Adapting the Right Measures for Pattern Discovery: A Unified View

Junjie Wu, *Member, IEEE*, Shiwei Zhu, Hui Xiong, *Senior Member, IEEE*, Jian Chen, *Fellow, IEEE*, and Jianming Zhu

***Abstract*—This paper presents a unified view of interestingness measures for interesting pattern discovery. Specifically, we first provide three necessary conditions for interestingness measures being used for association pattern discovery. Then, we reveal one desirable property for interestingness measures: the support-ascending conditional antimonotone property (SA-CAMP). Along this line, we prove that the measures possessing SA-CAMP are suitable for pattern discovery if the itemset-traversal structure is defined by a support-ascending set enumeration tree. In addition, we provide a thorough study on the family of the generalized mean (GM) measure and show their appealing properties, which are exploited for developing the GMiner algorithm for finding interesting association patterns. Finally, experimental results show that GMiner can efficiently identify interesting patterns based on SA-CAMP of the GM measure, even at an extremely low level of support.**

***Index Terms*—Conditional antimonotone property (AMP), correlation computation, generalized mean (GM), interestingness measure, set enumeration tree (SET).**

I. Introduction

RECENT years have witnessed an increasing interest in finding interesting but rare patterns in large-scale high-dimensional data from different application domains, such as Web texts, market-basket transactions, gene expression data, and graphs [5], [6], [9]. However, many previous studies limited their scope to the *postevaluation* of interesting patterns [23], which can be computationally expensive and often misses some interesting but infrequent patterns. In addition, while some previous studies worked on the interestingness measures [23], [25], these studies did not provide a comprehensive understanding of interestingness measures. In particular, it is not clear how to adapt the right measures for efficiently finding patterns with a low level of support.

Indeed, as many new interestingness measures have been developed in the literature, there is a critical need to understand the key properties of these measures for finding rare but interesting association patterns. To this end, we provide an organized study of 35 interestingness measures for pattern discovery. We focus on answering the following two questions.

1) What are the most suitable interestingness measures that can be used for identifying interesting association patterns even at a low level of support?
2) What are the desirable computational properties that interestingness measures should possess for pattern discovery?

Along the line of answering these two questions, three key contributions are provided in this paper as follows. **First**, we identify three necessary conditions for an interestingness measure to be a suitable measure for association pattern discovery. In particular, we have examined 35 measures for their uniqueness, extensibility, and antimonotonicity. **Second**, we extend the antimonotone property (AMP) to the notion of support-ascending conditional AMP (SA-CAMP). We prove that, if the itemset-traversal structure is defined by a support-ascending set enumeration tree (SET) (SA-SET), then any suitable measure that possesses SA-CAMP can be used for the interesting pattern discovery. **Third**, we provide a thorough study on the appealing properties of the family of the generalized mean (GM) measures and develop an Apriori-like algorithm called GMiner to mine the interesting patterns from large-scale data.

Finally, experiments on various real-world data sets show that GMiner is very efficient for mining interesting patterns with the help of SA-CAMP and the upper-bound pruning. Also, GMiner shows its merit in finding non-cross-support and nontrivial interesting patterns even at very low support levels.

II. Related Work

In this section, we briefly introduce the interestingness measures for association analysis and highlight a few works on interesting pattern discovery using measures with the AMP.

Manuscript received July 6, 2011; revised October 26, 2011 and January 11, 2012; accepted January 26, 2012. Date of publication March 9, 2012; date of current version July 13, 2012. The work of J. Wu was supported in part by the National Natural Science Foundation of China (NSFC) under Grants 71171007, 70901002, 71031001, 70890080, and 90924020 and in part by the Doctoral Fund of Ministry of Education of China under Grant 20091102120014. The work of H. Xiong was supported in part by NSFC under Grant 71028002 and in part by the National Science Foundation under Grants CCF-1018151 and IIP-1069258. The work of J. Chen was supported in part by NSFC under Grant 70890082. The work of J. Zhu was supported in part by NSFC under Grant 60970143. This paper was recommended by Associate Editor M. S. Obaidat.

J. Wu is with the Information Systems Department, School of Economics and Management, Beihang University, Beijing 100191, China (e-mail: wujj@buaa.edu.cn).

S. Zhu (corresponding author) and J. Zhu are with the School of Information, Central University of Finance and Economics, Beijing 100081, China (e-mail: zhusw@cufe.edu.cn; zjm@cufe.edu.cn).

H. Xiong is with the Management Science and Information Systems Department, Rutgers, The State University of New Jersey, New Brunswick, NJ 08901-8554 USA (e-mail: hxiong@rutgers.edu).

J. Chen is with the Department of Management Science and Engineering, School of Economics and Management, Tsinghua University, Beijing 100084, China (e-mail: chenj@sem.tsinghua.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2012.2188283

In the literature, many interestingness measures have been proposed to mine the truly interesting patterns [14], [29]. Piatetsky-Shapiro proposed the statistical independence of rules as an interestingness measure [19]. Brin *et al.* [4] proposed *lift* and χ^2 as correlation measures and developed an efficient mining method. Omiecinski [18] and Lee *et al.* [15] found that *all-confidence, coherence*, and *cosine* [28] were null invariant [23] and, thus, were good measures for mining correlation rules in transaction databases. Blanchard *et al.* [2] designed a rule interestingness measure, *directed information ratio*, based on information theory. Hilderman and Hamilton [12] and Tan *et al.* [23] provided well-organized comparative studies for the interestingness measures, respectively. More survey papers on interestingness measures for association rules can be found in [16] and [17]. Recently, Wu *et al.* [25] have used the notion of the GM to generalize some interestingness measures. The GM measure studied in this paper just comes from this idea.

However, existing research studies mainly use the aforementioned measures for the *postevaluation* of interesting patterns. The widely cited work by Tan *et al.* [23] is a good example. The primary problem preventing the aforementioned measures from being used in the mining algorithms is that they often lack of the AMP [1]. Nevertheless, there are still some measures possessing the AMP that have been identified. Lee *et al.* [15] proposed the CoMine algorithm for the mining of *all-confidence* and *coherence* patterns. Xiong *et al.* (2006) proposed the hyperclique mining algorithm for the efficient discovery of hyperclique patterns. They also pointed out that the *h-confidence* measure was equivalent to the *all-confidence* measure. Different from these studies, this paper makes a comprehensive study on the 35 measures from an *in-evaluation* perspective and finds the excellent measures (the GM family) with appealing properties for the interesting pattern discovery.

III. Problem Definition

A. Math Notation

Assume that $\mathcal{D}$ is a market-basket database containing multiple items (commodities) such as milk, bread, beer, etc. Let $\mathcal{I} = \{i_1, i_2, \ldots, i_K\}$ be the set of all items in $\mathcal{D}$ and $\mathcal{T} = \{t_1, t_2, \ldots, t_N\}$ be the set of all transactions containing the items. We denote the power set of $\mathcal{I}$ by $\mathcal{P}(\mathcal{I})$. The *support* of an itemset X is the proportion of transactions that contain X, denoted as $supp(X)$ or $X.supp$. The number of items contained by X is denoted by $|X|$. $X_k = \{x_k\}, x_k \in X$, is a 1-item subset of X. If the items of X have a preimposed order, then we use $x_{[k]}$ to denote the kth item and $X_{[k]}$ to denote the kth 1-item subset of X.

An interestingness measure M is a mapping of itemsets in $\mathcal{P}(\mathcal{I})$ to real scores in $\mathbb{R}$, i.e., $M : D_o \mapsto R_a$, where $D_o \subseteq \mathcal{P}(\mathcal{I})$ and $R_a \subseteq \mathbb{R}$. We use $D_o(M)$ and $R_a(M)$ to denote the domain and range of M, respectively. A higher $M(X)$ value indicates the higher interestingness of X. Based on $M(X)$, we have the following definition.

Definition 1 (M-Interesting Pattern): An itemset X is an M-interesting pattern if the following conditions hold: 1) $M(X) \geq min\_int$ and 2) $supp(X) \geq min\_supp$, where min_int and min_supp are the thresholds of interestingness and support, respectively.

B. Motivations and Problem Formulation

To mine M-interesting patterns is considered important, since the classic support–confidence framework employed for mining frequent patterns has two obvious drawbacks: 1) The confidence as the measure of rule interestingness may not disclose truly interesting relationships [9], [10], and 2) the support-based pruning strategy is not effective for data sets with skewed support distributions [27]. The first defect is usually illustrated by the well-known "coffee–tea" example [24], and the second defect can be understood from the dilemma of setting the support threshold. That is, if the threshold is low, we may extract too many spurious patterns involving items with substantially different support levels [27], such as {earrings, milk} in which the support of earrings is expected to be much lower than that of milk. On the contrary, if the threshold is high, we may miss many interesting patterns occurring at low levels of support [11], such as {earrings, gold ring, bracelet} that contains rare but expensive items.

Some existing studies [18], [23] exploited a *postevaluation* scheme to mine M-interesting patterns. In other words, they first generate frequent patterns and then identify interesting patterns from frequent patterns by applying interestingness measures M with the threshold. While it is simple, the *postevaluation* scheme has the same problem as the traditional frequent pattern mining algorithm such as Apriori; that is, rare but interesting patterns will not be identified if a higher threshold is used, and too many patterns will be generated for a low threshold.

Therefore, we exploit the *in-evaluation* scheme. In other words, the interestingness measures are used to generate interesting patterns directly during the mining process. Compared to the *postevaluation* scheme, the *in-evaluation* scheme has two advantages. By using the interestingness threshold, it first helps prune frequent but uninteresting patterns in a much more efficient manner. Also, it enables to set a low support threshold for rare but interesting patterns, since the pruning based on the interesting measure will play a significant role in the mining process. Thus, we define our problem to be studied as follows.

Problem Definition: Given a market-basket database $\mathcal{D}$, *select* a right interestingness measure M, to *find* all the M-interesting patterns using an *in-evaluation* scheme.

Finally, it is noteworthy that any interestingness measure mentioned in this study only uses the support information of the itemsets. This distinguishes our study from the existing studies on the subjective measures [21] and the constraint-based pattern discovery [13], which often use the *external* information such as the prices of the items.

IV. Candidate Interestingness Measures

In this section, we aim to collect the well-known interestingness measures and find the candidate measures that can be used for interesting pattern discovery.

A. Necessary Conditions

We first specify the necessary conditions for a measure to be a candidate measure. Denote the set of candidate measures as $\mathcal{C}$. We have the following properties.

Property 1 (Uniqueness): A candidate measure can only assign one interestingness score to an itemset. In other words, $M \in \mathcal{C} \Longrightarrow \forall X \in \mathcal{P}(\mathcal{I})$ and $X \in D_o(M)$, $M(X)$ is unique.

Remark: This property excludes the measures, such as confidence, which are designed for evaluating association rules. Such measures cannot hold the uniqueness since an itemset can generate various rules.

Property 2 (Extensibility): A candidate measure must have the ability to evaluate the interestingness of multi-itemsets; that is, $M \in \mathcal{C} \Longrightarrow (1)\ \forall X \in \mathcal{P}(\mathcal{I})$ and $|X| > 2$, $X \in D_o(M)$ and $(2)\ \forall X, Y \in \mathcal{P}(\mathcal{I})$, $M(X) < M(Y) \Rightarrow X \prec Y$, where $X \prec Y$ means that Y is more interesting than X.

Remark: Some interestingness measures, such as the odds ratio, are used for the evaluation of item pairs or two itemsets. To extend them to the multi-itemset case is very difficult. Thus, we exclude them from this study. Also, the extensibility property requires that the measure scores are comparable across itemsets of different sizes. This requirement does not hold for some measures, such as the χ^2 [4], the distribution of which suffers from the degree-of-freedom problem [7].

Property 3 (Antimonotonicity): A candidate measure must have the AMP; that is, $M \in \mathcal{C} \Longrightarrow \forall X, Y \in \mathcal{P}(\mathcal{I})$, $X \subseteq Y \Rightarrow M(X) \geq M(Y)$.

Remark: As we know, a measure possessing the AMP can help prune infrequent candidate supersets greatly. Thus, we adopt it as the requirement for the interesting pattern discovery. This differentiates our study from the direct computing studies [8]. Unfortunately, however, most existing interestingness measures do not have the antimonotonicity and, therefore, may be subject to the computation inefficiency problem.

B. Pool of Candidate Measures

Table I shows a collection of the interestingness measures, where "Uni.," "Ext.," and "Anti." denote the uniqueness, extensibility, and antimonotonicity properties, respectively. We then use the three necessary properties to filter out the unqualified measures.

1) Using the Uniqueness Property: As can be seen in Table I, 17 out of 35 measures do not hold the uniqueness property. This is not surprising since most of them are used purposefully for the association rule evaluation: $A \Rightarrow B$. One may argue that A and B in Table I can be viewed as two 1-itemsets derived from the 2-itemset: $A \bigcup B$; thus, the 17 measures can be also used to evaluate 2-itemsets. However, these measures still suffer from the asymmetric characteristic, which again violates the uniqueness property.

2) Using the Extensibility Property: As can be seen in Table I, five measures, namely, MI, α, Q, Y, and χ^2, do not hold the extensibility property. In the table, if we view A and B as two 1-itemsets, these five measures can be used to evaluate 2-itemsets for their symmetry characteristic. However, when it proceeds to the multi-itemset case, to derive the formulas of these measures is very difficult or even meaningless. Moreover, to the best of our knowledge, there is no existing research along this line except for the χ^2 measure [4]. However, χ^2 yet has its own problem of having varied degrees of freedom for itemsets of different sizes [7], which may make the comparison of two scores meaningless.

3) Using the AMP: Now, there are still 13 measures left for the checking of the AMP. Unfortunately, only two measures, i.e., $AllConf$ and h, can pass this check and finally enter the pool of candidate measures. This is not a coincident. Indeed, in [15], [18], and [27], $AllConf$ and h have been well studied for the interesting pattern discovery.

To sum up, we have only $\mathcal{C} = \{AllConf, h\}$. It seems that there is no other option. However, let us reconsider one important question: *Is the AMP the only way to reduce the computational complexity of interesting pattern discovery?* The answer is **no**. In the next section, we will introduce one alternative that can help enrich the pool of candidate measures.

V. Beyond the AMP

In this section, we propose the SA-CAMP. We prove that the interestingness measures holding the uniqueness, extensibility, and SA-CAMP can be used for interesting pattern discovery.

A. Definition of SA-CAMP

Here, we define the SA-CAMP as follows.

Definition 2 (SA-CAMP): A measure M is said to have the SA-CAMP if $\forall X, Y \in \mathcal{P}(\mathcal{I})$ and $X \subseteq Y$, given that

$$\forall i \in X, i' \in Y \setminus X, supp(\{i\}) \leq supp(\{i'\}) \tag{1}$$

we have $M(X) \geq M(Y)$.

Remark: As shown in (1), SA-CAMP makes a special request on the items in the difference set of the superset and the subset. That is, the items in the difference set must have higher support values than the items in the subset. Now, the remaining question is as follows: What are the measures that hold SA-CAMP?

B. Measures Possessing SA-CAMP

In [25], Wu *et al.* pointed out that the existing interestingness measures $AllConf$, $Kulc$, cos, h', and $MaxConf$ could be generalized by the GM measure

$$\begin{aligned} GM(X,p) &= \left(\frac{1}{K}\sum_{k} P(X|X_k)^{p'}\right)^{1/p'} \\ &= \frac{supp(X)}{\left(\frac{1}{K}\sum_{k=1}^{K} supp(X_k)^p\right)^{\frac{1}{p}}} \end{aligned} \tag{2}$$

where $p \in (-\infty, +\infty)$ and $p' = -p$. Then, when $p \to -\infty$, $-1, 0, 1$ and $+\infty$, we can get the $MaxConf$, $Kulc$, cos, h', and $AllConf$ measures, respectively. Note that, hereinafter, we may alternate the use of GM, $GM(p)$, $GM(X)$, or $GM(X,p)$ to call the GM measure, if there is no confusion. We have the following theorem.

TABLE I
PROPERTIES OF THE INTERESTINGNESS MEASURES

No.	Measure	Formula	Uni.	Ext.	Anti.
1	Confidence (c)	$P(B\|A)$	No	-	-
2	Centred Confidence (c')	$P(B\|A)-P(B)$	No	-	-
3	Laplace (L)	$(NP(A,B)+1)/(NP(A)+2)$	No	-	-
4	Conviction (V)	$P(A)P(\overline{B})/P(A,\overline{B})$	No	-	-
5	Certainty (T)	$(P(B\|A)-P(B))/(1-P(B))$	No	-	-
6	Added Value (AV)	$P(B\|A)-P(B)$	No	-	-
7	Klosgen (K)	$\sqrt{P(A,\overline{B})}(P(B\|A)-P(B))$	No	-	-
8	Bayes factor (B)	$(P(B\|A)/P(\overline{B}\|A))/(P(B)/P(\overline{B}))$	No	-	-
9	Least Contradiction (LC)	$(P(A,B)-P(A,\overline{B}))/P(B)$	No	-	-
10	Loevinger (Loe)	$(P(A,B)-P(A)P(B))/(P(A)P(\overline{B}))$	No	-	-
11	Sebag & Schoenauer (Seb)	$P(A,B)/P(A,\overline{B})$	No	-	-
12	Ganascia (Gan)	$2P(B\|A)-1$	No	-	-
13	Examples and Counter-Examples Rate (ECR)	$1-P(A,\overline{B})/P(A,B)$	No	-	-
14	Zhang (Z)	$(P(A,B)-P(A)P(B))/\max\{P(A,B)P(\overline{B}),P(A,\overline{B})P(B)\}$	No	-	-
15	Goodman-Kruskal's (λ)	$\frac{\sum_i \max_j P(A_i,B_j)+\sum_j \max_i P(A_i,B_j)-\max_i P(A_i)-\max_j P(A_j)}{2-\max_i P(A_i)-\max_j P(A_j)}$	No	-	-
16	Gini index (G)	$P(A)[P(B\|A)^2+P(\overline{B}\|A)^2]+P(\overline{A})[P(B\|\overline{A})^2+P(\overline{B}\|\overline{A})^2]-P(B)^2-P(\overline{B})^2$	No	-	-
17	J-Measure (J)	$P(A,B)\log(P(B\|A)/P(B))+P(A,\overline{B})\log(P(\overline{B}\|A)/P(\overline{B}))$	No	-	-
18	Mutual Information (MI)	$\frac{\sum_i\sum_j P(A_i,B_j)\log(P(A_i,B_j)/P(A_i)P(B_j))}{\min\{-\sum_i P(A_i)\log(P(A_i)),-\sum_j P(B_j)\log(P(B_j))\}}$	Yes	No	-
19	Odds ratio (α)	$(P(A,B)P(\overline{A},\overline{B}))/(P(A,\overline{B})P(\overline{A},B))$	Yes	No	-
20	Yule's Q (Q)	$\frac{P(A,B)P(\overline{A},\overline{B})-P(A,\overline{B})P(\overline{A},B)}{P(A,B)P(\overline{A},\overline{B})+P(A,\overline{B})P(\overline{A},B)}$	Yes	No	-
21	Yule's Y (Y)	$\frac{\sqrt{P(A,B)P(\overline{A},\overline{B})}-\sqrt{P(A,\overline{B})P(\overline{A},B)}}{\sqrt{P(A,B)P(\overline{A},\overline{B})}+\sqrt{P(A,\overline{B})P(\overline{A},B)}}$	Yes	No	-
22	Chi-Square (χ^2)	$\frac{N(P(A,B)P(\overline{A},\overline{B})-P(A,\overline{B})P(\overline{A},B))^2}{P(A)P(\overline{A})P(B)P(\overline{B})}$	Yes	No	-
23	Interest (I)	$P(X)/\prod_k P(X_k)$	Yes	Yes	No
24	Piatetsky-Shapiro's (PS)	$P(X)-\prod_k P(X_k)$	Yes	Yes	No
25	Information Gain (IG)	$\log(P(X)/\prod_k P(X_k))$	Yes	Yes	No
26	Collective Strength (S)	$\frac{1/(\prod_k P(X_k)+\prod_k P(\overline{X_k}))-1}{1/(P(X)+P(\overline{X}))-1}$	Yes	Yes	No
27	Kappa (κ)	$\frac{P(X)+P(\overline{X})-\prod_k P(X_k)-\prod_k P(\overline{X_k})}{1-\prod_k P(X_k)-\prod_k P(\overline{X_k})}$	Yes	Yes	No
28	ϕ-coefficient (ϕ)	$(P(X)-\prod_k P(X_k))/\sqrt[K]{\prod_k P(X_k)(1-P(X_k))}$	Yes	Yes	No
29	Max-Confidence ($MaxConf$)	$P(X)/\min_k P(X_k)$	Yes	Yes	No
30	Kulczynski ($Kulc$)	$\frac{P(X)}{K}\sum_k \frac{1}{P(X_k)}$	Yes	Yes	No
31	Cosine (cos)	$P(X)/\sqrt[K]{\prod_k P(X_k)}$	Yes	Yes	No
32	Cohesion' (h')	$KP(X)/\sum_k P(X_k)$	Yes	Yes	No
33	Generalized Mean (GM)	$(\frac{1}{K}\sum_k P(X\|X_k)^p)^{1/p}$	Yes	Yes	No
34	All-Confidence ($AllConf$)	$P(X)/\max_k P(X_k)$	Yes	Yes	Yes
35	Cohesion (h)	$P(X)/univ(X)$	Yes	Yes	Yes

Notes: 1. For first 17 measures, the association rule is: $A \Rightarrow B$; 2. It is easy to note that $\alpha \Leftrightarrow Q \Leftrightarrow Y$.
3. For the S and κ measures, we let $P(\overline{X}) \equiv P(\overline{X_1},\overline{X_2},\cdots,\overline{X_K})$ for simplicity; 4. h-confidence in [27] is equivalent to $AllConf$.
5. GM is the generalized measure of $AllConf$, $Kulc$, cos, h' and $MaxConf$ [25]; 6. $univ(X)=|\{t|\exists i\in t,\ i\in X\}|$.

Theorem 1: The GM measure holds SA-CAMP.

Proof: Without loss of generality, assume that $X=\{i_1,\ldots,i_K\}$ is a K-itemset ($K\geq 1$) and the superset $Y=X\bigcup\{i_{K+1},\ldots,i_{K+L}\}$ is a $(K+L)$-itemset ($L\geq 1$). $\forall k\leq K$ and $l\leq L$, $supp(Y_{K+l})\geq supp(X_k)$. Let $F(X,p)=((1/K)\sum_{k=1}^{K} supp(X_k)^p)^{1/p}$.

When $p\in(-\infty,0)\bigcup(0,+\infty)$, we have

$$F(Y,p)^p-F(X,p)^p = \frac{L}{K+L}\left(\frac{\sum_{l=1}^{L} supp(Y_{K+l})^p}{L}-\frac{\sum_{k=1}^{K} supp(X_k)^p}{K}\right). \quad (3)$$

Case 1) $p>0$. By (3), $F(Y,p)^p-F(X,p)^p\geq 0\Rightarrow F(Y,p)\geq F(X,p)$.

Case 2) $p<0$. By (3), $F(Y,p)^p-F(X,p)^p\leq 0\Rightarrow F(Y,p)\geq F(X,p)$.

Case 3) $p\to 0$. $F(X,0)=\sqrt[K]{\prod_{k=1}^{K} supp(X_k)}$, and $(F(Y,0)/F(X,0))^{(K+L/L)}=(\sqrt[L]{\prod_{l=1}^{L} supp(Y_{K+l})}/\sqrt[K]{\prod_{k=1}^{K} supp(X_k)})\geq 1$. Accordingly, $F(Y,0)\geq F(X,0)$.

Case 4) $p\to-\infty$. $F(X,-\infty)=\min_k supp(X_k)$. Accordingly, $F(Y,-\infty)=F(X,-\infty)$.

Case 5) $p\to+\infty$. $F(X,+\infty)=\max_k supp(X_k)$. Accordingly, $F(Y,+\infty)\geq F(X,+\infty)$.

In summary, regardless of the p value, $F(Y,p)\geq F(X,p)$. Since $supp(Y)\leq supp(X)$, we have

$$GM(Y,p)=\frac{supp(Y)}{F(Y,p)}\leq\frac{supp(X)}{F(X,p)}=GM(X,p).$$

Thus, the proof is completed. ■

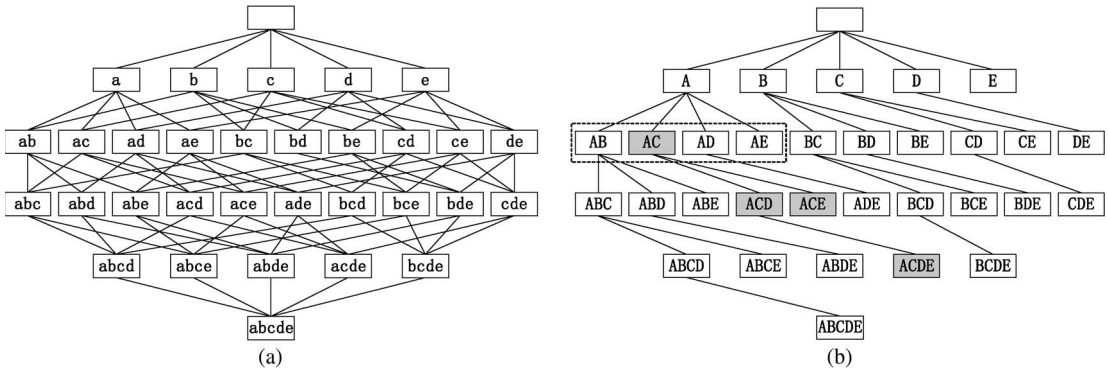

Fig. 1. Subset lattice and the SA-SET. (Note that (a) the shaded itemsets are for the illustration of the pruning effect of SA-CAMP in Example and (b) the itemsets in the dashed frame are for the illustration of the pruning effect of the GM upper bound in Lemma 1.) (a) Subset lattice of five items. (b) SA-SET of five items.

Remark: If we go through the measures from No. 23 to No. 33 in Table I, we can find that only the family of the GM measure holds SA-CAMP. This implies that the GM family can be added back to the pool of candidate measures for interesting pattern discovery, given that the SA-CAMP of GM can help reduce the computational complexity. Details are shown hereinafter.

C. GM for Interesting Pattern Discovery

To use GM for interesting pattern discovery, the key is to set an appropriate itemset-traversal sequence that can guarantee Formula 1 of SA-CAMP. Let us begin with a lattice view of itemsets as follows.

As we know, given the universal itemset $\mathcal{I}$ of a market-basket database, the search space consists of $2^{|\mathcal{I}|}$ different subsets. Fig. 1(a) shows the search space of five items in a Hasse lattice [22]. Altogether, 32 subsets must be traversed.

Here, we propose the SA-SET, a special SET [20], to simplify the lattice yet keep the completeness of the search. The SA-SET systematically enumerates all the itemsets using the *support-ascending order* on the underlying set of items; that is: 1) The root node of the SA-SET represents the empty set, and 2) the children of a node N enumerate those itemsets that can be formed by appending a single item of $\mathcal{I}$ to N, with the constraint that this item must be listed in the last and has a higher support than all the items in N.

Fig. 1(b) shows a sample SA-SET of five items from A to E, where $supp(\{A\}) \leq supp(\{B\}) \leq supp(\{C\}) \leq supp(\{D\}) \leq supp(\{E\})$. It is easy to note that the SA-SET is complete and correct—we can reach *every* itemset in Fig. 1(a) by traversing the SA-SET from top to bottom in Fig. 1(b), nothing more and nothing less. Given the notion of SA-SET, we have the following theorem.

Theorem 2: A measure M that holds uniqueness and extensibility can be used for the interesting pattern discovery if the following conditions hold: 1) M possesses SA-CAMP, and 2) the itemset-traversal structure is defined by SA-SET.

Proof: Given an uninteresting itemset X, let $Descend(X)$ denote the set of all the descendant nodes of X in the SA-SET. Then, $\forall Y \in Descend(X)$, $X \subseteq Y$, and $\forall i \in X$, $i' \in Y \setminus X$, $supp(\{i\}) \leq supp(\{i'\})$. Since M holds SA-CAMP, we have $M(Y) \leq M(X) < min\_int$. This means that, if X is uninteresting, then all its descendants in the SA-SET will be uninteresting and can be pruned safely. We complete the proof. ■

Corollary 1: GM can be used for interesting pattern discovery if the itemset-traversal structure is defined by the SA-SET.

Example: Let us look at the SA-SET in Fig. 1(b). Assume that the itemset $\{AC\}$ is uninteresting. Then, all its descendant supersets $\{ACD\}$, $\{ACE\}$, and $\{ACDE\}$ [the shaded nodes in Fig. 1(b)] are guaranteed to be uninteresting and can be pruned safely.

D. GM Upper Bound and Its AMP

The upper-bound pruning strategy was first introduced in the Two-step All-strong-Pairs corrElation queRy (TAPER) algorithm to efficiently find the strongly correlated item pairs [26]. Here, we derive an upper bound of GM as follows.

Definition 3 (The GM Upper Bound): For itemset $X = \{i_1, i_2, \ldots, i_K\}$ with items in a support-ascending order, $K \geq 2$, the GM upper bound is defined as

$$upper(GM(X,p)) = \frac{supp(X \setminus X_{[K]})}{\left(\frac{1}{K}\sum_{k=1}^{K} supp(X_k)^p\right)^{\frac{1}{p}}}. \quad (4)$$

Then, we have the following lemma.

Lemma 1: For two K-itemsets X and Y with items in a support-ascending order, if $X \setminus X_{[K]} = Y \setminus Y_{[K]}$, $supp(X_{[K]}) \leq supp(Y_{[K]})$, then $upper(GM(X,p)) \geq upper(GM(Y,p))$.

Lemma 1 indicates the simple AMP of the GM upper bound, and Fig. 1(b) shows its pruning effect. Assume that we have four candidate 2-itemsets $\{AB\}$, $\{AC\}$, $\{AD\}$, and $\{AE\}$, as highlighted by the dashed square. Since $supp(\{A\}) \leq supp(\{B\}) \leq supp(\{C\}) \leq supp(\{D\}) \leq supp(\{E\})$, by Lemma 1, we have $upper(GM(AB)) \geq upper(GM(AC)) \geq upper(GM(AD)) \geq upper(GM(AE))$. Now, if we find that $upper(GM(AB)) < min\_int$, then all these four itemsets must have GM values lower than the threshold and can be pruned safely without computing their exact GM values. If the number of the items is large, the computational savings can be considerable.

In addition to SA-CAMP and AMP of the upper bound, the GM measure also has some appealing properties such as the

The GMiner Algorithm

Input:

$\mathcal{D}$: the market-basket database.
min_supp: the minimum support threshold.
min_int: the minimum interestingness threshold.
p: the exponent parameter of GM.

Output:

$\mathcal{F} = \{X | X.supp \geq min\_supp, GM(X,p) \geq min\_int\}$.

1. Scan $\mathcal{D}$ once and get the set of frequent items $\mathcal{I}$;
2. Recode and sort the items in each trans. in a support ascending order; // $i \prec i' \Leftrightarrow \{i\}.supp < \{i'\}.supp$.
3. $k = 1$;
4. $\mathcal{F}_k = \{\{i\} | i \in \mathcal{I}\}$; $\mathcal{T}_k = \varnothing$;
5. **while** ($\mathcal{F}_k \neq \varnothing$) **do**
6. $\quad \mathcal{T}_{k+1} = \mathcal{C}_{k+1} = \varnothing$;
7. $\quad$ **foreach** $X \in \mathcal{F}_k, Y \in \mathcal{F}_k \bigcup \mathcal{T}_k$, $X \setminus \{x_{[k]}\} = Y \setminus \{y_{[k]}\}$, $x_{[k]} \prec y_{[k]}$ **do**
8. $\quad\quad L = X \bigcup \{y_{[k]}\}$ with $l_{[k+1]} = y_{[k]}$;
9. $\quad\quad$ **if** $\exists J \subset L, |J| = k, J.supp < min\_supp$ **then**
10. $\quad\quad\quad$ **continue;**
11. $\quad\quad$ **end if**
12. $\quad\quad$ **if** $upper(GM(L,p)) \geq min\_int$ **then**
13. $\quad\quad\quad \mathcal{C}_{k+1} = \mathcal{C}_{k+1} \bigcup L$;
14. $\quad\quad$ **else**
15. $\quad\quad\quad \mathcal{T}_{k+1} = \mathcal{T}_{k+1} \bigcup L$;
16. $\quad\quad$ **end if**
17. $\quad$ **end for**
18. $\quad$ **if** $\mathcal{C}_{k+1} \neq \varnothing$ **then**
19. $\quad\quad$ Count the support for each X in $\mathcal{C}_{k+1}$;
20. $\quad\quad \mathcal{F}_{k+1} = \{X | X \in \mathcal{C}_{k+1}, X.supp \geq min\_supp, GM(X,p) \geq min\_int\}$;
21. $\quad\quad \mathcal{T}_{k+1} = \{X | X \in \mathcal{C}_{k+1}, X.supp \geq min\_supp, GM(X,p) < min\_int\} \bigcup \mathcal{T}_{k+1}$;
22. $\quad\quad k = k + 1$;
23. $\quad$ **else**
24. $\quad\quad$ **break**;
25. $\quad$ **end if**
26. **end while**
27. **return** $\mathcal{F} = \bigcup_k \mathcal{F}_k$;

Fig. 2. GMiner algorithm.

cross-support property, which will be introduced in detail in the Appendix. Therefore, in the rest of this paper, we focus on the GM measure and propose algorithms for interesting pattern discovery using GM.

VI. GMiner Algorithm

In this section, we propose the GMiner algorithm for interesting pattern discovery using the GM measure and its SA-CAMP.

A. Overview of GMiner

Fig. 2 shows the pseudocodes of GMiner. Lines 1 and 2 are the preparations for the itemset traversal in SA-SET. The key point is to sort the items in a support-ascending order in line 2 ("$\prec$" indicates the order). The set of the frequent 1-itemsets $\mathcal{F}_1$ is then formed in lines 3 and 4. Note that we use the set $\mathcal{T}_k$ in line 4 for the purpose of generating candidate $(k+1)$-itemsets in the next iteration (lines 7 and 8). Detailed discussions about $\mathcal{T}_k$ will be given hereinafter.

Lines 5–26 describe the iterative process for the mining of all the remaining interesting itemsets. Each iteration consists of two main phases. Lines 7–17 are for the first phase—the generation of candidate itemsets. The generating and pruning strategies are the two key points that we will cover in detail in the next section. Then, in lines 18–25, GMiner scans the entire database to verify whether the support and GM values of the candidates can exceed the thresholds. Finally, in line 27, the interesting itemsets of different sizes are merged and returned.

B. Generation of Candidate Supersets

As we know, the classic Apriori algorithm employs the $\mathcal{F}_k \times \mathcal{F}_k$ strategy to generate candidate frequent itemsets. However, this strategy is not suitable for GMiner. The reason is that GM only possesses SA-CAMP rather than AMP. For example, as indicated by the SA-SET in Fig. 1(b), the fact that $\{AC\}$ is not interesting does not necessarily mean that $\{ABC\}$ is not interesting, since $\{ABC\}$ is the immediate superset of $\{AB\}$ rather than $\{AC\}$ according to the SA-SET. Thus, we may miss the candidate $\{ABC\}$ due to the absence of $\{AC\}$ if $\mathcal{F}_k \times \mathcal{F}_k$ is used.

To address this problem, we propose a novel $\mathcal{F}_k \times (\mathcal{F}_k \bigcup \mathcal{T}_k)$ strategy. Here, $\mathcal{F}_k$ contains all interesting k-itemsets, and $\mathcal{T}_k$ is a set to store the frequent but not interesting itemsets. The pseudocodes in lines 7–17 in Fig. 2 characterize the candidate generation process. If two itemsets X (from $\mathcal{F}_k$) and Y (from $\mathcal{F}_k \bigcup \mathcal{T}_k$) have the same prefix $(k-1)$-itemset and $supp(\{x_{[k]}\}) \leq supp(\{y_{[k]}\})$, a potential candidate will be generated as $L = \{x_{[1]}, \ldots, x_{[k]}, y_{[k+1]}\}$, as indicated by lines 7 and 8. This candidate is then subject to the frequency checking using the AMP of support in lines 9–11; that is, if a candidate's k-item subset is infrequent, then this candidate must be infrequent and should be discarded. Lines 12–16 are for the interestingness upper-bound checking of the candidates. Unlike the frequency checking, the candidates that have interestingness upper bounds smaller than the threshold will not be discarded but will be included into $\mathcal{T}_{k+1}$, which will be used for generating candidate $(k+2)$-itemsets. The candidates passing this checking will be included into $\mathcal{C}_{k+1}$ and subject to the final check of support and interestingness, as indicated by lines 20 and 21.

Now, we remain to prove that the $\mathcal{F}_k \times (\mathcal{F}_k \bigcup \mathcal{T}_k)$ strategy is complete, i.e., no interesting pattern will be missed. To this end, let us call the frequent k-itemset with an interesting prefix $(k-1)$-itemset the "FIP k-itemset." That is, a k-itemset X is an FIP k-itemset if the following conditions hold: 1) $supp(X) \geq min\_supp$, and 2) $X \setminus \{x_{[k]}\} \in \mathcal{F}_{k-1}$, where $x_{[k]}$ is the kth item in X. Apparently, an interesting k-itemset must be an FIP k-itemset, but the reverse is not true. Hereinafter, assume that all the items in an itemset have been ordered in a support-ascending order. We then have the following lemma.

Lemma 2: $\forall k \in \mathbb{Z}^+$, $k \geq 2$, if $\mathcal{F}_k \bigcup \mathcal{T}_k$ contains all FIP k-itemsets, then $\mathcal{F}_k \times (\mathcal{F}_k \bigcup \mathcal{T}_k)$ is complete, i.e., $\mathcal{F}_{k+1} \subseteq \mathcal{F}_k \times (\mathcal{F}_k \bigcup \mathcal{T}_k)$.

Proof: Assume that there is an interesting $(k+1)$-itemset $X = \{i_1, \ldots, i_{k+1}\} \notin \mathcal{F}_k \times (\mathcal{F}_k \bigcup \mathcal{T}_k)$. Let $I = \{i_1, \ldots, i_k\}$ and $I' = \{i_1, \ldots, i_{k-1}, i_{k+1}\}$. X is interesting $\Rightarrow I'$ is frequent, i.e., $supp(I') \geq min\_supp$. Also, according to

SA-CAMP of GM, X is interesting $\Rightarrow \{i_1, \ldots, i_{k-1}\}$ is interesting $\Rightarrow I' \setminus \{i_{k+1}\} \in \mathcal{F}_{k-1}$. Therefore, I' is an FIP k-itemset, i.e., $I' \in \mathcal{F}_k \bigcup \mathcal{T}_k$. Furthermore, according to the SA-CAMP of the GM, X is interesting $\Rightarrow I \in \mathcal{F}_k$. As a result, $X \in \mathcal{F}_k \times (\mathcal{F}_k \bigcup \mathcal{T}_k)$, which leads to a contradiction. ■

Now, we remain to ensure that $\mathcal{F}_k \bigcup \mathcal{T}_k$ indeed contains all FIP k-itemsets. To this end, we revisit the generation of $\mathcal{T}_k$ in the pseudocode. From lines 15 and 21 in Fig. 2, $\mathcal{T}_k$ contains the following: 1) the pruned k-itemsets due to the GM upper-bound pruning (line 15) and 2) the candidate k-itemsets that cannot enter $\mathcal{F}_k$ but are frequent (line 21). We therefore have the following lemma.

Lemma 3: $\forall k \in \mathbb{Z}^+,\ k \geq 2,\ \mathcal{F}_k \bigcup \mathcal{T}_k$ contains all FIP k-itemsets.

Proof: We use the mathematical induction.

$\boldsymbol{k = 2}$: Let $\mathcal{P}_2$ denote the set of all FIP 2-itemsets. We should first notice that $\mathcal{F}_1 \times (\mathcal{F}_1 \bigcup \mathcal{T}_1)$ contains all frequent 2-itemsets, so $\mathcal{P}_2 \subseteq \mathcal{F}_1 \times (\mathcal{F}_1 \bigcup \mathcal{T}_1)$. Then, according to the pseudocode in Fig. 2, we have

$$\mathcal{F}_1 \times \left(\mathcal{F}_1 \bigcup \mathcal{T}_1\right) = \mathcal{S}_2 \bigcup \mathcal{U}_2 \bigcup \mathcal{C}_2 \tag{5}$$

where $\mathcal{S}_2$ is the set of all deleted 2-itemsets that have infrequent subsets (lines 9–11) and $\mathcal{U}_2$ is the set of all the 2-itemsets deleted due to the upper-bound pruning (lines 12–16). Furthermore, according to the pseudocodes, $\mathcal{U}_2 \subseteq \mathcal{T}_2$, and $\mathcal{C}_2 \subseteq \mathcal{F}_2 \bigcup \mathcal{T}_2$, so we have

$$\mathcal{F}_1 \times \left(\mathcal{F}_1 \bigcup \mathcal{T}_1\right) = \mathcal{S}_2 \bigcup \left(\mathcal{F}_2 \bigcup \mathcal{T}_2\right). \tag{6}$$

Since $\mathcal{P}_2 \bigcap \mathcal{S}_2 = \varnothing$ and $\mathcal{P}_2 \subseteq \mathcal{F}_1 \times (\mathcal{F}_1 \bigcup \mathcal{T}_1)$, we have $\mathcal{P}_2 \subseteq \mathcal{F}_2 \bigcup \mathcal{T}_2$. The lemma holds.

Suppose that $\mathcal{F}_k \bigcup \mathcal{T}_k$ contains all FIP k-itemsets. We proceed to the $\boldsymbol{k+1}$ **case**. Assume that there is an FIP $(k+1)$-itemset $X = \{i_1, \ldots, i_{k+1}\} \notin \mathcal{F}_{k+1} \bigcup \mathcal{T}_{k+1}$. Let $I' = \{i_1, \ldots, i_{k-1}, i_{k+1}\}$ and $I = \{i_1, \ldots, i_k\}$. X is frequent $\Rightarrow I'$ is frequent. Also, I is interesting $\Rightarrow I' \setminus \{i_{k+1}\}$ is interesting. Thus, I' is an FIP k-itemset, i.e., $I' \in \mathcal{F}_k \bigcup \mathcal{T}_k$. Recalling that $I \in \mathcal{F}_k$, we have $X \in \mathcal{F}_k \times (\mathcal{F}_k \bigcup \mathcal{T}_k)$. Similar to the $k = 2$ case, we have

$$\mathcal{F}_k \times \left(\mathcal{F}_k \bigcup \mathcal{T}_k\right) = \mathcal{S}_{k+1} \bigcup \left(\mathcal{F}_{k+1} \bigcup \mathcal{T}_{k+1}\right) \tag{7}$$

where $\mathcal{S}_{k+1}$ is the set of all deleted $(k+1)$-itemsets that have infrequent subsets. X is interesting $\Rightarrow X \notin \mathcal{S}_{k+1}$, so we have $X \in \mathcal{F}_{k+1} \bigcup \mathcal{T}_{k+1}$, which leads to a contradiction. Thus, the lemma holds for the $k+1$ case. We therefore complete the proof. ■

During the generation of the candidate itemsets, two points are noteworthy, as indicated by the underlined pseudocodes in Fig. 2. First, in line 9, we check the supports of the immediate subsets of each candidate for the purpose of deleting infrequent candidates. However, due to the pruning effects of the GM threshold and the GM upper bound, GMiner may not record the supports of *all* the subsets. Therefore, GMiner only checks part of the subsets and deletes the candidate if it finds any infrequent subset. Second, in line 15, $\mathcal{T}_k$ is used to contain the candidates deleted by the GM upper-bound pruning. Although the pseudocodes here check the upper-bound value of *each* candidate, GMiner actually employs the antimonotonicity of the upper bound to save the computational costs. When the number of items is huge, the computation saving can be quite considerable. We will show this in the experimental section.

TABLE II
SOME CHARACTERISTICS OF DATA SETS

Data set	#Item	#Record	Avg.Length	Density	Source
Chess	75	3196	37.0	0.4933	UCI†
Pumsb_star	2088	49046	50.5	0.0242	IBM‡
Product	14462	57671	7.2	0.0005	company
Retail	16470	88162	10.3	0.0006	Belgian‡
La1	29714	3204	151.1	0.0051	TREC§

†: Available at http://archive.ics.uci.edu/ml/index.html.
‡: Available at http://fimi.cs.helsinki.fi/data/.
§: Available at http://trec.nist.gov.

C. Completeness and Correctness

The GMiner algorithm is complete—it never misses any interesting pattern. The evidences are as follows: 1) The SA-SET guarantees that the search space is complete, i.e., every itemset can be traversed (refer to Section V-C), and 2) the candidate generation strategy $\mathcal{F}_k \times (\mathcal{F}_k \bigcup \mathcal{T}_k)$ is complete (refer to Section VI-B).

The GMiner algorithm is correct—every itemset found by GMiner is guaranteed to have GM and support values above the thresholds. This is because GMiner computes the support and GM values for every candidate (refer to Section VI-B).

In general, GMiner is an Apriori-like algorithm, which has the computational complexity similar to the Apriori algorithm (for more details about the computational complexity of the Apriori algorithm, please refer to [24, Ch. 6, p. 349]). The extra cost of GMiner may be due to the following: 1) the use of $\mathcal{F}_k \times (\mathcal{F}_k \bigcup \mathcal{T}_k)$ strategy instead of the $\mathcal{F}_k \times \mathcal{F}_k$ strategy; 2) the extra interestingness computation for candidate patterns; and 3) the extra interestingness upper-bound computation for candidate patterns. Among them, only the first extra cost may have substantial impact on GMiner, since it may require additional scans on the entire database. Nonetheless, the introduction of the interestingness threshold and the upper-bound pruning scheme can help dramatically reduce the number of candidate patterns, particularly when a small support threshold is set to find rare patterns. That is why GMiner often shows much higher efficiency than the *postevaluation* scheme in the experimental results.

VII. EXPERIMENTAL RESULTS

In this section, we study the performances of GMiner on real-world data sets. Specifically, we will show the following: 1) the pruning effect of SA-CAMP; 2) the pruning effect of the GM upper bound; 3) the cross-support property of GM; and 4) the interestingness of the patterns found by GMiner.

A. Experimental Setup

In our experiments, we use five real-world data sets whose general characteristics are summarized in Table II. Note that

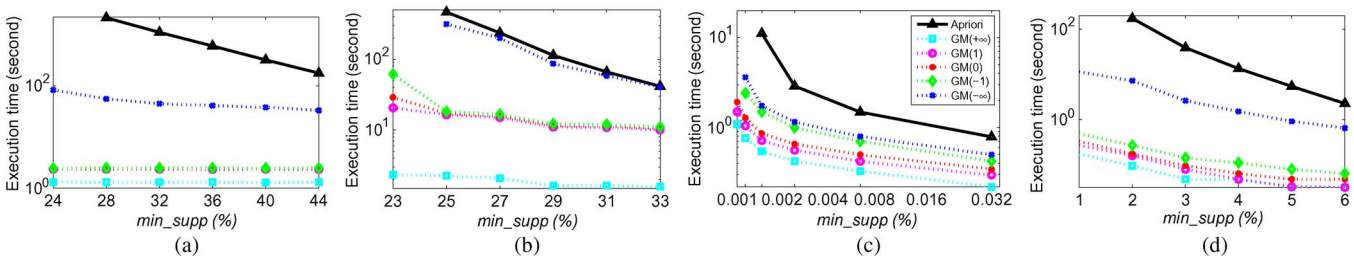

Fig. 3. Execution time comparison along min_supp. (a) *Chess*: $min\_int = 0.75$. (b) $Pumsb'$: $min\_int = 0.75$. (c) *Retail*: $min\_int = 0.5$. (d) *La1*: $min\_int = 0.5$.

the "Density" attribute describes the ratio of nonzero entries of each data set.

As we know, GMiner is an Apriori-like algorithm that performs the candidate generate-and-test approach and breadth-first search strategy. Therefore, to evaluate the efficiency of GMiner, the traditional Apriori algorithm with the *postevaluation* should be used as the benchmark for the comparison. Our implementations of GMiner and the Apriori algorithm are based on the "Apriori" source codes provided by Borgelt [3].[1] We modified the codes so as to incorporate the GM measure as follows: 1) an *in-evaluation* measure for GMiner and 2) a *postevaluation* measure for the classic Apriori algorithm, respectively. For simplicity, hereinafter, we use "Apriori" to denote the *postevaluation* implementation of the Apriori algorithm if no confusion happens. Finally, it is noteworthy that, for GMiner, we use the upper-bound pruning strategy only for the candidate 2-itemsets.

All the algorithms were coded in C, built by Microsoft Visual C++ 2008, and run on a Microsoft Windows 7 Ultimate platform. The experimental PC is ThinkPad T400, with an Intel P8700 CPU and a 4-GB DDRIII 1066-MHz RAM.

B. Efficiency of GMiner

In this section, we study the efficiency of GMiner from various perspectives. We first compare the performances of GMiner and Apriori. Then, we explore the pruning effect of SA-CAMP by looking into the mining process. Third, we study the pruning effect of the upper bound of GM. Finally, we compare the efficiency between SA-CAMP and AMP.

1) Comparison of the Overall Performances: Fig. 3 shows the experimental results on the four data sets. We set $min\_int = 0.75$ for *Chess* and *Pumsb_star* and $min\_int = 0.50$ for *Retail* and *La1*, with the purpose of finding more interesting patterns. The range of the minimum support threshold is also set carefully to meet the very characteristic of each data set.

As can be seen in Fig. 3, for all the four data sets, GMiner consumes much less execution time than Apriori. This means that GMiner dominates Apriori in terms of the efficiency. Furthermore, it is more interesting to note that, with the decrease of the support values, the execution time of GMiner increases much more slowly than that of Apriori, which therefore results in the bigger gaps. For the GM family, the bigger the exponent p is, the less the execution time consumes, which is in accordance with the inherent order of the measures. Also note that, in the extreme cases, for example, $min\_supp < 25\%$ for *Chess* and $min\_supp < 0.004\%$ for *Retail*, Apriori crashed. Moreover, in the case of $min\_supp < 0.002\%$ for *Retail*, GMiner with $p = -\infty$ and $p = -1$ also crashed due to the unavailability of new memory. These led to the missing points for the lines in Fig. 3.

We further study the impact of the interestingness threshold on GMiner and Apriori. Fig. 4 shows the results on *Chess* and *Retail*. As can be seen, with the increase of the threshold, the execution time of GMiner drops sharply. On the contrary, Apriori is completely insensitive to the threshold—the execution time of Apriori remains unchanged along different threshold levels.

[1] Available at http://www.borgelt.net.

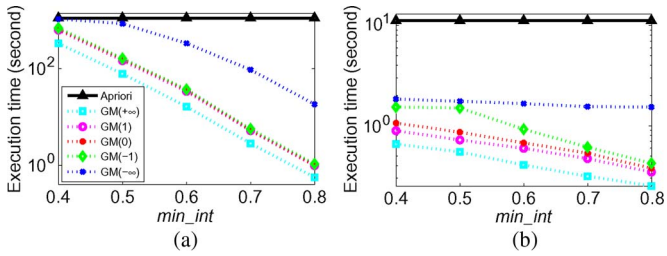

Fig. 4. Execution time comparison along min_int. (a) *Chess*: $min\_supp = 32\%$. (b) *Retail*: $min\_supp = 0.004\%$.

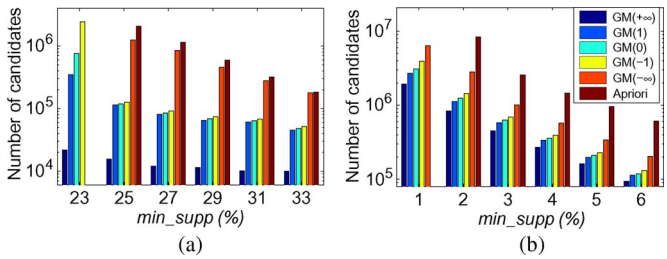

Fig. 5. Comparison of candidate numbers. (a) $Pumsb'$: $min\_int = 0.75$. (b) *La1*: $min\_int = 0.5$.

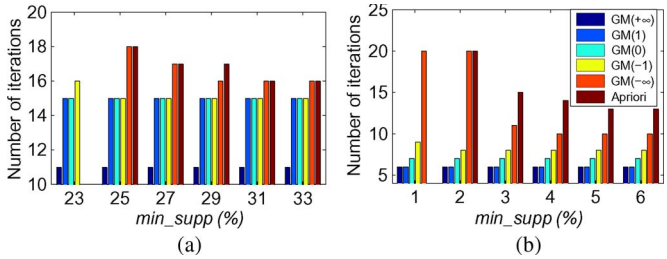

Fig. 6. Comparison of iteration numbers. (a) $Pumsb'$: $min\_int = 0.75$. (b) *La1*: $min\_int = 0.5$.

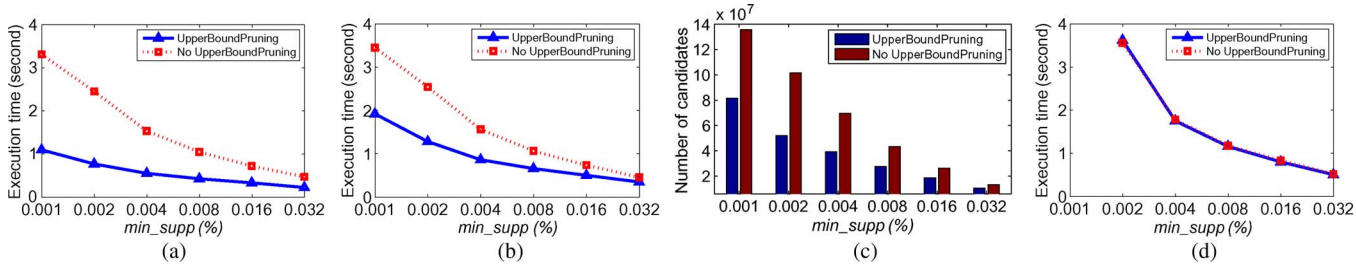

Fig. 7. Upper-bound pruning effect on *Retail*. (a) $min\_int = 0.5$; $GM(+\infty)$. (b) $min\_int = 0.5$; $GM(0)$. (c) $min\_int = 0.5$; $GM(0)$. (d) $min\_int = 0.5$; $GM(-\infty)$.

This observation indeed justifies the merit of having GM as an *in-evaluation* measure rather than a *postevaluation* measure.

2) Pruning Effect of SA-CAMP: The pruning effect of SA-CAMP lies in two aspects. First, it may help reduce the number of candidate itemsets by the early pruning of uninteresting itemsets. Second, it may reduce the number of iterations needed to find the frequent long patterns. These long patterns may be uninteresting according to the given interestingness threshold and, therefore, can be pruned in the early iterations.

Fig. 5 shows the numbers of candidate itemsets when applying GMiner and Apriori on the *Pumsb_star* and *La1* data sets, respectively. Note that the missing bars are due to "out of memory" given such small support thresholds. As can be seen in the figure, for both cases, the numbers of candidates are reduced substantially by GMiner.

Fig. 6 shows the numbers of iterations when applying GMiner and Apriori on the aforementioned two data sets, respectively. As can be seen in the figure, the numbers of iterations of the data sets are substantially reduced by GMiner. It is also interesting to note that, since there are many long patterns in *Pumsb_star*, the iteration number reduction for *Pumsb_star* is less effective than that for *La1*. Nevertheless, recall that *Pumsb_star* enjoys a significant reduction of candidates in Fig. 5(a) and the overall performance of GMiner on *Pumsb_star* is still considered fairly good, as shown in Fig. 3(b).

In summary, by using GM as an *in-evaluation* measure in GMiner, the interesting pattern mining efficiency can be improved substantially. This is even more remarkable for the case when we need to set a small support threshold to find some real interesting but rare patterns. As the key factor for this high efficiency, the SA-CAMP possessed by GM shows advantages in reducing the numbers of candidates and iterations during the mining process.

3) Pruning Effect of the GM Upper Bound: Here, we study the pruning effect of the GM upper bound. The sparse date set *Retail*, which contains multiple items, is used here. Fig. 7 shows the results by GMiner with and without upper-bound pruning for candidate 2-itemsets.

As can be seen in the figure, for $GM(+\infty)$ and $GM(0)$, both the reductions of the execution time and the number of candidate 2-itemsets are considerable by using the upper-bound pruning, as shown in Fig. 7(a)–(c). Furthermore, as the support threshold goes down, the advantage of the upper-bound pruning tends to be more significant. However, for the $GM(-\infty)$ case in Fig. 7(d), there are no obvious improvements, for the upper-bound pruning effect is too weak for $p = -\infty$.

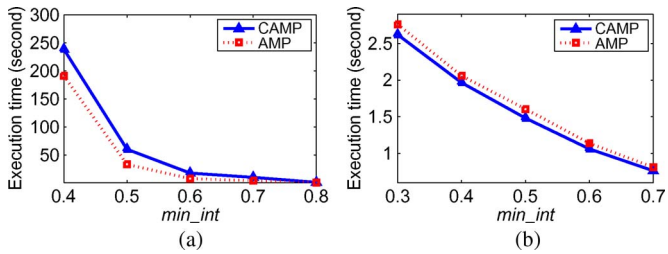

Fig. 8. SA-CAMP versus AMP. (a) *Pumsb_star*: $min\_supp = 25\%$. (b) *Product*: $min\_supp = 0.001\%$.

In summary, the antimonotonicity of the upper bound can be used to speed up the mining process. Its pruning effect can be significant at the presence of multiple items or small support thresholds.

4) SA-CAMP Versus AMP: Here, we compare the pruning efficiency of SA-CAMP and AMP. Since $GM(+\infty)$ holds both SA-CAMP and AMP, we use it with SA-CAMP and AMP, respectively, in GMiner and compare their efficiencies. Fig. 8 shows the results for *Pumsb_star* and *Product*. For dense *Pumsb_star*, using AMP consumes less time than using SA-CAMP, but the difference is small. For sparse *Product*, however, using AMP consumes more time than using SA-CAMP. This is due to the upper-bound pruning when using SA-CAMP in GMiner. In general, AMP has a higher pruning efficiency than SA-CAMP, but in most cases, the gap is very small.

Note that, since $GM(+\infty)$ is also called $AllConf$ in CoMine [15] or h-confidence in hyperclique miner [27], the experiments previously mentioned can also be viewed as the comparison of GMiner (using SA-CAMP) with CoMine or hyperclique miner (using AMP). The results indicate that GMiner is competitive to CoMine or hyperclique miner in terms of computational efficiency. Finally, it is noteworthy that CoMine and hyperclique miner only work for $GM(+\infty)$. In contrast, GMiner provides a flexible framework that can work for the family of GM measures with any p value, which is considered important for diversified real-world applications.

C. Effect of the Cross-Support Property

Here, we study the cross-support property of the GM family. The dense date set *Pumsb_star* and sparse date set *Product* are used here. We define

$$TPR = \frac{the\ number\ of\ deleted\ cross\text{-}support\ patterns}{the\ number\ of\ all\ cross\text{-}support\ patterns}$$

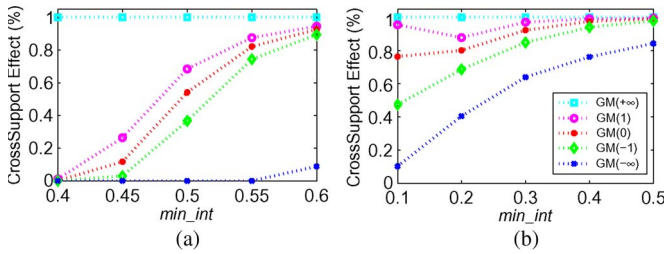

Fig. 9. Effect of cross-support property of GM family. (a) *Pumsb_star*: $min\_supp = 25\%$. (b) *Product*: $min\_supp = 0.01\%$.

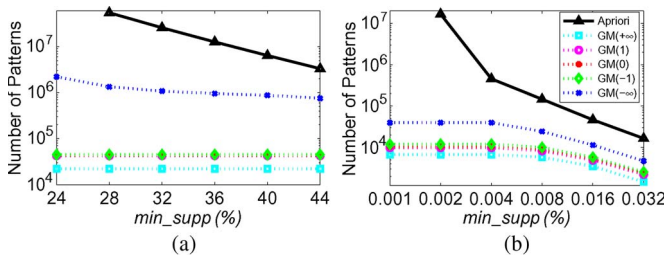

Fig. 10. Impact of min_supp on pattern generation. (a) *Chess*: $min\_int = 0.75$. (b) *Product*: $min\_int = 0.5$.

which indicates the strength of the cross-support property of a measure. Let the cross-support threshold $\theta = min\_int$; we observe the TPR values along different min_int (θ) values.

As can be seen in Fig. 9, each measure of the GM family has the effect of the cross-support property, although the strengths are quite different. $GM(+\infty)$ has the strongest effect—all the cross-support patterns are filtered out by $GM(+\infty)$, i.e., $TPR = 1$. Moreover, $GM(-\infty)$ has the weakest effect, particularly for the dense data set *Pumsb_star*. However, with the increase of the interestingness threshold, the gaps of the strengths narrow quickly.

D. Interestingness of the Patterns Found

In this section, we examine the interestingness of the patterns found by GMiner. Our validation focuses on two aspects: 1) the number of interesting patterns generated and 2) the real meaning of the interesting patterns.

First, we study the number of interesting patterns found by the GM family. Fig. 10 shows the experimental results on data sets *Product* and *Chess*. Note that, when $min\_int = 0$, GMiner reduces to the classic Apriori algorithm, as indicated by "Apriori" in the figure. As can be seen in Fig. 10, Apriori generates the most patterns, the next one is $GM(-\infty)$, and the rest of the measures act closely to one another. For instance, for the *Product* data set, when $min\_int = 0.5$ and $min\_supp = 0.002\%$, the number of patterns generated by Apriori reaches 10^7, among which only 6640 patterns are interesting if $GM(+\infty)$ is used instead.

One may argue that we can also reduce uninteresting patterns by improving the support threshold. However, by doing so, we will take the risk of missing many really interesting but rarely occurring patterns. Table III shows such examples found by GMiner with $p = 0$ in *La1* and *Product*. It can be seen that these patterns contain closely related words or products at a very low support level. For example, in the *La1* data set, the pattern {nagorno, azerbaijan, karabakh, armenian} is about the Nagorno–Karabakh Republic, which is de jure part of Azerbaijan. For another example, in the *Product* data set, the pattern {bumper pad, diaper, crib sheet, comftr} is about baby products. These patterns occur rarely in the transactions and, therefore, cannot be identified directly using the standard frequent pattern mining algorithms. However, with the GM-enabled GMiner, they can be found quickly and easily.

TABLE III
SOME PATTERNS FOUND BY $GM(0)$

Interesting Pattern	Support
From Data Set La1	
{nagorno, azerbaijan, karabakh, armenian}	0.001
{zurbriggen, pirmin, downhill, switzerland}	0.001
{mansdorf, edberg, steffi, bonk, shriver, stefan, lendl}	0.001
{najibullah, kabul, afghan}	0.002
{yitzhak, gaza, palestinian, israe, occupi}	0.004
From Data Set Product	
{nokia battery, nokia adapter, nokia wireless phone}	0.00049
{earrings, gold ring, bracelet}	0.00019
{sham, pillow, valance, curtains}	0.00031
{bumper pad, diaper, crib sheet, comftr}	0.00028
{bath set, sheet bath, shower, hunter bath, towel set}	0.00001

VIII. CONCLUSION

In this paper, we have studied the problem of mining interesting but infrequent patterns from large-scale multi-item databases using the GM measure. Specifically, we first made a comprehensive study on 35 interestingness measures by setting three necessary conditions. Then, we proved that the GM family possesses SA-CAMP and some appealing properties and, therefore, can be used as an *in-evaluation* measure in mining interesting patterns. Also, we have proposed the GM upper bound to further prune candidate itemsets. These two pruning strategies were implemented in an Apriori-like algorithm: GMiner. The experimental results show that GMiner is much more efficient than the *postevaluation* algorithm and can find real interesting patterns with support at extremely low levels.

APPENDIX
CROSS-SUPPORT PROPERTY OF GM

In real-world applications, many market-basket data have inherently skewed support distributions which often lead to the "cross-support patterns" [26], [27]. These patterns typically represent spurious associations among items with substantially different support levels. In [27], Xiong *et al.* proved that $AllConf$ had the "cross-support property" and can help to eliminate cross-support patterns. We revisit this from the perspective of the whole GM family.

Definition 4 (Cross-Support Pattern): Given $\theta \in (0, 1)$, X is a cross-support pattern w.r.t. θ if X contains two items x and y such that $supp(\{x\})/supp(\{y\}) < \theta$.

Lemma 4: Any cross-support pattern X w.r.t. a threshold $\theta \in (0, 1)$ is guaranteed to have $GM(X, p) < \sqrt[K]{\theta}$, where $K = |X|$ and $p \geq 0$.

Proof: Without loss of generality, assume that $supp(X_1)/supp(X_2) < \theta$. Since $p \geq 0$, we have

$$\begin{aligned} GM(X,p) \leq cos(X) &= \frac{supp(X)}{\sqrt[K]{\prod_{k=1}^{K} supp(X_k)}} \\ &\leq \sqrt[K]{\frac{supp(X_1)^2 \prod_{k=3}^{K} supp(X_k)}{\prod_{k=1}^{K} supp(X_k)}} \\ &= \sqrt[K]{\frac{supp(X_1)}{supp(X_2)}} < \sqrt[K]{\theta}. \end{aligned} \tag{8}$$

We thus complete the proof. ■

Lemma 4 finds a simple GM upper bound for a cross-support pattern. This implies that, if θ is small, a cross-support pattern tends to have a small GM value and may probably be pruned as an uninteresting pattern. In other words, the GM measure has the power of eliminating suspicious cross-support patterns. Note that, however, since the scaling is not strict, the GM upper bound is loose—for instance, $GM(X, +\infty)$ indeed has a much tighter upper bound: θ. Therefore, to know the exact effect of the cross-support property of GM via this upper bound is not feasible. We leave it to the experimental studies.

REFERENCES

[1] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. Int. Conf. Very Large Data Bases*, 1994, pp. 487–499.

[2] J. Blanchard, F. Guillet, R. Gras, and H. Briand, "Using information-theoretic measures to assess association rule interestingness," in *Proc. 5th IEEE Int. Conf. Data Mining*, Houston, TX, 2005, pp. 66–73.

[3] C. Borgelt, "Recursion pruning for the apriori algorithm," in *Proc. Workshop FIMI*, Melbourne, FL, 2003.

[4] S. Brin, R. Motwani, and C. Silverstein, "Beyond market basket: Generalizing association rules to correlations," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Tucson, AZ, 1997, pp. 265–276.

[5] J. Cheng, Y. Ke, A. Fu, J. Yu, and L. Zhu, "Finding maximal cliques in massive networks by h*-graph," in *Proc. SIGMOD*, 2010, pp. 447–458.

[6] Q. Ding, Q. Ding, and W. Perrizo, "PARM—An efficient algorithm to mine association rules from spatial data," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 6, pp. 1513–1524, Dec. 2008.

[7] W. DuMouchel and D. Pregibon, "Empirical Bayes screening for multi-item associations," in *Proc. ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining*, 2001, pp. 67–76.

[8] W. Fan, K. Zhang, H. Cheng, J. Gao, X. Yan, J. Han, P. S. Yu, and O. Verscheure, "Direct mining of discriminative and essential frequent patterns via model-based search tree," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining*, 2007, pp. 230–238.

[9] J. Han, H. Cheng, D. Xin, and X. F. Yan, "Frequent pattern mining: Current status and future directions," *Data Mining Knowl. Disc.*, vol. 15, no. 1, pp. 55–86, Aug. 2007.

[10] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, 2nd ed. San Mateo, CA: Morgan Kaufmann, 2005.

[11] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Berlin, Germany: Springer-Verlag, 2001.

[12] R. J. Hilderman and H. J. Hamilton, *Knowledge Discovery and Measures of Interest*. Norwell, MA: Kluwer, 2001.

[13] L. Lakshmanan, J. Pei, and J. Han, "Mining frequent itemsets with convertible constraints," in *Proc. Int. Conf. Data Eng.*, 2001, pp. 433–442.

[14] Y. Ke, J. Cheng, and J. Yu, "Efficient discovery of frequent correlated subgraph pairs," in *Proc. ICDM*, 2009, pp. 239–248.

[15] Y.-K. Lee, W.-Y. Kim, Y. D. Cai, and J. Han, "Comine: Efficient mining of correlated patterns," in *Proc. 3rd IEEE Int. Conf. Data Mining*, Melbourne, FL, 2003, pp. 581–584.

[16] P. Lenca, P. Meyer, B. Vaillant, and S. Lallich, "On selecting interestingness measures for association rules: User oriented description and multiple criteria decision aid," *Eur. J. Oper. Res.*, vol. 184, no. 2, pp. 610–626, 2008.

[17] K. McGarry, "A survey of interestingness measures for knowledge discovery," *Knowl. Eng. Rev.*, vol. 20, no. 1, pp. 39–61, Mar. 2005.

[18] E. Omiecinski, "Alternative interest measures for mining associations," *IEEE Trans. Knowl. Data Eng.*, vol. 15, no. 1, pp. 57–69, Jan./Feb. 2003.

[19] G. Piatetsky-Shapiro, "Notes of AAAI'91 workshop knowledge discovery in databases," in *Proc. AAAI Knowl. Disc. Databases Workshop*, Anaheim, CA, 1991, pp. 174–185.

[20] R. Rymon, "Search through systematic set enumeration," in *Proc. 3rd Int. Conf. Principles Knowl. Represent. Reason.*, 1992, pp. 268–275.

[21] A. Silberschatz and A. Tuzhilin, "What makes patterns interesting in knowledge discovery systems," *IEEE Trans. Knowl. Data Eng.*, vol. 8, no. 6, pp. 970–974, Dec. 1996.

[22] S. Stanley and H. Sankappanavar, *A Course in Universal Algebra*. Berlin, Germany: Springer-Verlag, 1981.

[23] P.-N. Tan, V. Kumar, and J. Srivastava, "Selecting the right interestingness measure for association patterns," in *Proc. SIGKDD*, Edmonton, AB, Canada, 2002, pp. 32–41.

[24] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Reading, MA: Addison-Wesley, 2005.

[25] T. Wu, Y. Chen, and J. Han, "Association mining in large databases: A re-examination of its measures," in *Proc. 11th Eur. Conf. Principles Pract. Knowl. Disc. Databases*, 2007, pp. 621–628.

[26] H. Xiong, S. Shekhar, P.-N. Tan, and V. Kumar, "Exploiting a support-based upper bound of Pearson's correlation coefficient for efficiently identifying strongly correlated pairs," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining*, 2004, pp. 334–343.

[27] H. Xiong, P.-N. Tan, and V. Kumar, "Hyperclique pattern discovery," *Data Mining Knowl. Disc.*, vol. 13, no. 2, pp. 219–242, Sep. 2006.

[28] S. Zhu, J. Wu, H. Xiong, and G. Xia, "Scaling up top-k cosine similarity search," *Data Knowl. Eng.*, vol. 70, no. 1, pp. 60–83, Jan. 2011.

[29] X. Zhu, "Quantitative association rules," in *Encyclopedia of Database Systems*. New York: Springer-Verlag, 2008.

Junjie Wu (M'10) received the B.E. degree in civil engineering and the Ph.D. degree in management science and engineering in 2008 from Tsinghua University, Beijing, China.

He is currently an Associate Professor with the Information Systems Department, School of Economics and Management, Beihang University, Beijing, where he is also the Laboratory Director. His general area of research is data mining and complex networks. He is currently the Principal Investigator of three National Natural Science Foundation of China projects and one Ministry of Education project and has published over 35 papers in refereed conference proceedings and journals in data mining area.

Dr. Wu is a member of Association for Information Systems and Association for Computing Machinery. He was the recipient of the 2011 Program for New Century Excellent Talents in University and the 2010 National Excellent Doctoral Dissertation Award, both in China.

Shiwei Zhu received the B.E. degree in information management and information system and the Ph.D. degree in management science and engineering from Beihang University, Beijing, China, in 2004 and 2011, respectively.

He is currently a Lecturer with the School of Information, Central University of Finance and Economics, Beijing. His research interests include data mining, management information system, and electronic commerce.

Hui Xiong (SM'07) received the B.E. degree from the University of Science and Technology of China, Hefei, China, the M.S. degree from the National University of Singapore, Singapore, Singapore, and the Ph.D. degree from the University of Minnesota, Minneapolis.

He is currently the Vice Chair of the Management Science and Information Systems Department, Rutgers, The State University of New Jersey, New Brunswick, where he is also an Associate Professor. He is a Coeditor of Clustering and Information Retrieval (Kluwer Academic Publishers, 2003) and a Coeditor-in-Chief of Encyclopedia of GIS (Springer, 2008). He is an Associate Editor of the *Knowledge and Information Systems* journal. His general area of research is data and knowledge engineering, with a focus on developing effective and efficient data analysis techniques for emerging data-intensive applications. He has published over 100 technical papers in peer-reviewed journals and conference proceedings.

Dr. Xiong is a Senior Member of the ACM. He has served regularly in the organization committees and the program committees of a number of international conferences and workshops.

Jian Chen (F'08) received the B.Sc. degree in electrical engineering and the M.Sc. and Ph.D. degrees in systems engineering from Tsinghua University, Beijing, China, in 1983, 1986, and 1989, respectively.

He is currently the Chairman of the Department of Management Science and Engineering, Tsinghua University, where he is also the Lenovo Chair Professor and the Director of the Research Center for Contemporary Management. He is the Editor of the "Journal of Systems Science and Systems Engineering" and the Area Editor or Associate Editor of many prestigious journals in his areas. He has published over 150 papers in refereed journals and has been the Principal Investigator for over 30 grants or research contracts with the National Science Foundation of China, governmental organizations, and companies. His main research interests include supply chain management, electronic commerce, and decision support systems.

Dr. Chen was a recipient of the Fudan Management Excellence Award and the IBM Faculty Award and was recognized as a Ministry of Education Changjiang Scholar. He serves as a Regional Vice-President (VP) of the Production and Operations Management Society, the Chairman of the Service Systems and Organizations Technical Committee of the IEEE Systems, Man, and Cybernetics Society, and the VP or a steering member of many research societies in China.

Jianming Zhu received the Ph.D. degree in computer application technology from Xidian University, Xi'an, China.

He is currently a Full Professor with the School of Information, Central University of Finance and Economics, Beijing, China. His research interests include information security, cryptography, and electronic commerce.