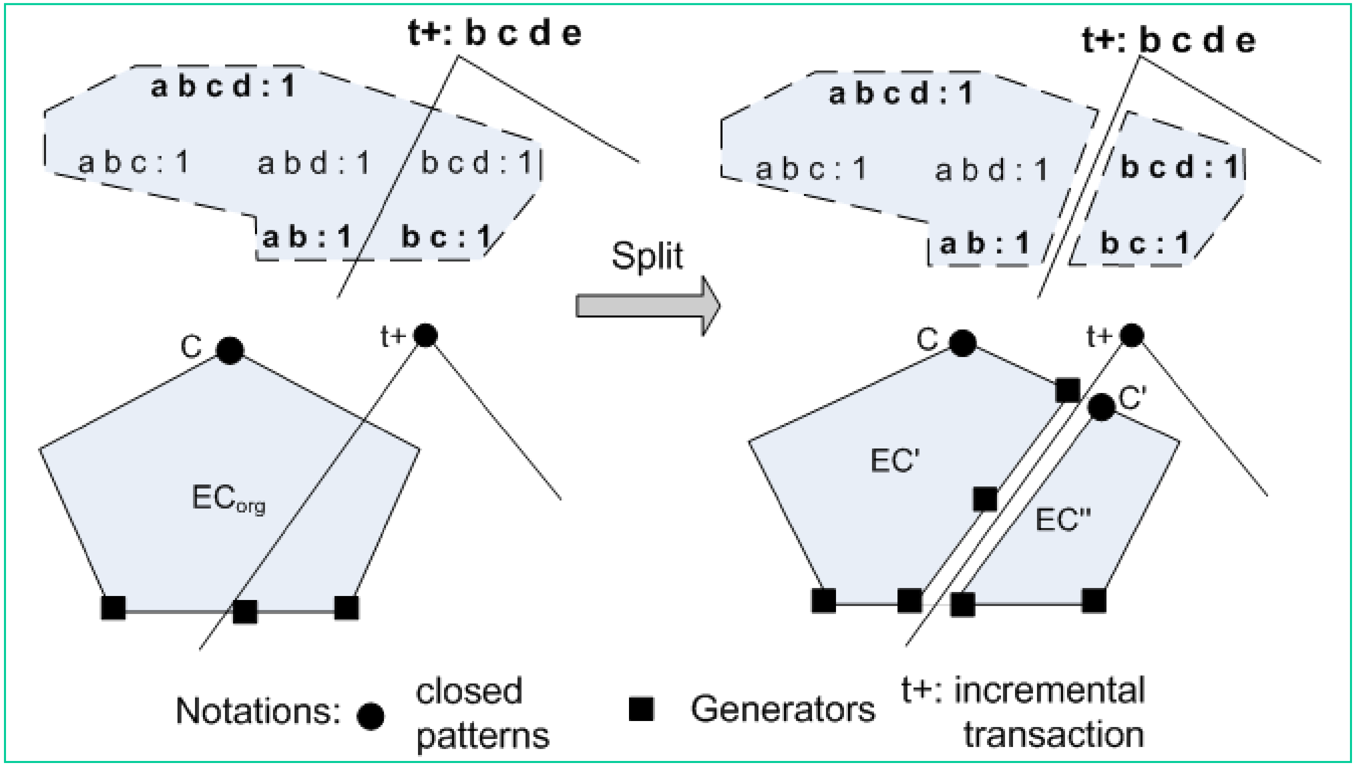
- **No structural change and no change to support**
- **Condition: $\forall$ G $\not\subset$ t+**

Equiv Class Evolution
Incremental Updates: Case 2



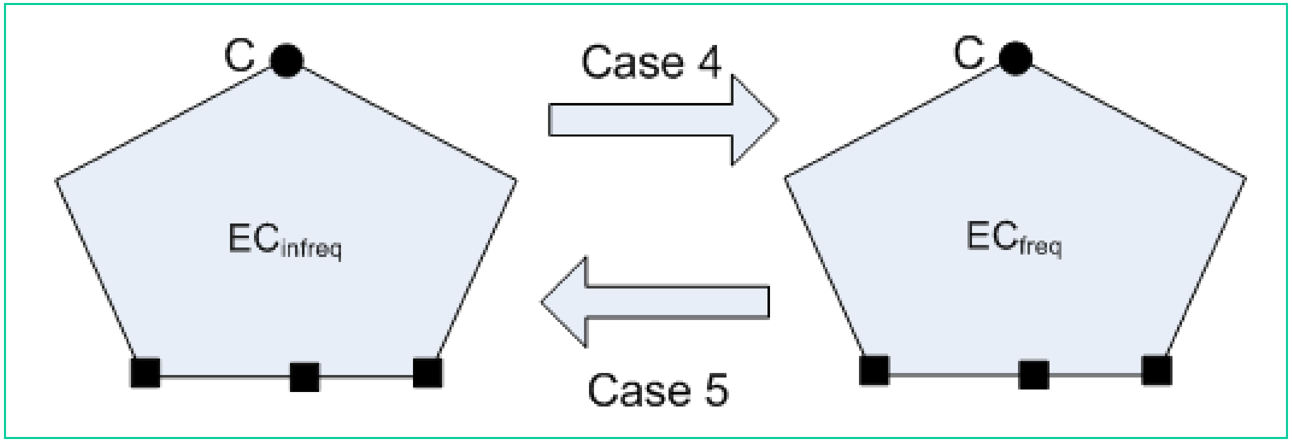
- **Structurally unchanged but increased in support**
- **Condition: C $\subseteq$ t+**

Equiv Class Evolution
Incremental Updates: Case 3



- **Split into two**
- **Condition: C $\not\subseteq$ t+, but $\exists$ G $\subseteq$ t+**

Equiv Class Evolution
Incremental Updates: Case 4 & 5



- **Case 4: Emerge to be NEW freq equiv class**
- **Case 5: Become infreq**
  - $ms_a = \text{ceil}(ms_{\%} \times |D|)$

  $\therefore D_{inc}\text{-->}D \Rightarrow |D| \uparrow \Rightarrow (ms_{\%} \times |D|) \uparrow \Rightarrow ms_a \uparrow$

Key Incremental Maintenance Tasks

- **Support update**
  - $O(N_{EC} \times m)$
- **Class splitting**
  - $O(N_{EC} \times m)$
- **New class discovery**
  - $O(2^n - N_{FP})$
- **Obsolete class removal**
  - $O(N_{EC})$

PSM+ does these tasks efficiently

Equiv Class Evolution: Decremental Updates

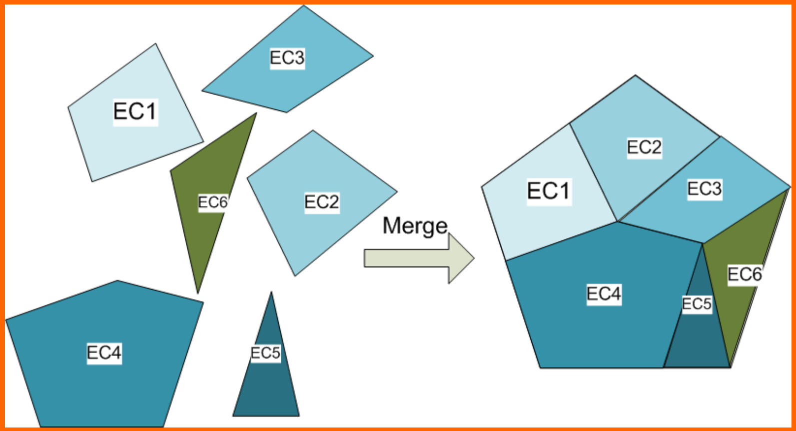
- **Incremental**
  - No change
  - ↑ in support
  - Split up
  - Emerge as freq class due ↑ in support
  - Become infreq due to ↑ in ms_a
- **Decremental**
  - No change
  - ↓ in support
  - Merge w/ other class
  - Become infreq due to ↓ in support
  - Emerge to be freq class due to ↓ in ms_a

Key Decremental Maintenance Tasks

- **Update support**
- **Merge Class**
- **Discover new freq class**
- **Remove obsolete class**

PSM– and TRUM do these tasks efficiently

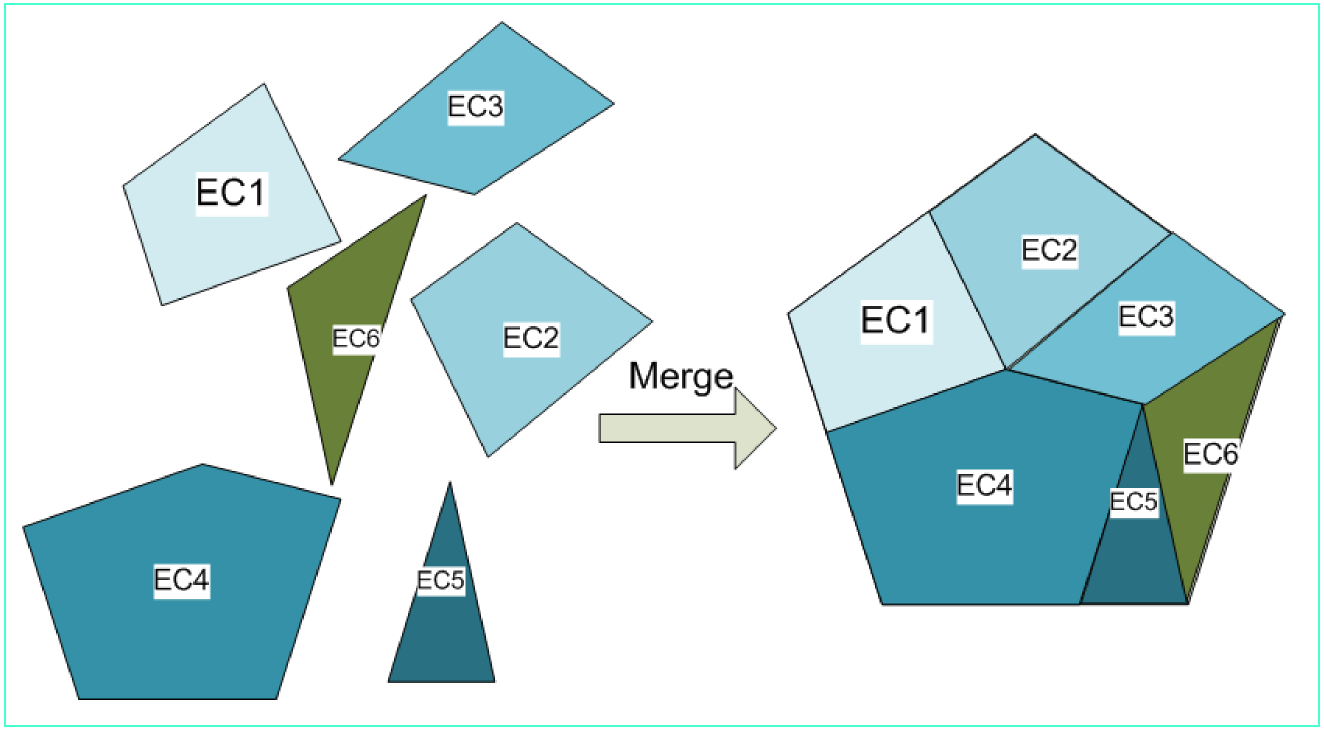
Transaction Removal Update Maintainer, TRUM



National University of Singapore

TRUM

- **A decremental maintainer**
- **Major challenge: Merging of classes**



Transaction ID-tree (Tid-tree)

Original Dataset
(ms_a =2)

TID	Transactions
1	a, b, c, d
2	b, d
3	a, c, d
4	a, c
5	b

Discovery of Equivalence Classes ⇒

Frequent equivalence classes:

EC_1: { {a}, {c}, {a, c} } : ***1, 3, 4*** (3)

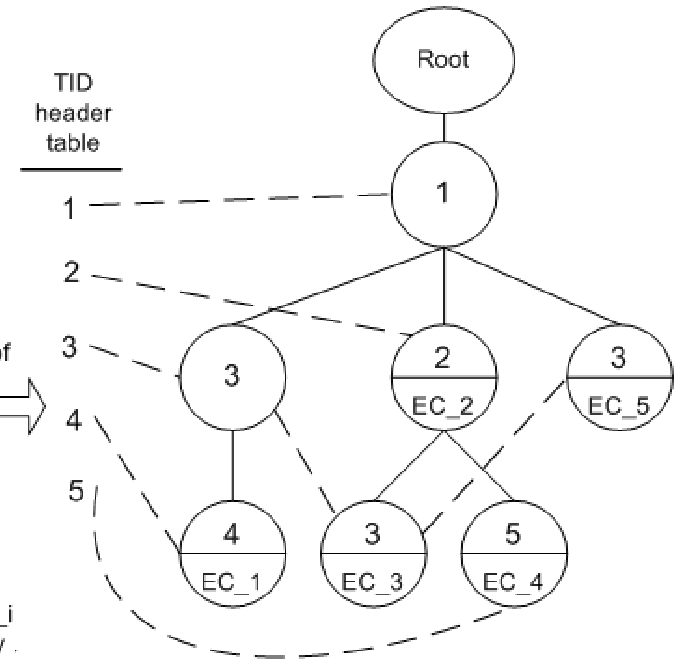
EC_2: { {b, d} } : ***1, 2*** (2)

EC_3: { {d} } : ***1, 2, 3*** (3)

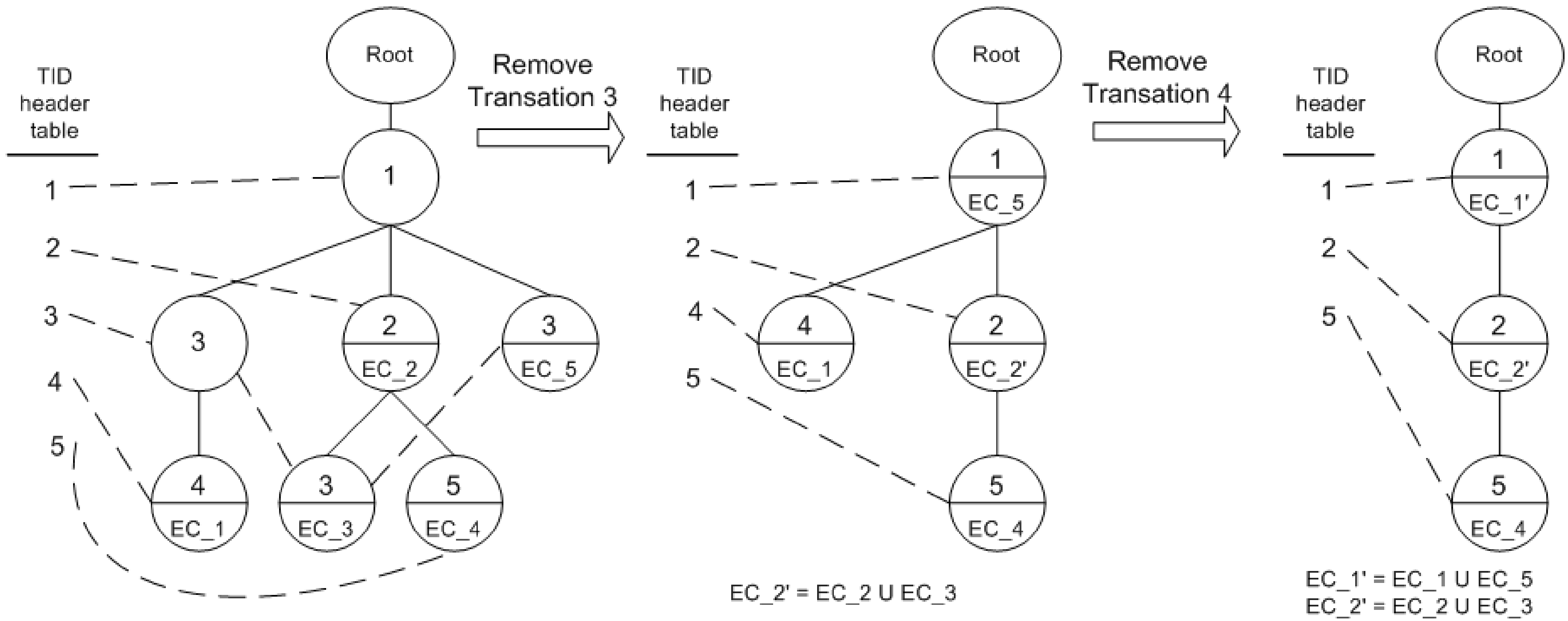
EC_4: { {b} } : ***1, 2, 5*** (3)

EC_5: { {a, d}, {c, d}, {a, c, d} } : ***1, 3*** (2)

Construction of *TID-tree* ⇒



Notation: EC_i: {.} : x (y) refers to an equivalence class EC_i, where EC_i consists patterns {.}, the TID-list of EC_i is ***x***, and the support of EC_i is y .

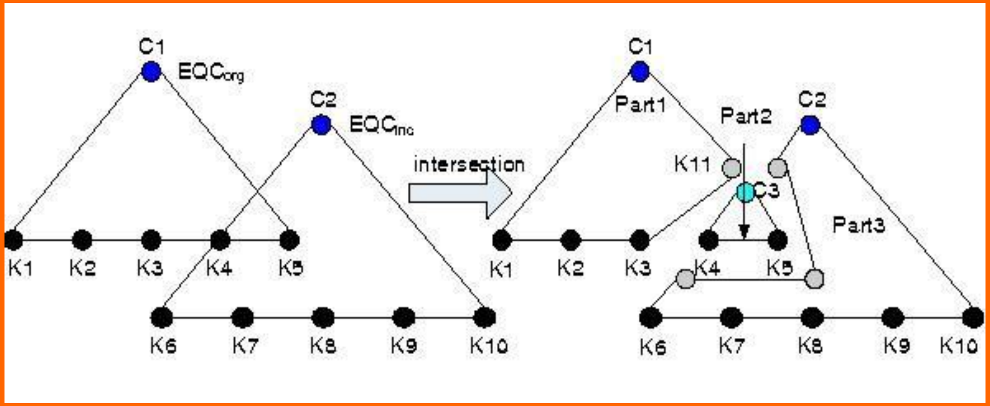


Fast Decremental Maintenance on Tid-tree

Performance: Speed Up

Dataset	Discovery Algorithms		Maintenance Algorithms		
	FP-growth*	GC-growth	Borders	ZIGZAG	moment
chess ms_a = 1.5k	130	13	1,980	28	10,600
connect ms_a = 30k	24	1.5	2,400	10	182
mushroom ms_a = 500	1,240	31	6,500	486	10,700
retail ms_a = 100	58	306	48	818	208
t10i4d100k ms_a = 500	64	113	66	90	1,288
average	**119**	**80**	**2,268**	**174**	**2,848**

Pattern Space Maintainer, PSM



National University of Singapore

PSM: A Complete Maintainer

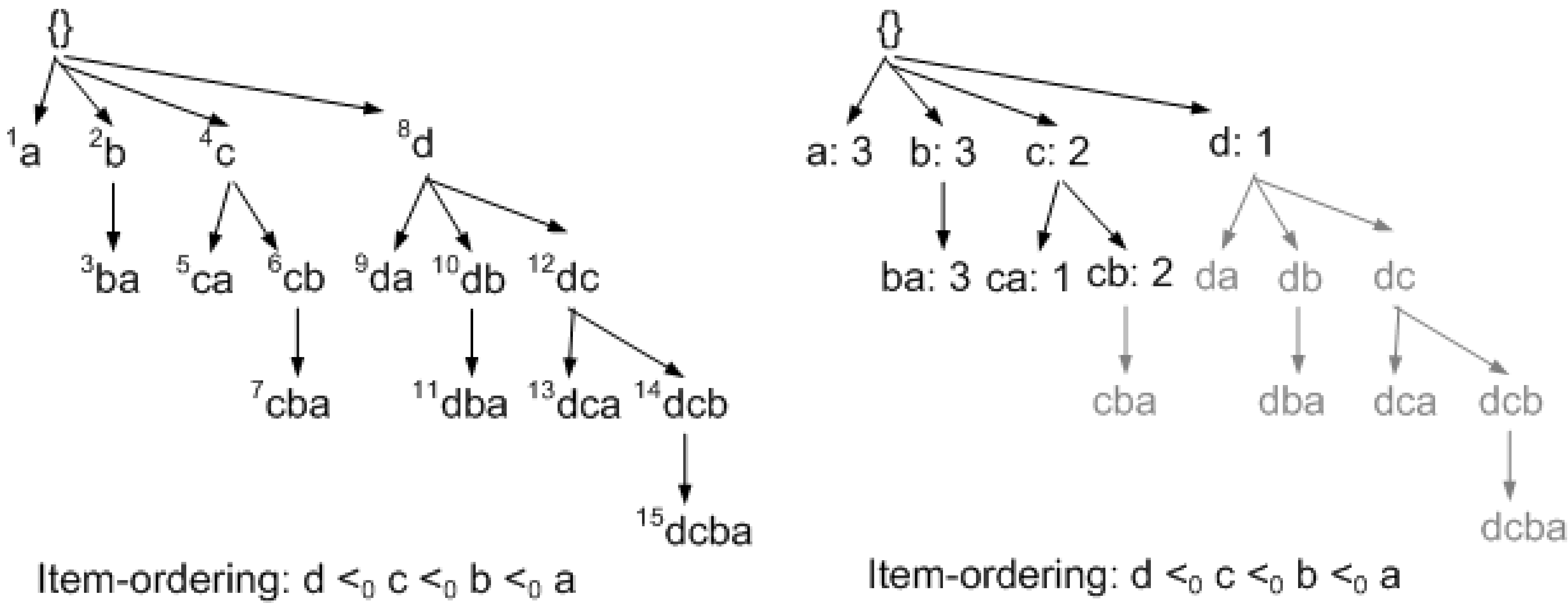
- **Incremental: PSM+**
- **Decremental: PSM-**
- **Threshold adjustment: PSMΔ**

PSM+

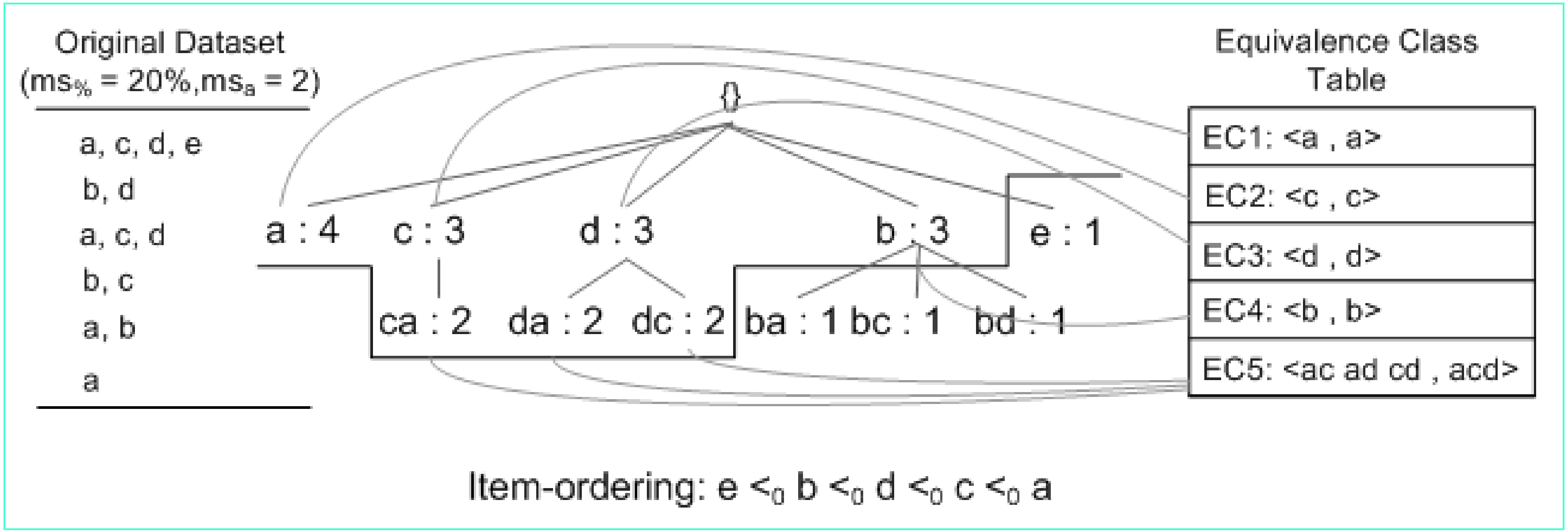
- **Key idea**
  - Only update those who need to be updated
  - $O(N_{affectedEC})$
- **Solution**
  - Generator Enumeration tree (GE-tree)

- **Key tasks**
  - Support update
  - Class splitting
  - New class discovery
  - Obsolete class removal

Set-Enumeration Tree

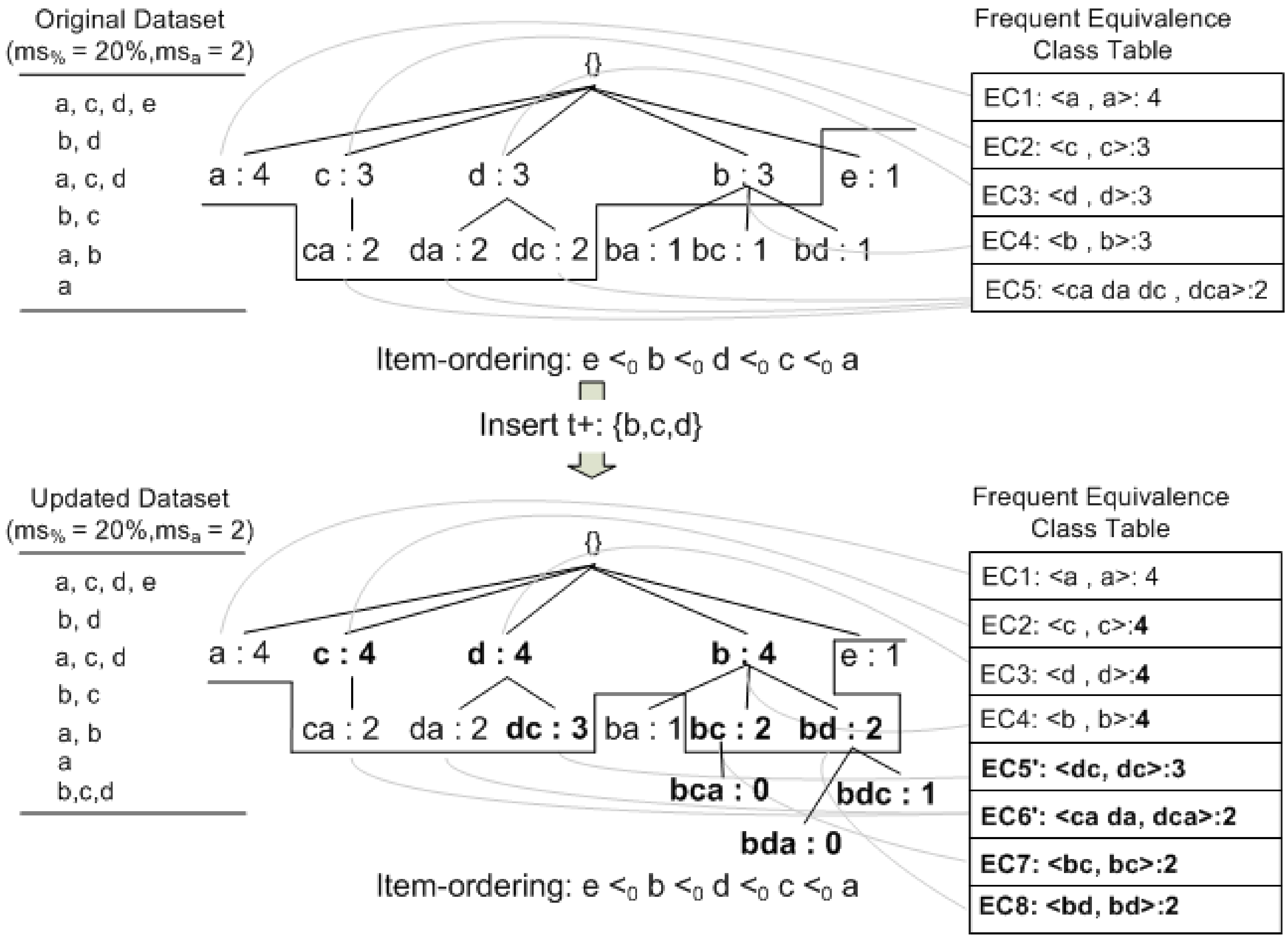


Generator-Enumeration Tree



- **Key features**
  - Generators only
  - Link to corresponding equiv class
  - Negative border generators

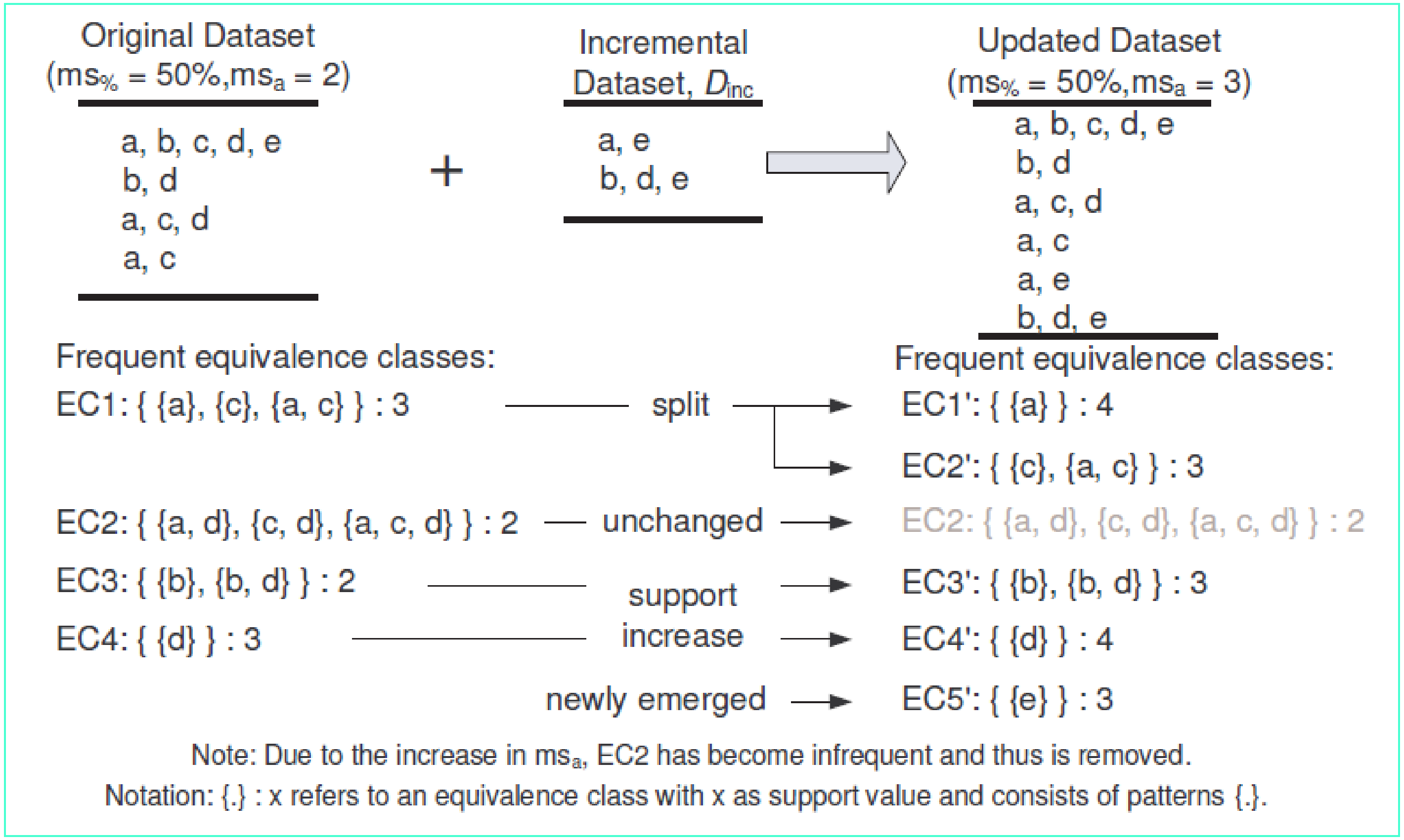
Update of GE-tree



PSM+: Speed Up

Dataset	Discovery Algorithms		Maintenance Algorithms			
	FP-growth*	GC-growth	Borders	CanTree	ZIGZAG	moment
chess $ms_{\%}$=50%	590	96	3,400	620	1,395	13,000
connect $ms_{\%}$=50%	2280	8.2	5200	2340	1400	826
mushroom $ms_{\%}$=0.1%	3085	380	6700	3121	47800	3216
retail $ms_{\%}$=0.1%	640	247	36000	735	27100	18210
t10i4d100k $ms_{\%}$=0.5%	150	374	1540	200	261	609
average	**672**	**262**	**12800**	**746**	**7067**	**5878**

Conclusions



- **Analysis of evolution of freq pattern space**
- **TRUM, efficient decremental maintenance**
- **PSM, efficient complete maintenance**

Efficiently Finding Best Parameter for Rule-Based Classifier PCL

Emerging Patterns

- **Freq patterns: Set of items appearing in many records in the dataset**
- **Jumping emerging patterns (JEP): Patterns freq in one class but absent in other classes**
- **JEPs capture characteristics of the class that distinguish them from other classes**
- **App in classification: JEPs are used to make predictions**

Assoc Rule-Based Classification

- **A set of rules is constructed from data**
- **Class labels of test instances are determined by these rules**

- **3 main types of rule-based classifiers**
  - Best pattern is used to make prediction
  - A set of patterns is used to make prediction
  - A set of patterns is used as the best features and then a normal classifier is then trained on these features

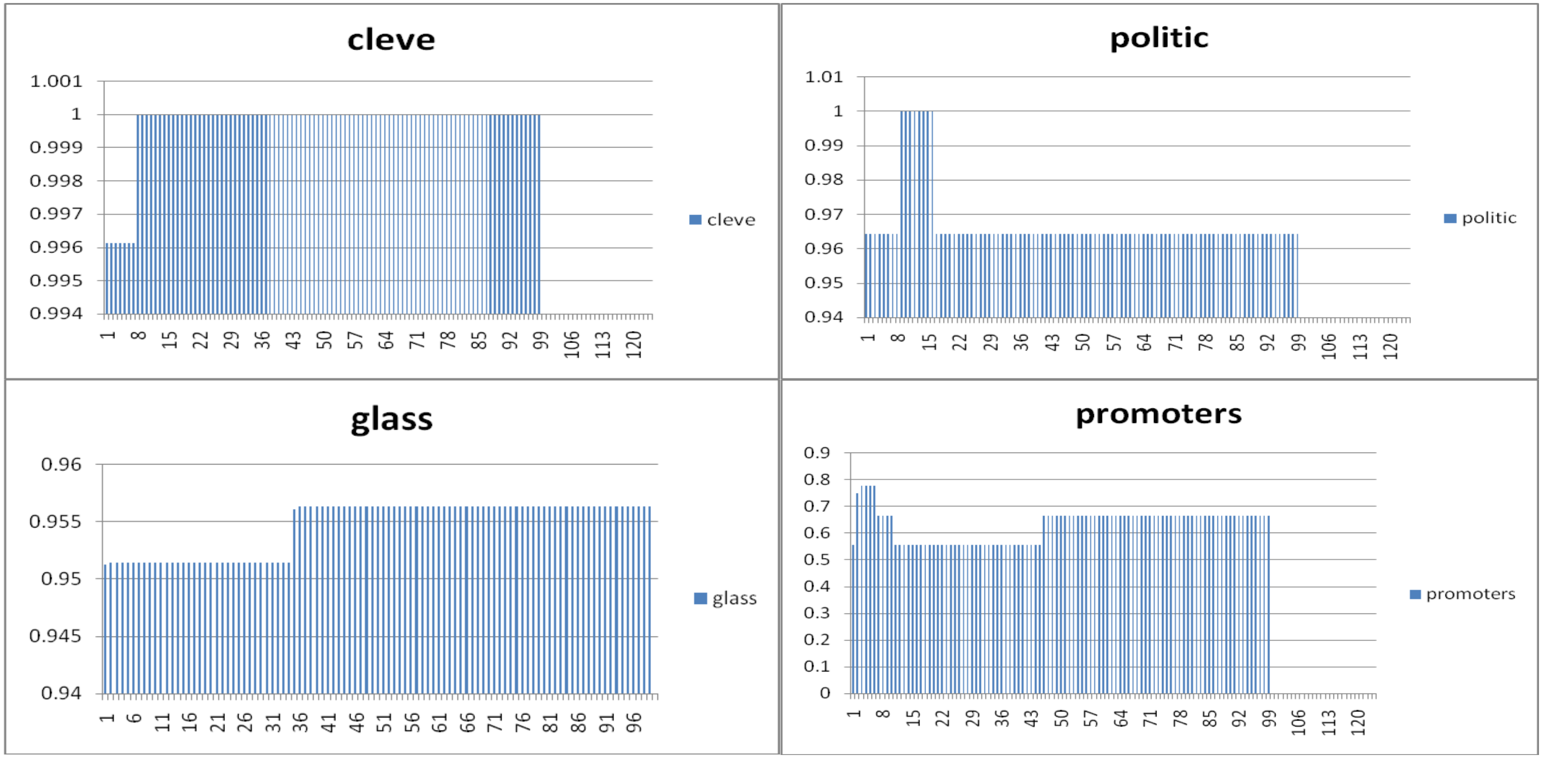
PCL

- **Construct rules from JEPs**
- **Given training set, collect JEPs for each class**
- **Given test instance, score for each class is sum of support of top K JEPs that cover the test instance divided by the sum of support of top K JEPs in this class**

$$Score(t,+) = \frac{\sum_{i=1}^{k} \sup(EP_{t_i}^{+}, D)}{\sum_{i=1}^{k} \sup(EP_i^{+}, D)} \qquad Score(t,-) = \frac{\sum_{i=1}^{k} \sup(EP_{t_i}^{-}, D)}{\sum_{i=1}^{k} \sup(EP_i^{-}, D)}$$

- **Class whose score is higher wins**
  - Value of K affects prediction results

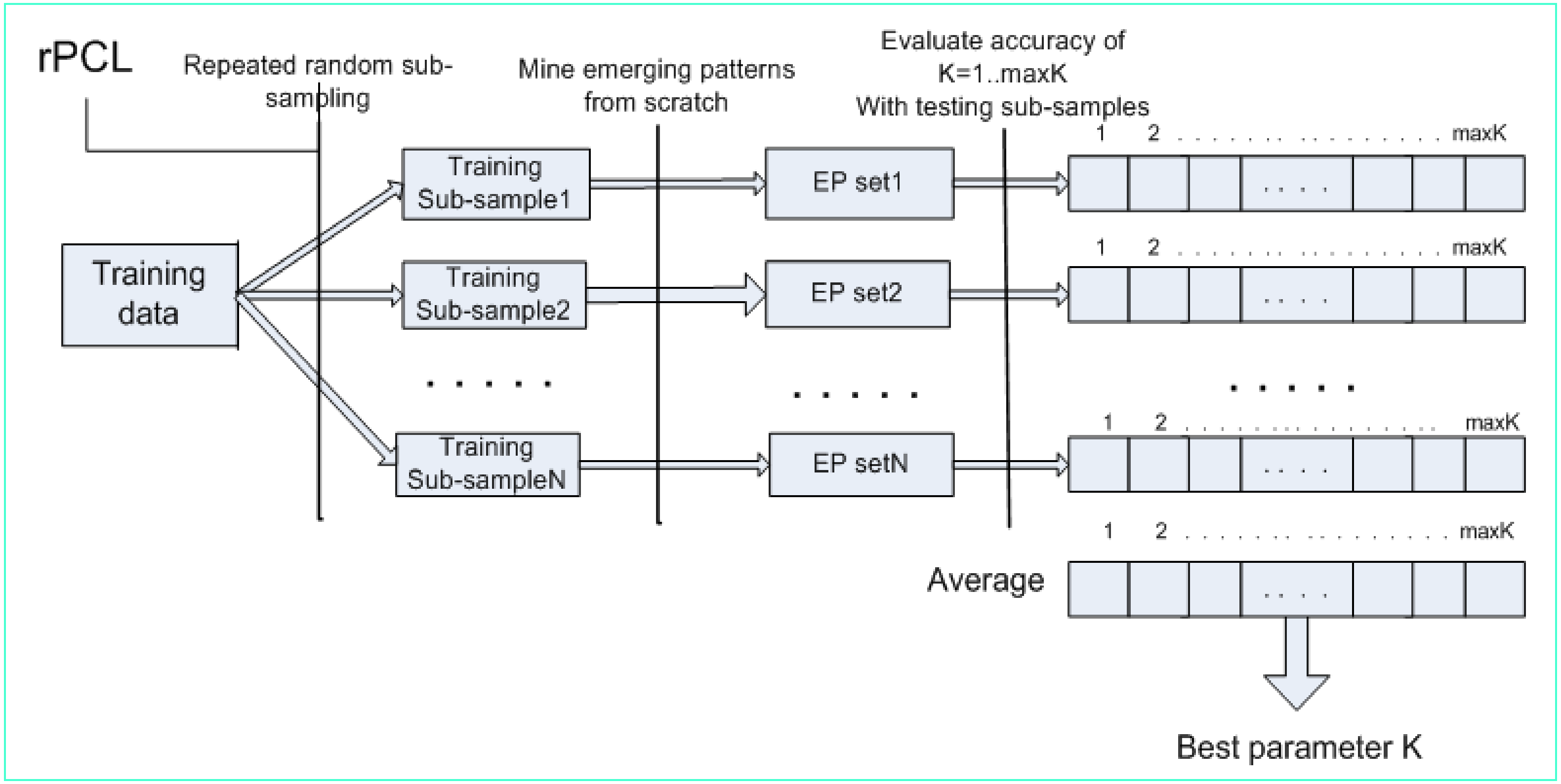
Value of K Affects Prediction



- **K too small: Lose power of small-support JEPs**
- **K too big: Suffer over-fitting from too many JEPs**

⇒ How to choose K ?

rPCL: Optimize Parameter by CLT



- **Simulate proc of classification in a training set on each K**
- **Select K that gives best estimated performance**
- **Correctness is guaranteed by CLT**

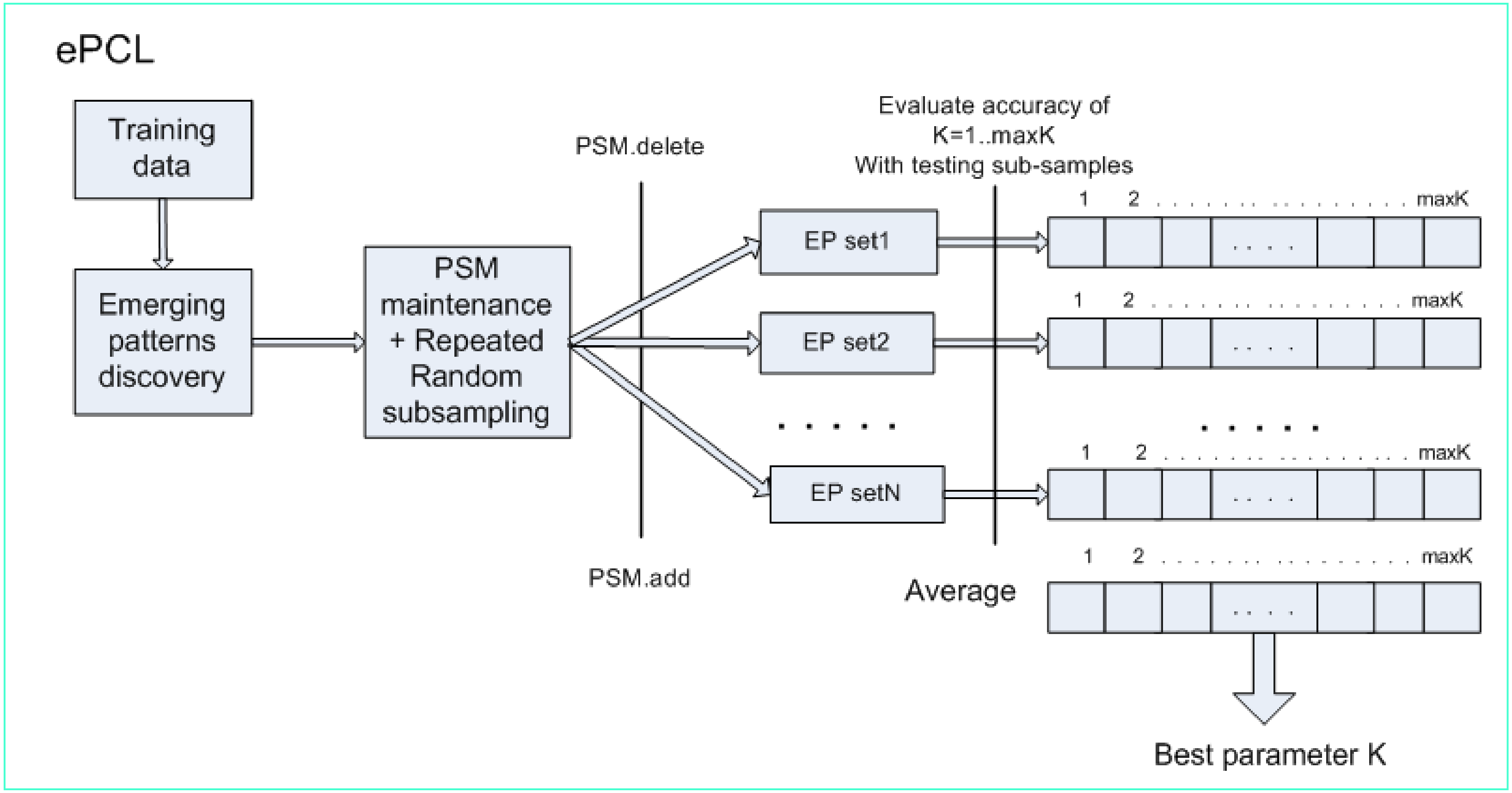
Pattern Space Maintenance

- **Pattern space is set of freq patterns in a data set**
- **Small change in data set unlikely to cause big change in pattern space**

	Original	After removal
Dataset	abc abd ade ade	 abd ade ade
Frequent patterns	a, b, d, e, ab, ad, ae, de ade	a, d, e, ad, ae, de, ade

- **Pattern space maintained efficiently by PSM algo**

ePCL: Use PSM to improve rPCL



- **Maintain freq JEPs using PSM**

$\Rightarrow$ **PCL can be constructed fast from one sampling to others**

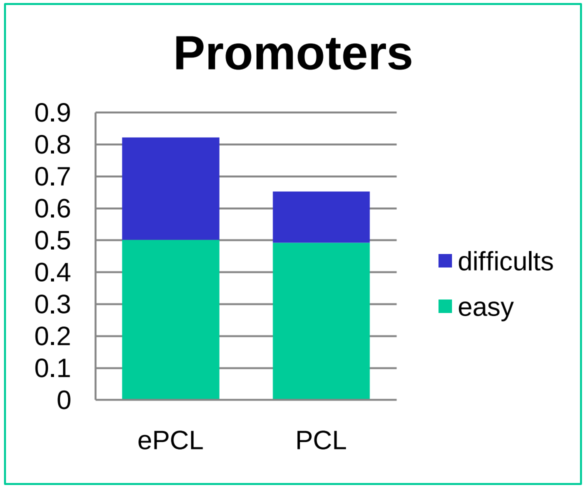
Accuracy Improved in Most Cases

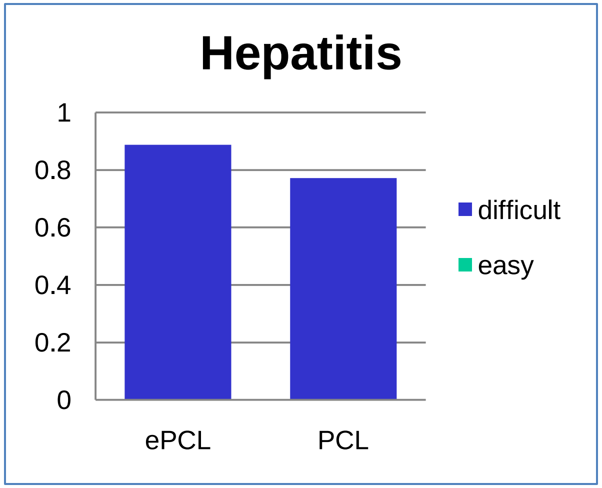
	ePCL	PCL	Improvement (%)
Promoters	0.82	0.65	25.92
Hepatitis	0.89	0.77	14.98
Wine	0.98	0.88	11.39

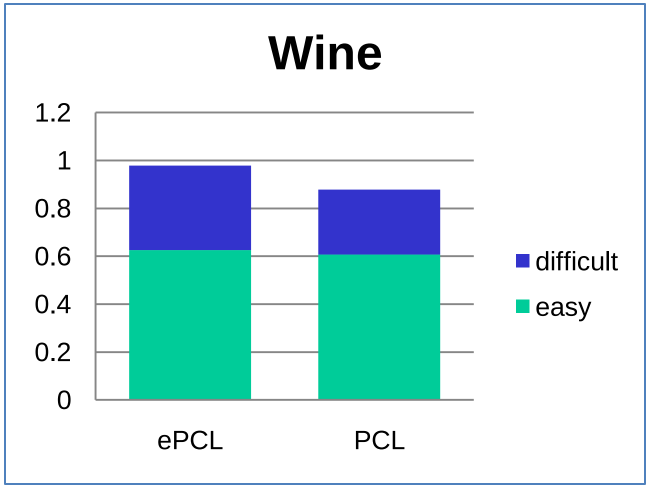
- **E.g., promoters, hepatitis, & wine datasets improved by 26% , 15%, & 11% respectively**

Difficult Cases

- **Improvement in difficult cases is more significant. Difficult cases are cases when scores for both classes are non-zero**







Efficiency

- **ePCL & rPCL same results but ePCL is lots faster**
- **ePCL slower than PCL due to repeated sampling**

Datasets	PCL	rPCL	ePCL	Speed up (rPCL/ePCL)
Iris	2.0	99.0	3.0	33.0
zoo	5.0	291.0	7.0	41.5
splice	2.5	129.0	4.0	32.2
hepatitis	0.5	38.0	3.0	12.6

Running time for 10 folds cross-validation (in seconds)

Conclusions

- **Good choice of K for PCL is important**
- **We introduce ePCL to choose optimal K**
  - ePCL uses pattern maintenance for efficiency
  - ePCL uses sub-sampling and CLT to choose K
- **Our technique improves PCL's accuracy and running time**
  - Especially in difficult cases !

Acknowledgements

Mengling Feng

Jinyan Li

Guimei Liu

Thanh-Son Ngo

- **… many slides in this presentation are contributed by Mengling and Son**

References

- M. Feng, G. Dong, J. Li, Y.-P. Tan, L. Wong. Evolution and maintenance of frequent pattern space when transactions are removed. *PAKDD 2007* : 489-497
- M. Feng, J. Li, Y.-P. Tan, L. Wong. Negative generator border for effective pattern maintenance. *ADMA 2008* : 217-228
- T.-S. Ngo, M. Feng, G. Liu, L. Wong. Efficiently finding the best parameter for the emerging pattern-based classifier PCL. *PAKDD 2010* : Part I, 121-133
- M. Feng, G. Dong, J. Li, Y.-P. Tan, L. Wong. Pattern space maintenance for data updates & interactive mining. *Computational Intelligence*, 26(3):282-317, Aug 2010

Thank You!

NUS

National University of Singapore