2A1-IOS-3b-3

A Decremental Utility Mining Algorithm Based on the Pre-large Concepts

Chun-Wei Lin[1] Tzung-Pei Hong[*2,*3] Guo-Cheng Lan[4] Jia-Wei Wong[3] Wen-Yang Lin[2]

[1] IIIRC, Harbin Institute of Technology Shenzhen Graduate School, P. R. China

[*2] Department of CSIE, National University of Kaohsiung, Taiwan, R.O.C.

[*3] Department of CSE, National Sun Yat-sen University, Taiwan, R.O.C.

[4] Department of CSIE, National Cheng Kung University, Taiwan, R.O.C.

In the past, utility mining was proposed to measure the utility values of purchased items for revealing high utility itemsets from a quantitative database. In dynamic data mining, transactions may be inserted, or deleted in the database. A batch mining procedure must rescan the whole updated database to maintain the up-to-date information. In this paper, a decremental mining algorithm is thus proposed for efficiently maintaining the discovered high utility itemsets due to transaction deletion based on the pre-large concept. Experimental results show that the proposed decremental high utility mining algorithm outperforms existing batch algorithms.

1. Introduction

In conventional data mining techniques, mining association rules [Agrawal 1993, Agrawal 1994, Park 1995] is the most popular approach. In association rule mining, each item is treated as a binary variable for discovering interesting relationships between itemsets. The frequency of an itemset, however, is insufficient for identifying highly profitable itemsets.

Utility mining [Chan 2003, Yao 2006, Liu 2005a, Lan 2011] was thus proposed to improve the limitations of frequent itemsets. Liu et al. designed a two-phase algorithm for efficiently extracting high utility itemsets based on the downward closure property [Liu 2005b]. Transaction-weighted utilization is used as an effective upper bound of each candidate itemset in the transaction to reduce the number of candidate itemsets for later processing. An additional database scan is performed to determine the real utility values of the remaining candidates to identify high utility itemsets. In the above approaches, the database is assumed to be static and the mining process is performed in a batch mode.

In real-world applications, transactions may be inserted [Cheung 1996] or deleted [Cheung 1997] from the database. The discovered frequent itemsets may become invalid, or some new information may emerge in the updated database. Hong et al. proposed pre-large itemsets to solve these problems [Hong 2001, Hong 2007]. In this paper, a decremental mining algorithm based on pre-large concept for transaction deletion to efficiently update the discovered high utility itemsets is proposed. A two-phase approach [Liu 2005b] is used to reduce the number of candidate itemsets in utility mining, thus reducing the processing time for mining high utility itemsets. Based on the proposed algorithm, it is seldom to rescan the original database for maintaining and updating the high utility itemsets.

Contact: Tzung-Pei Hong, National University of Kaohsiung, 700, Kaohsiung University Rd., Nanzih District, 811. Kaohsiung, Taiwan, R.O.C., +88675919191, tphong@nuk.edu.tw

2. Review of Related Work

In this section, mining association rules and high utility itemsets are briefly reviewed.

2.1 Mining association rules

Data mining can be divided into many specific areas due to its applications [Lin 2013a, Lin 2013b], but the most common approach is to extract patterns or rules from data sets in a particular representation Traditional data mining is used to extract useful itemsets or rules from binary database. In 1993, Agrawal and Srikant first proposed the Apriori algorithm [Agrawal 1994] to discover association rules from a set of transactions in a level-wise way. It applied the downward closure property to prune unpromising candidate itemsets, thus improving the efficiency for discovering the frequent itemsets. Several algorithms have also been proposed to efficiently discover the desired association rules [Park 1995, Hong 2008, Lin 2009].

In real-world applications, transaction database usually grow over time and time and the procedure for mining association rules is performed in a batch way. Some new association rules may be generated and some old ones may become invalid. Cheung et al. thus respectively proposed the FUP (Fast-Updated) [Cheung 1996] and FUP2 [Cheung 1997] algorithms to effectively handle transaction insertion and transaction deletion for maintaining the frequent itemsets. Hong et al. proposed the pre-large itemsets for efficiently maintaining the discovered information in dynamic databases [Hong 2001, Hong 2007]. The algorithms are needless to rescan the original database until a number of transactions have been inserted or deleted. The nine cases for transaction deletion based on the pre-large concept are shown in Figure 1.

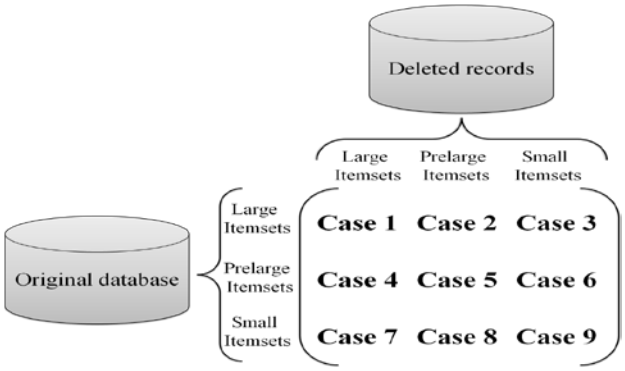

Figure 1: Nine cases arisen due to the transaction deletion.

2.2 High Utility Itemsets

Utility mining [Liu 2005a, Yao 2006, Lan 2011, Lin 2012], an extension of frequent itemset mining, is based on the measurement of local transaction utility and external utility. In the past, many algorithms were proposed for mining high utility itemsets. Yao et al. proposed an algorithm for efficiently mining high utility itemsets based on mathematical property of utility constraints [Yao 2006]. Two pruning strategies were used to reduce the search space based on the utility upper bounds and expected utility upper bounds, respectively. Liu et al. proposed a two-phase algorithm for efficiently discovering high utility itemsets [Liu 2005b] based on downward closure property. The proposed algorithm consists of two phases to level-wisely generate-and-test high utility itemsets. In the first phase, the transaction-weighted utility is used as an effective upper bound of each candidate itemset in the transactions according to the downward closure property of transaction-weighted utilization. This method could be adopted in the search space to reduce the number of candidate itemsets. In the second phase, an additional database scan is then performed to find the real utility values of the remaining candidate itemsets for discovering high utility itemsets.

3. Proposed Decremental Utility Mining Algorithm for Transaction Deletion

When transactions are deleted from the original database, the proposed decremental algorithm is executed to maintain the discovered high utility itemsets. The algorithm is described below.

***The designed algorithm*:**

INPUT: A profit table P of items, an original database D, an upper utility threshold S_u (= the minimum high utility threshold), a lower utility threshold S_l, the total utility TU^D of D, the large (high) transaction-weighted utilization itemsets $HTWU^D$ and the pre-large transaction-weighted utilization itemsets $PTWU^D$ with their transaction-weighted utilization values and actual utility values discovered from D, the safety transaction utility buffer *buf* for preserving the total utility value of the last processed transactions, and a set of deleted transactions d extracted from the original database D.

OUTPUT: The high utility itemsets (HU^U) in the updated database $U = (D - d)$.

STEP 1: If *buf* is not equal to 0, calculate the safety transaction utility bound f as:

$$f = \frac{S_u - S_l}{S_u} \times TU^D.$$

STEP 2: Calculate the item utility value u_{kj} of each item i_j in each deleted transaction t_k as:

$$u_{kj} = q_{kj} \times p_j,$$

where q_{kj} is *the* quantity of i_j in t_k and p_j is the profit of i_j in the profit table P; sum up the utility values of all the items in each transaction t_k as the transaction utility tu_k by:

$$tu_k = \sum_{j=1}^{m} u_{kj};$$

add the transaction utilities for all the deleted transactions in d as the total utility TU^d of d by:

$$TU^d = \sum_{k=1}^{n} tu_k.$$

STEP 3: Find the total utility TU^U for the updated database as:

$$TU^U = TU^D - TU^d,$$

where TU^D is the total utility in the original database D, and TU^d is the total utility in the deleted transactions d.

STEP 4: If ($buf + TU^d$) $\le f$, set $buf = buf + TU^d$ and do the STEP 5;
Otherwise, do the following substeps:
Substep 4-1: Rescan the updated database U to discover the large (high) transaction-weighted utilization itemsets $HTWU^U$ and pre-large transaction-weighted utilization itemsets $PTWU^U$.
Substep 4-2: Set $HTWU^D = HTWU^U$ as the set of the original large (high) transaction-weighted utilization itemsets and $PTWU^D = PTWU^U$ as the set of the original pre-large transaction-weighted utilization itemsets for the next transaction deletion in decremental mining.
Substep 4-3: Set $TU^D = TU^U$ as the output for high utility itemsets.
Substep 4-4: Set the $buf = 0$.

STEP 5: Find the items appearing in the deleted transactions d as the candidate 1-itemsets C_1.

STEP 6: Set $r = 1$, where r is the size of the itemsets currently being processed.

STEP 7: For each candidate r-itemset X in C_r, respectively find the transaction-weighted utilization $twu^d(X)$ and the actual utility $au^d(X)$ as:

$$twu^d(X) = \sum_{t_k \in d \,\&\, t_k \supseteq X} tu_k, \text{ and } (X) = \sum_{t_k \in d \,\&\, t_k \supseteq X \,\&\, i_j \in X} u_{kj},$$

where $twu^d(X)$ is the sum of the transaction utilities containing the itemset X in the deleted transactions d,

and $au^d(X)$ is the sum of the actual item utilities containing itemset X in the deleted transactions d.

STEP 8: For each large (high) transaction-weighted utilization itemset in $HTWU_r^D$ in the original database, do the following substeps (Cases 1, 2, and 3):

Substep 8-1: Set the updated transaction weighted-utilization $twu^U(X)$ of itemset X in the entire updated database as:

$$twu^U(X) = twu^D(X) - twu^d(X),$$

where $twu^D(X)$ is kept in the set of $HTWU^D$ in the original database, and $twu^d(X)$ is calculated in STEP 7 from the deleted transactions.

Substep 8-2: Set the updated actual utility $au^U(X)$ of itemset X in the entire updated database as:

$$au^U(X) = au^D(X) - au^d(X),$$

where $au^D(X)$ is kept in the set of $HTWU^D$ in the original database, and $au^d(X)$ is calculated in STEP 7 from the deleted transactions.

Substep 8-3: If $twu(X)/TU^U \geq S_u$, put itemset X in the set of $HTWU_r^U$, which is the large (high) transaction-weighted utilization r-itemset in the updated database;

Otherwise, if, $S_l \leq twu(X)/TU^U < S_u$, put itemset X in the set of $PTWU_r^U$, which is the pre-large transaction-weighted utilization r-itemset in the updated database;

Otherwise, discard itemset X since it becomes small after the database is updated.

STEP 9: For each pre-large transaction-weighted utilization itemset in $PTWU_r^D$ in the original database, do the same substeps in STEP 8 (Cases 4, 5, and 6).

STEP 10: Generate the candidate (r+1)-itemsets C_{r+1} from the large (high) transaction-weighted utilization r-itemsets $HTWU_r^U$ and the pre-large transaction-weighted utilization r-itemsets $PTWU_r^U$ ($HTWU_r^U \cup PTWU_r^U$).

STEP 11: Set $r = r + 1$.

STEP 12: Repeat STEPs 7 to 11 until no updated large (high) or pre-large transaction-weighted utilization itemsets are found.

STEP 13: Process each itemset X in the set of $HTWU^U$; if $au^U(X)/TU^U \geq S_u$, itemset X is a high utility itemset. Put itemset X into the set of HU^U.

STEP 14: Set $HTWU^D = HTWU^U$ as the set of the original large (high) transaction-weighted utilization itemsets and $PTWU^D = PTWU^U$ as the set of the original pre-large transaction-weighted utilization itemsets for the next transaction deletion in decremental mining.

After STEP 14, the algorithm stops and the set of HUU in STEP 13 included all high utility itemsets after the database is updated.

4. An Example

Assume the original database is shown in Table 1. It consists of 12 transactions with 5 items, denoted by *A* to *E*

Table 1: An original database in the example.

TID	*A*	*B*	*C*	*D*	*E*
1	6	0	0	0	0
2	0	1	0	5	0
3	0	6	0	0	0
4	0	0	5	0	0
5	0	0	0	0	8
6	2	0	2	0	3
7	0	1	0	4	0
8	0	4	0	0	0
9	0	2	3	7	0
10	0	0	0	0	1
11	0	5	2	5	0
12	0	3	0	3	0

Suppose the upper and lower utility threshold are respectively defined as 30%, 15%. The profit table for the items is shown in Table 2.

Table 2: The profit table

Item	Profit
A	6
B	2
C	15
D	7
E	10

Firstly, the two-phase-like algorithm is performed to find large (high) transaction-weighted utilization itemsets and pre-large transaction-weighted utilization itemsets with their actual utilities by the two utility thresholds. The results are respectively shown in Table 3 and Table 4.

Table 3: The large (high) transaction-weighted utilization itemsets and their actual utilities.

Itemset	Transaction-weighted utility	Actual utility
{*B*}	287	44
{*C*}	320	180
{*D*}	267	168
{*BC*}	173	89
{*BD*}	267	192
{*CD*}	173	159
{*BCD*}	173	173

Table 4: The pre-large transaction-weighted utilization itemsets and their actual utilities.

Itemset	Transaction-weighted utility	Actual utility
{*A*}	108	48
{*E*}	162	120

Suppose the last four transactions shown in Table 1 are chosen as the deleted transactions from the original database. The safety transaction utility bound is calculated to evaluate whether the original database is required to be rescanned or not, which is 280. The utility value of each item occurring in each deleted transaction (the last four transactions in Table 1) is first calculated. The results are shown in Table 5. The total utility of the four deleted transactions is calculated as 210, and the total utility of the updated database is calculated as 350.

Table 5: The transaction utilities for the last four deleted transactions.

TID	*A*	*B*	*C*	*D*	*E*	*tu*
9	0	2	3	7	0	98
10	0	0	0	0	1	10
11	0	5	2	5	0	75
12	0	3	0	3	0	27

In this example, since the (buf + TU^d) (= 0 + 210) is smaller than the safety transaction utility bound (f = 280). The original database is not required to be rescanned for updating. The appeared items in the deleted transactions are {*B*}, {*C*}, {*D*} and {*E*}. The transaction-weighted utilization and the actual utility of each candidate 1-itemset in the deleted transactions are thus calculated. The results are shown in Table 6.

Table 6: The transaction-weighted utilization and the actual utility of each candidate 1-itemset in the deleted transactions.

1-itemset	*twu*	*au*
{*B*}	200	20
{*C*}	173	75
{*D*}	200	105
{*E*}	10	10

For each 1-itemset whether in the set of large (high) transaction-weighted utilization 1-itemset $HTWU_1^D$ in the original database and in the set of pre-large transaction-weighted utilization 1-itemset $PTWU_1^D$ in the original database are respectively processed. The results are shown in Table 7.

Table 7: The large (high) transaction-weighted utilization 1-itemsets with their actual utilities

	Itemset	Transaction-weighted utilization	Actual utility
Large (high) transaction-weighted utilization	{*A*}	108	48
	{*C*}	147	105
	{*E*}	152	110
Pre-large transaction-weighted utilization	{*B*}	87	24
	{*D*}	67	63

After that, the candidate 2-itemsets are then formed by Apriori-like approach from Table 7. The generated results are {*AB*}, {*AC*}, {*AD*}, {*AE*}, {*BC*}, {BD}, {*BE*}, {*CD*}, {*CE*} and {*DE*}. The STEPs 7 to 11 are then repeated until no candidate itemsets are generated. The final results are then generated and shown in Table 8.

Table 8: The final results for large (high) transaction-weighted utilization itemsets with their actual utilities.

	Itemset	Transaction-weighted utilization	Actual utility
Large (high) transaction-weighted utilization	{*A*}	108	48
	{*C*}	147	105
	{*E*}	152	110
Pre-large transaction-weighted utilization	{*B*}	87	24
	{*D*}	67	63
	{*AC*}	72	42
	{*AE*}	72	42
	{*BD*}	67	67
	{*CE*}	72	60
	{*ACE*}	72	72

Finally, the large (high) transaction-weighted utilization itemsets in Table 8 are then determined to evaluate whether they are high utility itemsets or not in the updated database. After that, the results are then shown in Table 9.

Table 9: The final results.

	Itemset	Actual utility	Ratio (%)	HU
Large (high) transaction-weighted utilization	{*A*}	48	13.7%	-
	{*C*}	105	30.0%	HU
	{*E*}	110	31.4%	HU

In this example, the set of $HTWU^U$ = {*A*, *C*, *E*}, and the set of $PTWU^U$ = {*B*, *D*, *AB*, *AC*, *BD*, *CE*, *ACE*}. They are then considered as the set of large (high) transaction-weighted utilization $HTWU^D$ and the set of pre-large transaction-weighted utilization $PTWU^D$, respectively for the next transaction deletion in decremental mining. Thus, the final results for the high utility itemsets are {*C*, *E*}.

5. Experimental Results

In the experiments, the foodmart database [Microsoft] is used to evaluate the performance of the proposed algorithm. The foodmart dataset is collected from an anonymous chain store composed by a quantitative database about the products sold by the chain store. There are 21,556 transactions and 1,559 items in the dataset. Figure 2 shows the distribution of profit values in the utility table for the foodmart dataset.

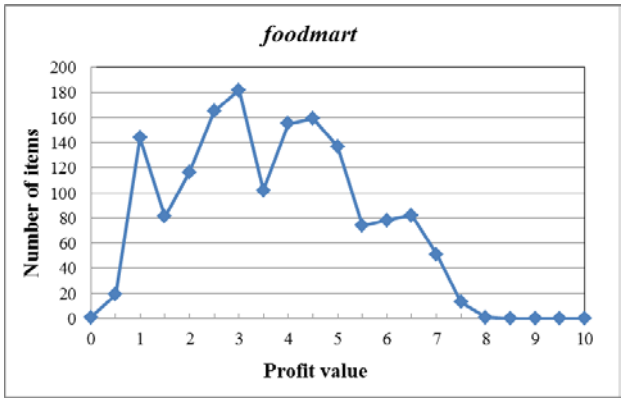

Figure 2: The profit value distribution in the foodmart dataset.

In the experimental evaluation, the two-phase high utility mining (TP-HUI) algorithm [Liu 2005b], the decremental high utility mining algorithm based on FUP concepts (FUP-HUI-DEL) algorithm and the proposed high utility mining algorithm based

on pre-large concepts (PRE-HUI-DEL) algorithm are then compared to respectively show the performance.

The first 21,556 transactions are initially used to mine the large (high) and pre-large transaction-weighted utilization itemsets with their actual utility values as well. The minimum high utility threshold (upper utility threshold) is set at 0.01% to evaluate the performance of three algorithms. The lower utility threshold is thus set at 0.0098%. Figure 3 showed the execution times of the three algorithms. The 100 transactions are then sequentially deleted from the original database.

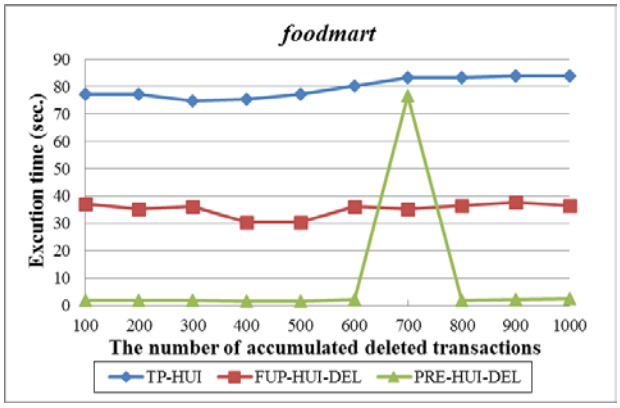

Figure 3: The comparisons of the execution times for transaction deletion.

It can be observed from Figure 3 that the proposed PRE-HUI-DEL algorithm also ran faster than the other two algorithms in transaction deletion. Experiments are also made to evaluate the efficiency of the proposed PRE-HUI-DEL algorithm in different minimum high utility threshold values shown in Figure 4. The minimum high utility threshold (upper utility threshold) is then set from 0.01% to 0.014%, increases 0.001% each time. The lower utility threshold for the proposed algorithm is thus set at 0.0098% to 0.0138%, decreases 0.0002% at each time of the upper utility threshold.

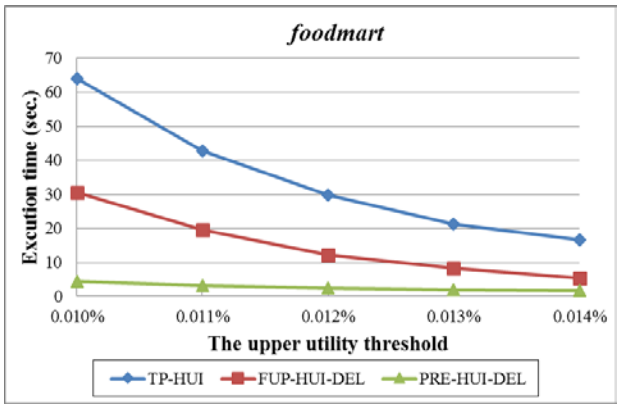

Figure 4: The comparisons of the execution time in different minimum utility thresholds.

It can easily be observed from Figure 4 that the execution time of the proposed PRE-HUI-DEL is much less than that by the TP-HUI and FUP-HUI-DEL algorithms for handling transaction deletion at different minimum high utility thresholds in the foodmart dataset.

6. Conclusions

In this paper, a decremental algorithm for efficiently maintaining high utility itemsets was proposed for transaction deletions based on the concept of pre-large itemsets. When transactions are removed from the original database, the proposed decremental algorithm partitions itemsets in the deleted transactions into three cases according to whether they have large (high), pre-large, or small transaction-weighted utilization in the original database. Each part is then processed individually to maintain the discovered high utility itemsets. Experimental results show that the proposed decremental high utility mining algorithm outperforms existing high utility mining algorithms.

References

[Agrawal 1993] Agrawal, R., Imieliński, T., and Swami, A.: Mining association rules between sets of items in large databases, *ACM SIGMOD International Conference on Management of Data*, pp. 207-216 (1993)

[Agrawal 1994] Agrawal, R. and Srikant, R.: Fast algorithms for mining association rules in large databases, *The International Conference on Very Large Data Bases*, pp. 487-499 (1994)

[Chan 2003] Chan, R., Yang, Q., and Shen, Y. D.: Mining high utility itemsets, *IEEE International Conference on Data Mining*, pp. 19-26 (2003)

[Cheung 1996] Cheung, D. W., Han , J., Ng, V., and Wong, C. Y.: Maintenance of discovered association rules in large databases: an incremental updating technique, *The International Conference on Data Engineering*, pp. 106-114 (1996)

[Cheung 1997] Cheung, D. W., Lee, S. D., and Kao, B.: A general incremental technique for maintaining discovered association rules, *The International Conference on Database Systems for Advanced Applications*, pp. 185-194 (1997)

[Hong 2001] Hong, T. P., Wang, C. Y., and Tao, Y. H.: A new incremental data mining algorithm using pre-large itemsets, *Intelligent Data Analysis*, Vol. 5, pp. 111-129 (2001)

[Hong 2007] Hong, T. P. and Wang, C. Y.: Maintenance of association rules using pre-large itemsets, *Intelligent Databases: Technologies and Applications*, pp. 44-60 (2007)

[Hong 2008] Hong, T. P., Lin, C. W., and Wu, Y. L.: Incrementally fast updated frequent pattern trees, *Expert Systems with Applications*, Vol. 34, Issue 4, pp. 2424-2435, (2008)

[Lan 2011] Lan, G. C., Hong, T. P., and Tseng, Vincent S.: Discovery of High Utility Itemsets from On-shelf Time Periods of Products, *Expert Systems with Application*, Vol. 38, Issue 5, pp. 5851-5857 (2011)

[Lin 2009] Lin, C. W., Hong, T. P., and Lu, W. H.: The Pre-FUFP algorithm for incremental mining, *Expert Systems with Applications*, Vol. 36, Issue 5, pp. 9498-9505 (2009)

[Lin 2012] Lin, C. W., Lan, G. C., and Hong, T. P.: An incremental mining algorithm for high utility itemsets, *Expert Systems with Applications*, Vol. 39, Issue 8, pp. 7173-7180 (2012)

[Lin 2013a] Lin, C. W., Hong, T. P., Chang, C. C., and Wang, S. L.: A Greedy-based Approach for Hiding Sensitive Itemsets by Transaction Insertion, *Journal of Information Hiding and*

Multimedia Signal Processing, Vol. 4, No. 4, pp. 201-227 (2013)

[Lin 2013b] Lin, C. W. and Hong, T. P.: A survey of fuzzy web mining, *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Vol. 3, pp. 190-199 (2013)

[Liu 2005a] Liu, Y., Liao, W. K., and Choudhary, A.: A fast high utility itemsets mining algorithm, *The International Workshop on Utility-based Data Mining*, pp. 90-99 (2005)

[Liu 2005b] Liu, Y., Liao, W. K., and Choudhary, A.: A two-phase algorithm for fast discovery of high utility itemsets, *Lecture Notes in Computer Science*, Vol. 3518, pp. 141-143 (2005)

[Microsoft] Microsoft, "Example database foodmart of Microsoft analysis services," Available: http://msdn.microsoft.com/en-us/library/aa217032(SQL.80).aspx.

[Park 1995] Park, J. S., Chen, M. S., and Yu, P. S.: An effective hash-based algorithm for mining association rules, *ACM SIGMOD Record*, Vol. 24, pp. 175-186 (1995)

[Yao 2006] Yao, H. and Hamilton, H. J.: Mining itemset utilities from transaction databases, *Data and Knowledge Engineering*, Vol. 59, pp. 603-626 (2006)