Rare Pattern Mining

Data Mining Algorithms 1 (DMA 1)
Winter 2024/25

Prof. Dr. Gabriel Marques Tavares
LMU Munich, Chair for Database Systems and Data Mining
Munich Center for Machine Learning (MCML.AI)

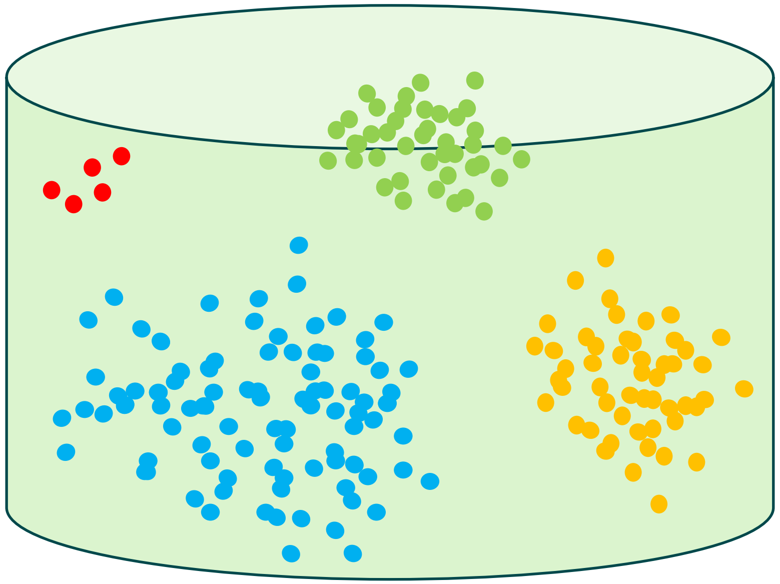
Significance of Low-support Patterns

- Contrast *infrequent patterns* with *frequent patterns*
  - Lot of existing work in frequent pattern mining

- Perspective of *relative support*
  - Infrequent patterns seem not important
  - For example, 0.1% (rare) vs. 10% (frequent)

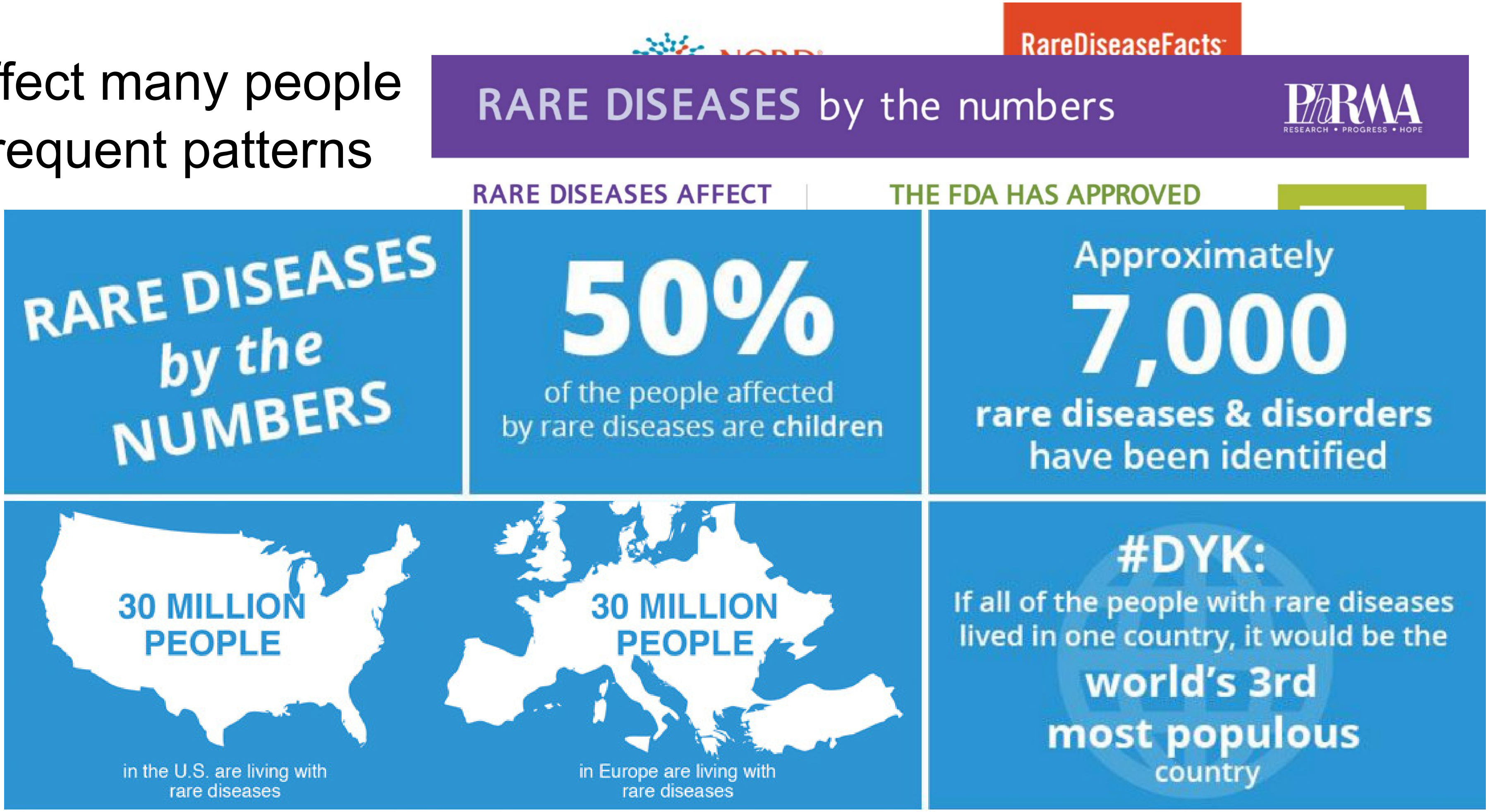
- Perspective of *absolute support*
  - Minor patterns maybe nevertheless **significant**
  - In 100.000 transactions: 100 occurrences vs. 10.000 occurrences

- Many applications with important rare patterns (more than just outliers)
  - Scientific and medical domains, vehicle accidents data, synthetic data generation



Low-support patterns in medical datasets

- Rare deseases affect many people
- Don’t appear as frequent patterns



Source: Global Genes. https://globalgenes.org/rare-diseases-facts-statistics/

https://rareaction.org/about/rare-diseases/

Low-support patterns in vehicle accidents datasets

- Dense representation
  - Categories for road conditions, speed, severity, etc.
- Example
  - Pattern {*bad-road-condition, high-speed, serious-accident*} may be rare.
  - Should be avoided, nevertheless.
- Actionable
  - Improve traffic conditions to prevent from high damages and from hurting people seriously.

Low-support patterns for synthetic dataset generation with real data

- Extend rare-pattern datasets for benchmarking purposes (*data augmentation*)

- Better generation quality
  - Overcome difficulty to cover all low support patterns by sampling

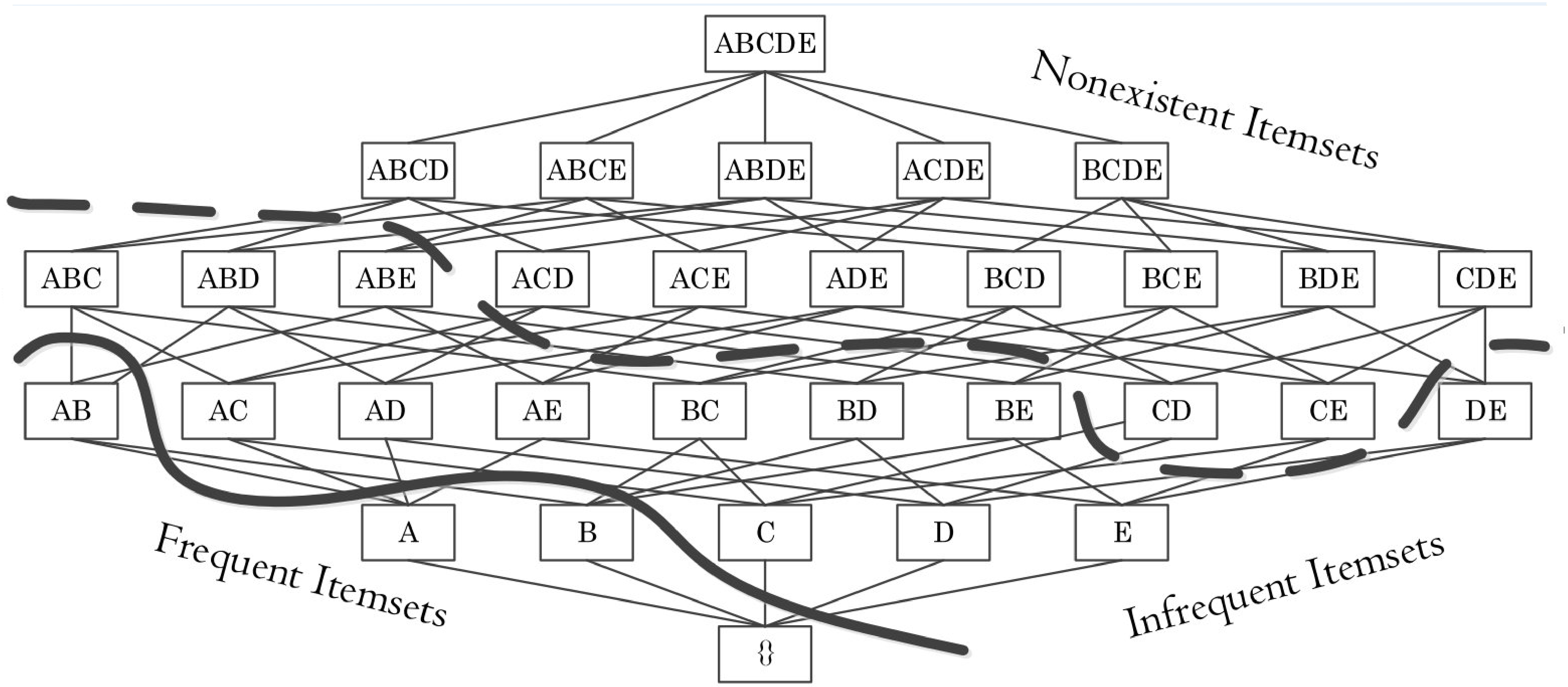
- More flexibility in controlling bias
  - Increase occurrence of low support patterns
  - Decrease occurrence of high support patterns

https://tdwi.org/articles/2017/04/12/dimensional-models-in-the-big-data-era.aspx

Existing approaches (selected)

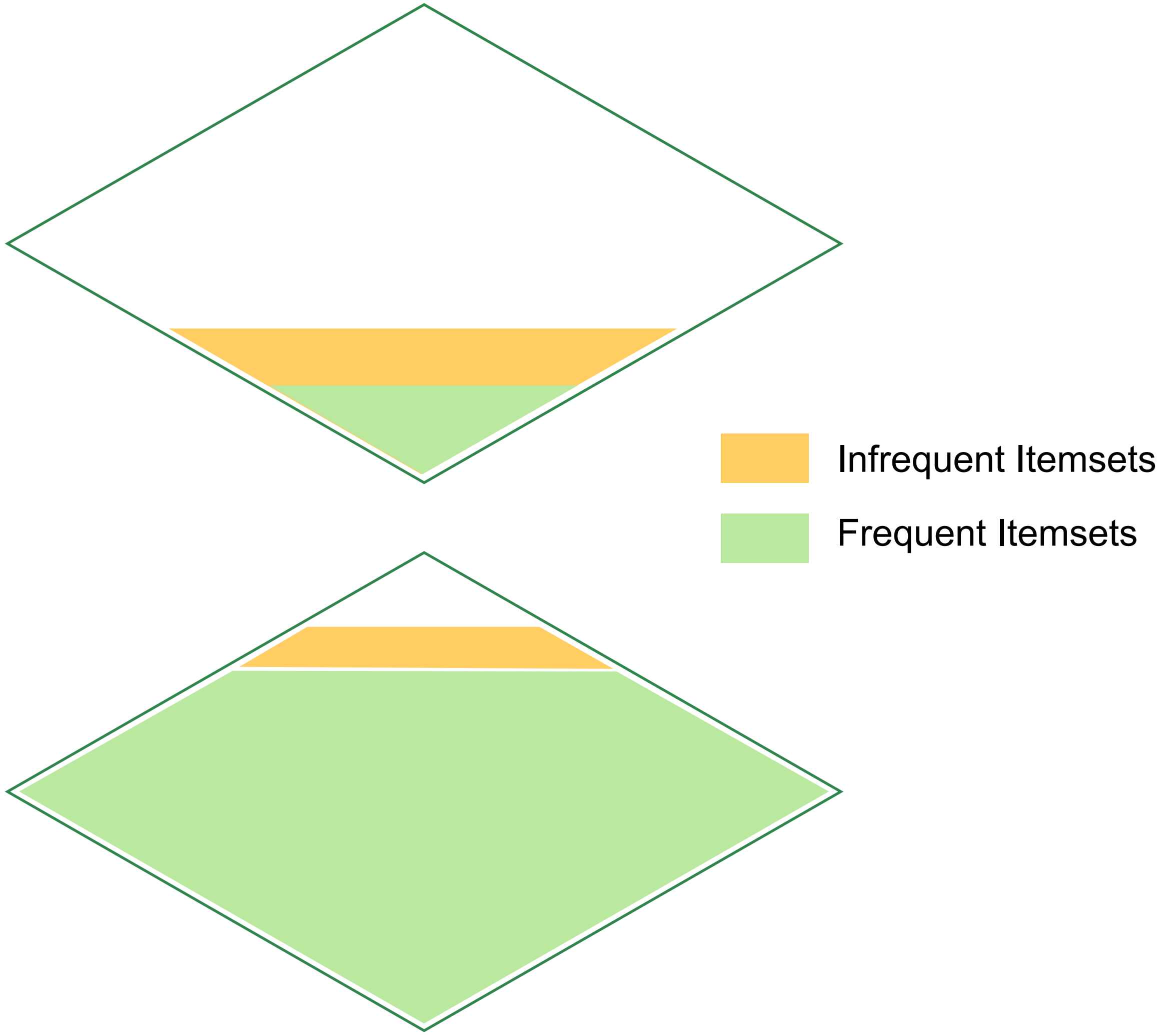
- T. Uno, M. Kiyomi, H. Arimura: *LCM ver. 2: Efficient mining algorithms for frequent / closed / maximal itemsets*. **FIMI** @ICDM 2004, CEUR 126.
- L. Troiano, G. Scibelli: *A time-efficient breadth-first level-wise lattice-traversal algorithm to discover rare itemsets*. **Data Mining and Knowledge Discovery** 28(3): 773–807, 2014.
- P. Fournier-Viger, J. C.-W. Lin, A. Gomariz, T. Gueniche, A. Soltani, Z. Deng, H. T. Lam: *The SPMF open-source data mining library version 2*. **ECML PKDD** 2016: 36–40.
- Y. Lu, T. Seidl: *Towards Efficient Closed Infrequent Itemset Mining Using Bi-Directional Traversing*. **DSAA** 2018: 140-149.
- Y. Lu, F. Richter, T. Seidl: *Efficient Infrequent Itemset Mining Using Depth-First and Top-Down Lattice Traversal*. **DASFAA** (1) 2018: 908-915.
- Y. Lu, F. Richter, T. Seidl: *Efficient Infrequent Pattern Mining Using Negative Itemset Tree*. **Complex Pattern Mining** 2020: 1-16.
- F. Richter, Y. Lu, L. Zellner, J. Sontheim, T. Seidl: *TOAD: Trace Ordering for Anomaly Detection*. **ICPM** 2020.
- L. Zellner, F. Richter, J. Sontheim, A. Maldonado, T. Seidl: *Concept Drift Detection on Streaming Data with Dynamic Outlier Aggregation*. **SA4PM** @ ICPM 2020

Power set lattice of itemsets (Hasse diagram)



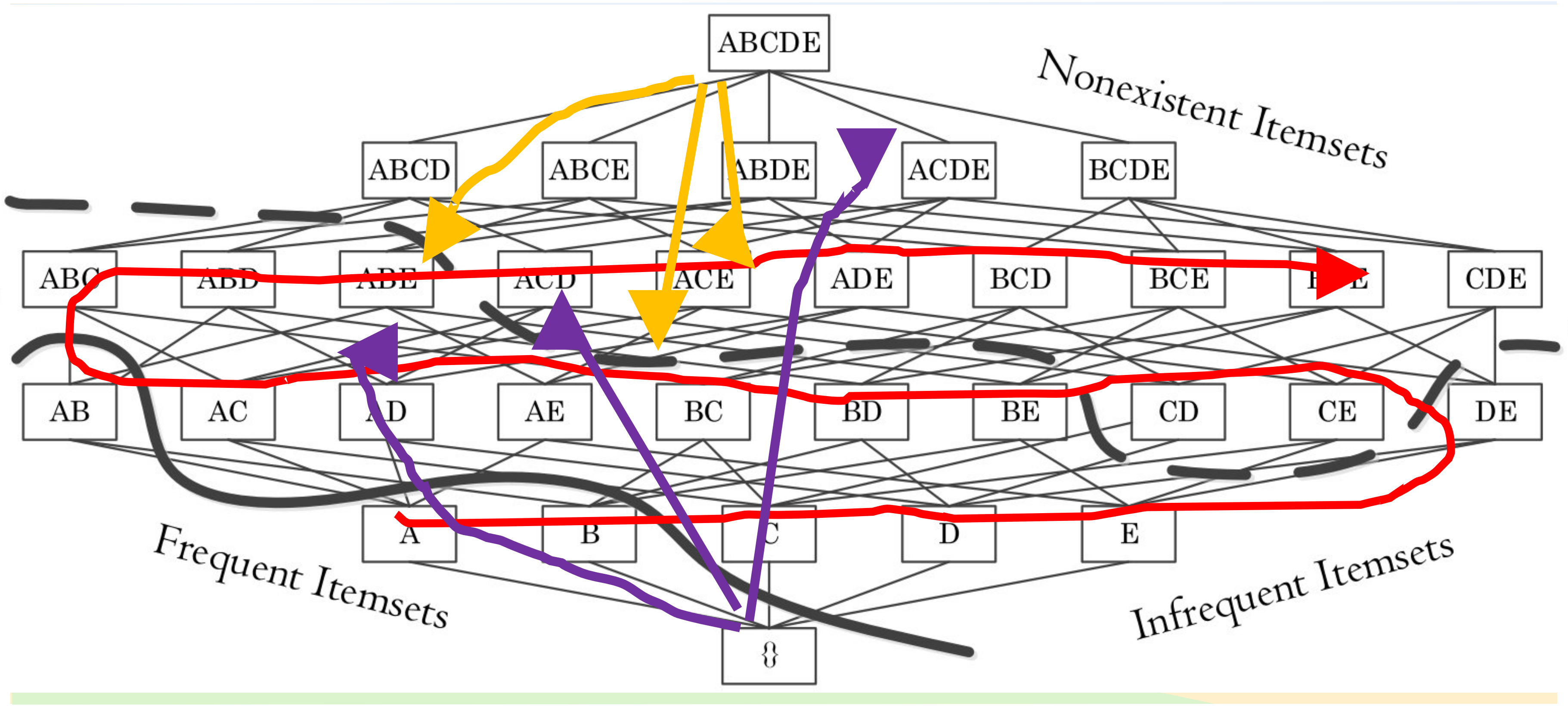
Sparse and dense datasets

- Sparse datasets
  - Large number of distinct items
  - Short transactions
  - Bottom-up traversal is efficient



- Dense datasets
  - (Relatively) small number of distinct items
  - Long transactions
  - Top-down may be preferred

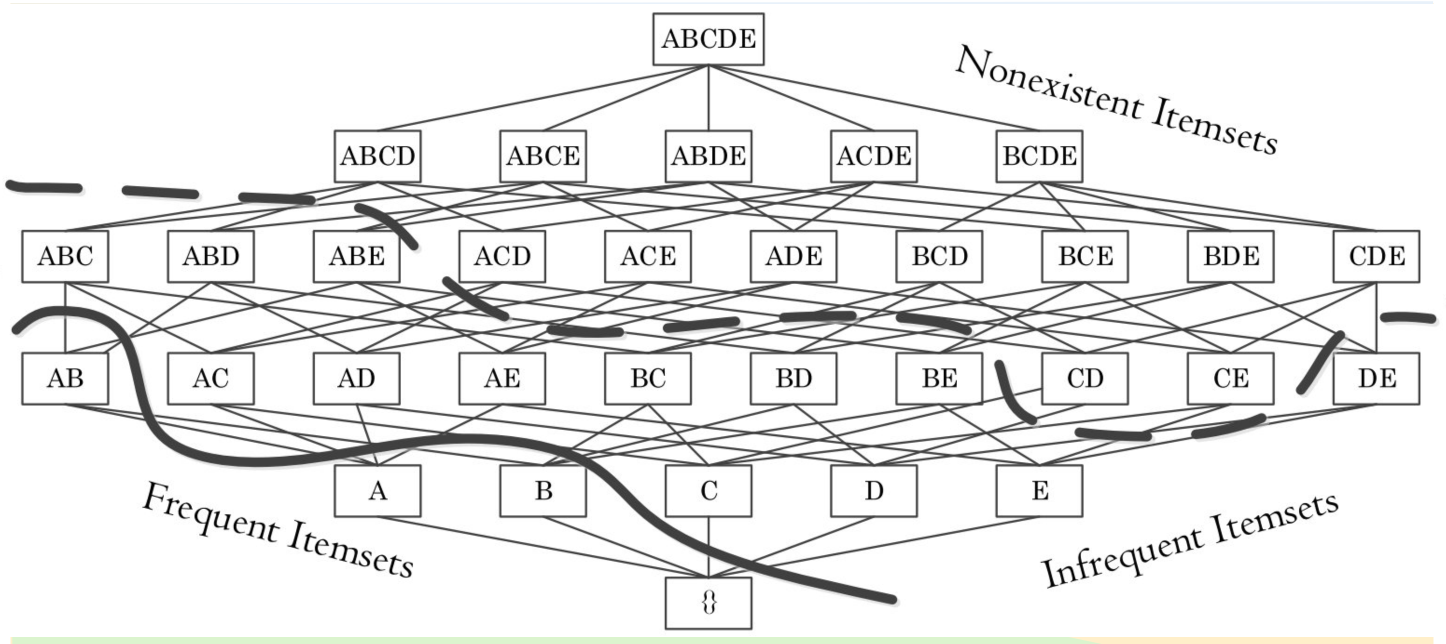
Traversal orders for the power set lattice



Breadth-first vs. Depth-first
(e.g., Apriori vs. FP-growth)

Bottom-up vs. Top-down
(Apriori, FP-growth vs. Rarity)

Traversal orders: Some observations



- Well-known results:
  - Breadth-first traversal is slower than depth-first traversal
  - Bottom-up traversal (almost all existing algorithms) is extremely slow when $minSup$ is small

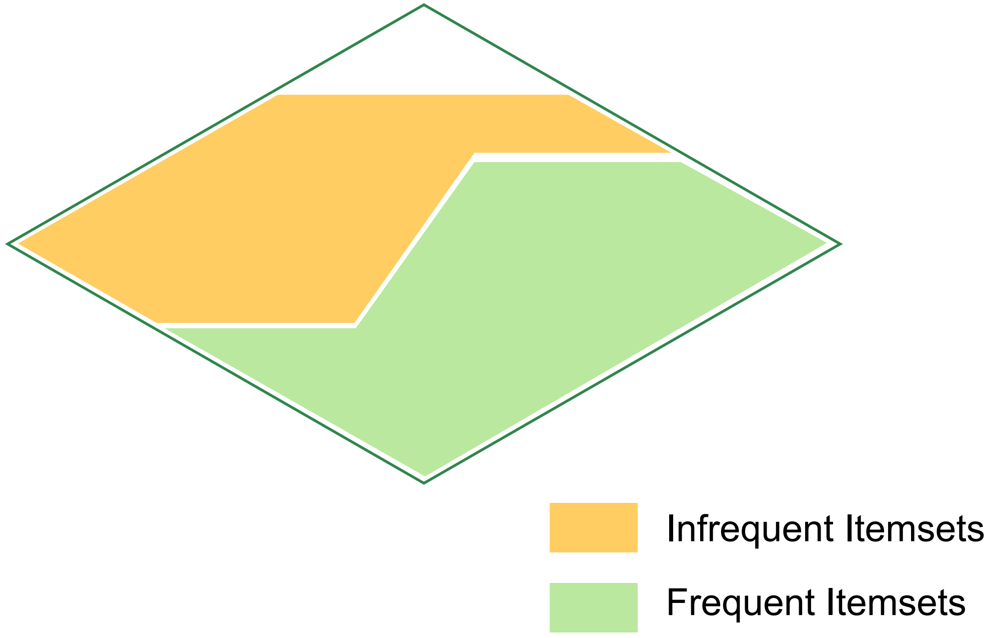
- Top-down traversal is better if the **number of frequent patterns** is **huge**

Traversal orders: What is the best for real datasets?

- In reality, dataset is both sparse and dense
  - a:4, b:5, c:2, d:2, e:2
  - Less frequent items: c, d, e
  - More frequent items: a, b

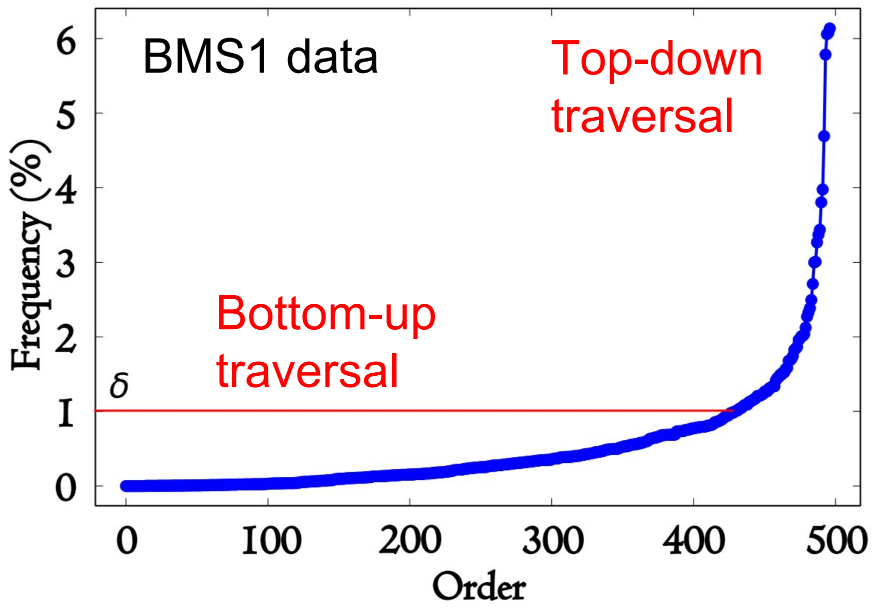
Tid	Itemset
1	{a,b,c}
2	{a,b,d}
3	{b,c}
4	{a,b}
5	{a,b,e}
6	{d,e}

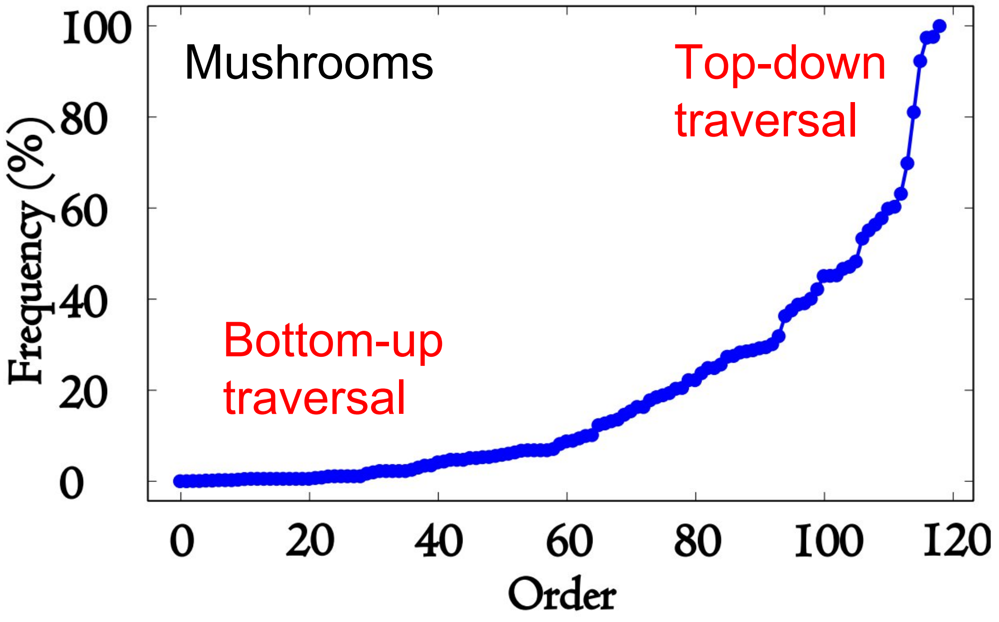
Tid	Itemset
1	{a,b}
2	{a,b}
3	{b}
4	{a,b}
5	{a,b}



RaCloMiner – Bi-directional traversal

- Basic idea: Combine both, bottom-up and top-down traversal
- Our proposal: **Ra**re **Clo**sed Itemset **Miner** [Lu, Seidl, DSAA 2018]
- As top-down traversal is not efficient, select split point carefully.
- Preliminary heuristics: choose value around inflection point.



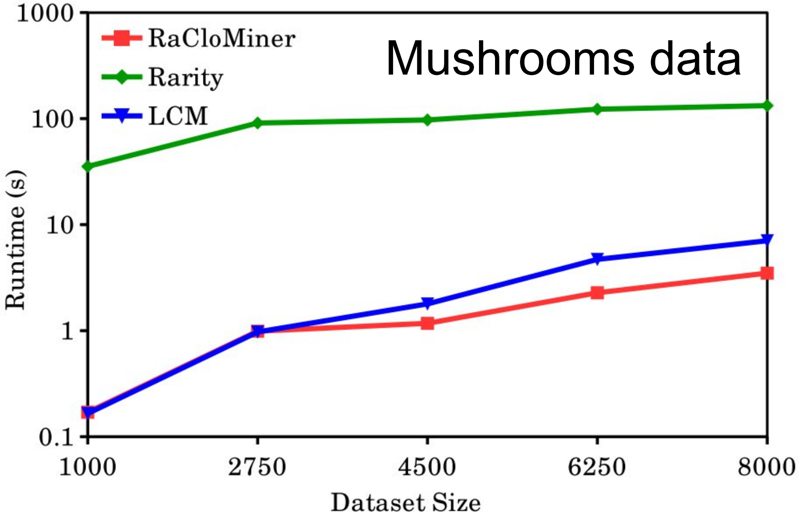


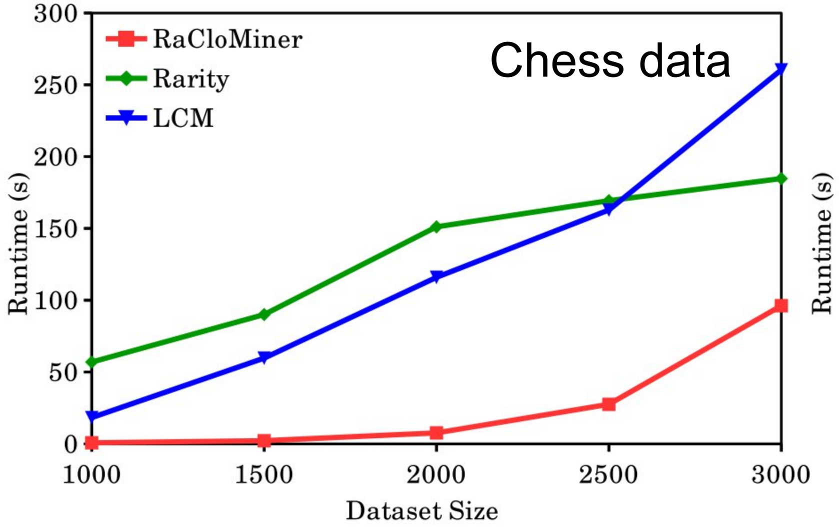
Experimental setting

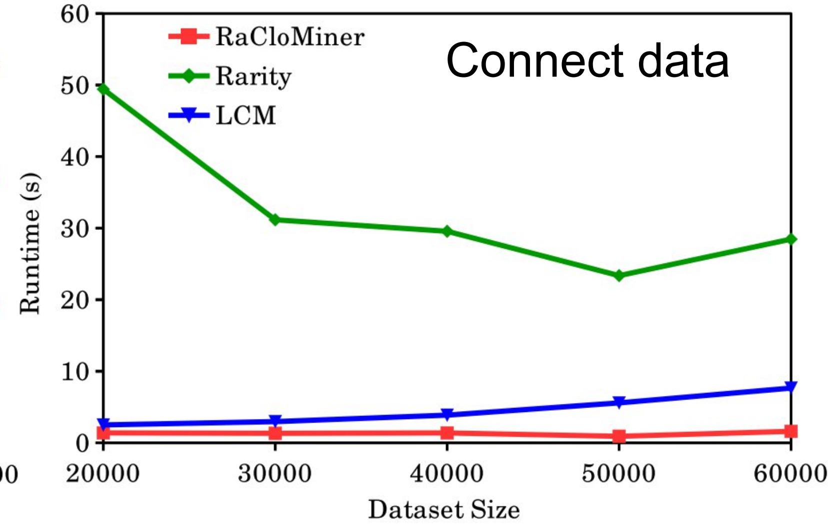
- Datasets from FIMI repository (U Antwerp) – 3 sparse datasets, 3 dense datasets
- **RaCloMiner**, implemented in Java – [Lu, Seidl, DSAA 2018]
- **LCM** implemented by using SPMF[2] library – [Uno et al., FIMI 2004] [Fournier-Viger et al., ECML PKDD 2016]
- **Rarity**: a breadth-first top-down approach for comparison – [Troiano, Scibelli, DMKD 2014]

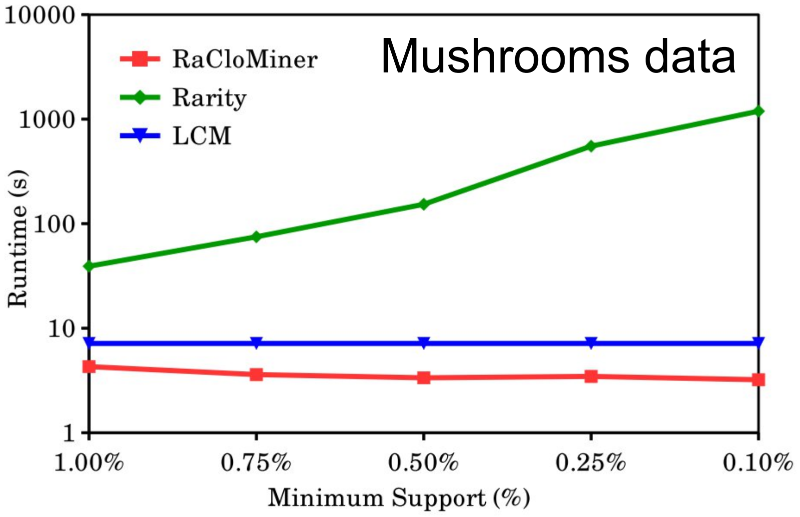
Database	Size (N)	Items ($\|I\|$)	Average length (L)
retail	88162	16470	10.3
BMS1	59602	497	2.5
BMS2	77512	3340	4.6
mushrooms	8415	119	23
chess	3196	75	37
connect	67556	129	43

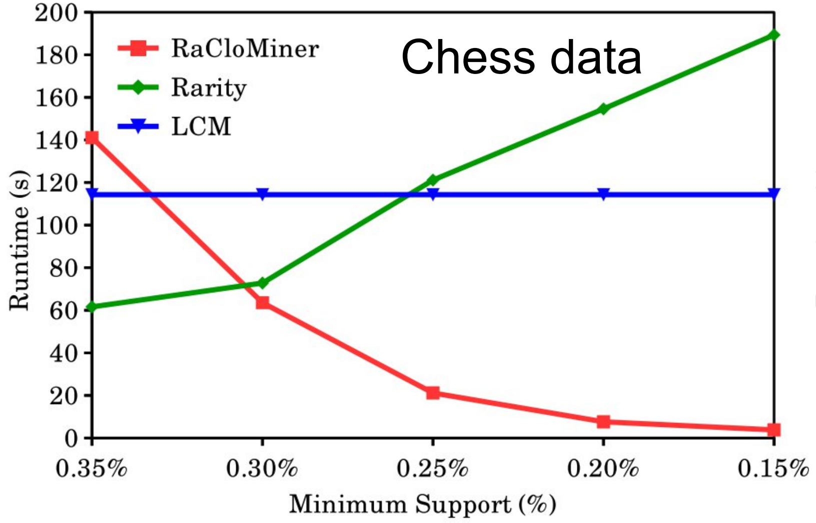
Experimental results for *dense* datasets

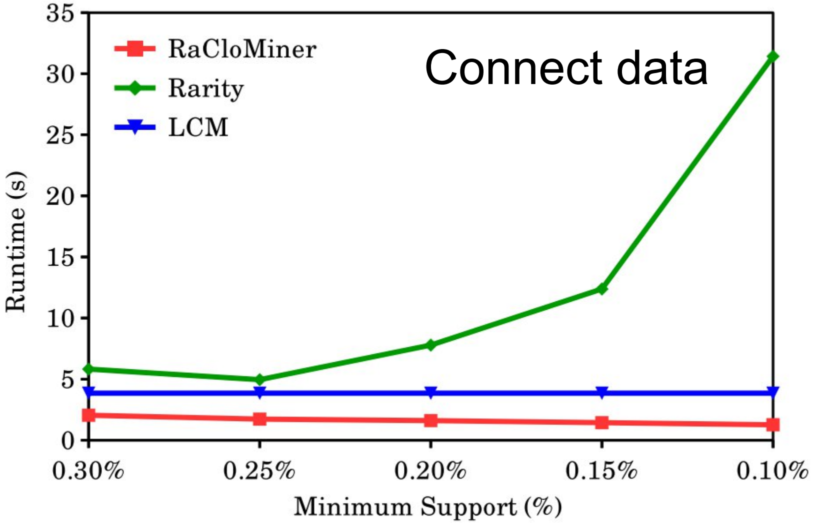












Experimental results for *sparse* datasets