Research Paper Engineering

Comparitive Study of Apriori & FP Growth Algorithms

* Pratiksha Shendge ** Tina Gupta

* Asst. Prof Prestige Institute of Engineering & Science

** Asst. Prof Prestige Institute of Engineering & Science

ABSTRACT

This paper presents a comparison between classical frequent pattern mining algorithms that use candidate set generation and test and the algorithms without candidate set generation. In order to have some experimental data to sustain this comparison a representative algorithm from both categories mentioned above was chosen (the Apriori,FP-growth).

Keywords: Data Mining, frequent pattern, Apriori, FP-growth.

Introduction

In data mining, association rule learning is a popular and well researched method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using different measures of interestingness.Based on the concept of strong rules, introduced association rules for discovering regularities between products in large-scale transaction data recorded by point-of-sale (POS) systems in supermarkets. For example, the rule {onions,potatoes}=>{burger} found in the sales data of a supermarket would indicate that if a customer buys onions and potatoes together, he or she is likely to also buy

hamburger meat. Such information can be used as the basis for decisions about marketing activities such as, e.g., promotional pricing or product placements. In addition to the above example from market basket analysis association rules are employed today in many application areas including Web usage mining, intrusion detection, Continuous production and bioinformatics. As opposed to sequence mining, association rule learning typically does not consider the order of items either within a transaction or across transactions.

The Apriori Algorithm in a Nutshell

- Find the frequent itemsets: the sets of items that have minimum support
- A subset of a frequent itemset must also be a frequent itemset
- i.e., if {AB} is a frequent itemset, both {A} and {B} should be a frequent itemset – Iteratively find frequent itemsets with cardinality from 1 to k (k-itemset)
- Use the frequent itemsets to generate association rules.

FP-Growth in a Nutshell

It allows frequent itemset discovery without candidate itemset generation. Two step approach:

– Step 1: Build a compact data structure called the FP-tree
- Built using 2 passes over the data-set.

– Step 2: Extracts frequent itemsets directly from the FP-tree

Step 1: FP-Tree Construction

· FP-Tree is constructed using 2 passes over the data-set:
Pass 1:
– Scan data and find support for each item.
– Discard infrequent items.
– Sort frequent items in decreasing order based on their support.

Use this order when building the FP-Tree, so common prefixes can be shared.

Pass 2:

Nodes correspond to items and have a counter

1. FP-Growth reads 1 transaction at a time and maps it to a path
2. Fixed order is used, so paths can overlap when transactions share items (when they have the same prfix).

– In this case, counters are incremented

3. Pointers are maintained between nodes containing the same item, creating singly linked lists (dotted lines)

– The more paths that overlap, the higher the compression. FP-tree may fit in memory.

4. Frequent itemsets extracted from the FP-Tree.

Step 2: Frequent Itemset Generation

· FP-Growth extracts frequent itemsets from the FP-tree.
· Bottom-up algorithm - from the leaves towards the root
· Divide and conquer: first look for frequent itemsets ending in e, then de, etc. . . then d, then cd, etc. . .
· First, extract prefix path sub-trees ending in an item(set). (hint: use the linked lists)

FP-Growth vs. Apriori

· Apriori visits each transaction when generating a new candidate sets; FP-Growth does not o `Can use data structures to reduce transaction list
· FP-Growth traces the set of concurrent items; Apriori generates candidate sets
· FP-Growth uses more complicated data structures & mining techniques

Algorithm Analysis Results

· FP-Growth IS NOT inherently faster than Apriori
· Intuitively, it appears to condense data
· Mining scheme requires some new work to replace candidate set generation
· Recursion obscures the additional effort
· FP-Growth may run faster than Apriori in circumstances
· No guarantee through complexity which algorithm to use for efficiency Improvements to FP-Growth
· None currently reported
· MLFPT

- o Multiple Local Frequent Pattern Tree
- o New algorithm that is based on FP-Growth.
- o Distributes FP-Trees among processors
- · No reports of complexity analysis or accuracy of FP-Growth

Real World Applications

- · Zheng, Kohavi, Mason – "Real World Performance of Association Rule Algorithms"
- o Collected implementations of Apriori, FP-Growth, CLOSET, CHARM, MagnumOpus
- o Tested implementations against 1 artificial and 3 real data sets
- o Time-based comparisons generated

Apriori & FP-Growth

Apriori

- o Implementation from creator Christian Borgelt (GNU Public License)
- o C implementation
- o Entire dataset loaded into memory

FP-Growth

- o Implementation from creators Han & Pei
- o Version – February 5, 2001

Datasets

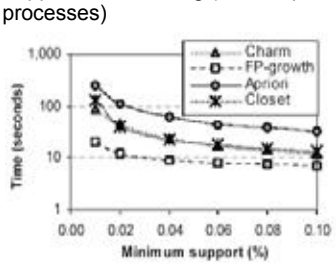
- · IBM-Artificial
- o Generated at IBM Almaden (T10I4D100K)
  Often used in association rule mining studies
- · BMS-POS
- o Years of point-of-sale data from retailer
- · BMS-WebView-1 & BMS-WebView-2
- o Months of clickstream traffic from e-commerce web sites

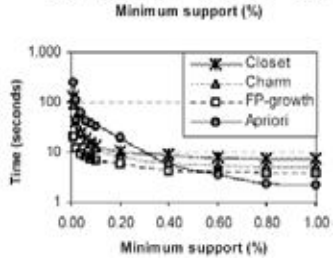
Dataset Characteristics

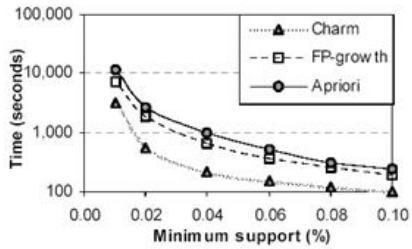
	Transactions	Distinct Items	Maximum Trans. Size	Average Trans. Size
IBM-Artificial	100,000	870	29	10.1
BMS-POS	515,597	1,657	164	6.5
BMS-WebView-1	59,602	497	267	2.5
BMS-WebView-2	77,512	3,340	161	5.0

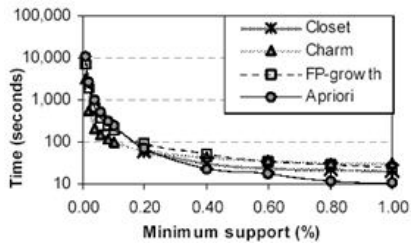
Experimental Considerations

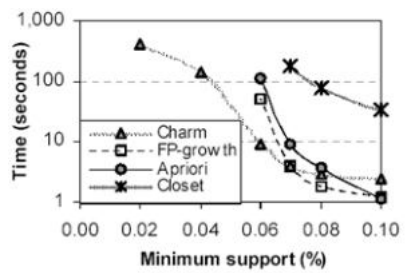
- · **Hardware Specifications**
- o Dual 550MHz Pentium III Xeon processors
- o 1GB Memory
- o Support { 1.00%, 0.80%, 0.60%, 0.40%, 0.20%, 0.10%, 0.08%, 0.06%, 0.04%, 0.02%, 0.01% }
- o Confidence = 0%
- o No other applications running (second processor handles system processes)

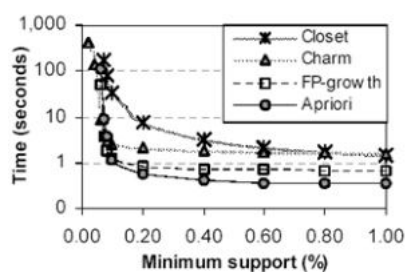


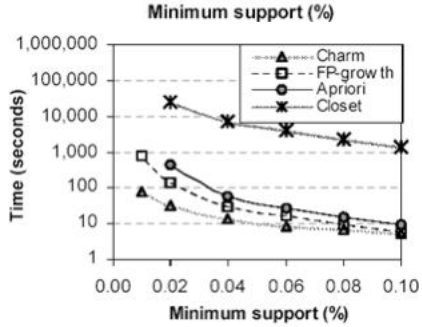


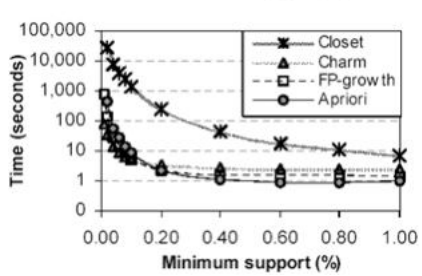












Study Results – Real Data

- · At support m 0.20%, Apriori performs as fast as or better than FP-Growth
- · At support < 0.20%, Apriori completes whenever FP-Growth completes

- o Exception – BMS-WebView-2 @ 0.01%
- · When 2 million rules are generated, Apriori finishes in 10 minutes or less
- o Proposed – Bottleneck is NOT the rule algorithm, but rule analysis

Real Data Results

		BMS-POS		BMS-WebView-1		BMS-WebView-2	
Algorithm	**Support**	**Time**	**Rules**	**Time**	**Rules**	**Time**	**Rules**
Apriori	0.01	186m	214,300,568	Failed	Falied	Failed	Failed
FP-Growth		120m		Failed		13m 12s	
Apriori	0.04	16m 9 s	5,061,105	Failed	Failed	58s	1,096,720
FP-Growth		10m 41s		Failed		29s	
Apriori	0.06	8m 35s	1,837,824	1m 50s	3,011,836	28s	510,233
FP-Growth		6m 7s		52s		16s	
Apriori	0.10	3m 58s	530,353	1.2s	10,360	9.1s	119,335
FP-Growth		3m 12s		1.2s		5.9s	
Apriori	0.20	1m 14s	103,449	0.4s	1,516	2.4s	12,665
FP-Growth		1m 35s		0.7s		2.3s	

Real-World Study Conclusions

- · FP-Growth (and other non-Apriori) perform better on artificial data.
- · On all data sets, Apriori performs sufficiently well in reasonable time periods for reasonable result sets
- · FP-Growth may be suitable when low support, large result count, fast generation are needed
- · Future research may best be directed toward analyzing association rules

REFERENCES

Han, Jiawei, Pei, Jian, and Yin, Yiwen, “Mining Frequent Patterns without Candidate Generation”. In Proc. of the ACM SIGMOD Int. Conf. on Management of Data, pages 1-12, Dallas, Texas, USA, 2000. | Zaiane, Osmar, El-Hajj, Mohammad, and Lu, Paul, “Fast Parallel Association Rule Mining Without Candidacy Generation”. In Proc. of the IEEE 2001 International Conference on Data Mining (ICDM'2001), San Jose, CA, USA, November 29-December 2, 2001