Efficient Discovery of the Most Interesting Associations

GEOFFREY I. WEBB, Monash University
JILLES VREEKEN, University of Antwerp

Self-sufficient itemsets have been proposed as an effective approach to summarizing the key associations in data. However, their computation appears highly demanding, as assessing whether an itemset is self-sufficient requires consideration of all pairwise partitions of the itemset into pairs of subsets as well as consideration of all supersets. This article presents the first published algorithm for efficiently discovering self-sufficient itemsets. This branch-and-bound algorithm deploys two powerful pruning mechanisms based on upper bounds on itemset value and statistical significance level. It demonstrates that finding top-*k* productive and nonredundant itemsets, with postprocessing to identify those that are not independently productive, can efficiently identify small sets of key associations. We present extensive evaluation of the strengths and limitations of the technique, including comparisons with alternative approaches to finding the most interesting associations.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*Data mining*

General Terms: Algorithms

Additional Key Words and Phrases: Association mining, itemset mining, interestingness, statistical association mining

ACM Reference Format:
Geoffrey I. Webb and Jilles Vreeken. 2014. Efficient discovery of the most interesting associations. ACM Trans. Knowl. Discov. Data 8, 3, Article 15 (May 2014), 31 pages.
DOI: http://dx.doi.org/10.1145/2601433

1. INTRODUCTION

Association discovery [Agrawal et al. 1993] is one of the most researched topics in data mining. However, the fielded applications appear to be relatively few. It has been suggested that this is due to the susceptibility of conventional association discovery techniques to finding large numbers of associations that are unlikely to be interesting to the user [Webb 2010]. Even with measures such as requiring rules to be *nonredundant* [Bastide et al. 2000; Zaki 2000], *closed* [Bastide et al. 2000; Zaki and Hsiao 2002], *derivable* [Calders and Goethals 2002], and *productive* [Webb 2007], extraordinarily large numbers of associations can be discovered. Webb [2011] has proposed six principles for identifying associations that are unlikely to be of interest:

(1) A conjunction of terms is unlikely to be interesting if its frequency can be predicted by assuming independence between any partition thereof.
(2) A conjunction of terms is unlikely to be interesting if a proper subset y of those terms contains a term i such that i subsumes $y \setminus i$.

This work is supported by the Australian Research Council under grant DP120100553.
Authors' addresses: G. I. Webb, Faculty of Information Technology, Monash University, Clayton, Vic, 3800, Australia; email: geoff.webb@monash.edu; J. Vreeken, Department of Mathematics and Computer Science, University of Antwerp, Middelheimlaan 1, 2020 Antwerp, Belgium; email: Jilles.Vreeken@uantwerp.be.

DOI: http://dx.doi.org/10.1145/2601433

(3) A conjunction of terms is unlikely to be interesting if its frequency can be predicted by the frequency of its specializations.
(4) Appropriate statistical testing should be used to assess Principles 1 and 3.
(5) Measures of interest for a conjunction of terms should measure the smallest deviation from independence between any two partitions of the conjunction.
(6) If a conjunction of terms is unlikely to be interesting, then any rule composed from those terms is unlikely to be interesting.

This article presents efficient algorithms to discover positive associations using these principles and conducts a case study to explore their credibility.

This association discovery task is particularly challenging, as the objective function is very expensive to calculate, requiring assessment of every binary partition of an itemset, and may be neither monotone nor antimonotone. The primary contributions of this paper are

(1) a new bound on well-behaving measures of interest;
(2) a new variant of the OPUS search algorithm, which uses the new bound for the specific task of discovering *self-sufficient itemsets*;
(3) detailed experimental investigation of the algorithm and the application of the above bounds therein, including
    —analysis of the types of measures for which the new bound is tight;
    —analysis of the types of measure of interest for which bounds on a Fisher Exact Test of independence are useful; and
    —the first experimental comparison of self-sufficient itemsets with alternative approaches to finding succinct summaries of the most interesting associations.

The article is organized as follows. Section 2 describes and motivates the self-sufficient itemset discovery problem. Section 3 defines the measures of interest that we will use. Section 4 derives a new constraint on well-behaving measures of interest. Section 5 presents OPUS Miner, an efficient algorithm that can find top-k itemsets within the challenging constraints of self-sufficient itemset discovery. Section 6 presents a complexity analysis. Section 7 discusses related research. Section 8 evaluates the efficiency of the algorithm on the FIMI datasets, including ablation studies assessing key elements of the algorithm and analysis of the types of measure of interest for which bounds on the values of the measure and bounds on a Fisher Exact Test of independence are each useful. It also presents a case study to investigate the credibility of the techniques and performance relative to key alternative approaches. Section 9 discusses directions for future research and presents conclusions.

2. SELF-SUFFICIENT ITEMSETS

Given a domain of items $\mathcal{I}$, an itemset x is a set of items $x \subseteq \mathcal{I}$. A dataset D is a vector of n records $\langle D_1 \ldots D_n \rangle$. Each record D_i is an itemset. For transactional data, items are atomic terms. For attribute-value data, there exists a set of m attributes $A_1 \ldots A_m$, each attribute A_i has a domain of values $\mathrm{dom}(A_i)$, each item is an attribute-value pair denoted as $A_i = v$ where $v \in \mathrm{dom}(A_i)$, and each record D_i contains at most one item for each attribute. The index for a record is called a transaction identifier (TID). The *cover* of an itemset s is the set of TIDs for records that contain s,

$$\mathrm{cov}(s, D) = \{i : 1 \le i \le |D| \wedge s \subseteq D_i\}. \quad (1)$$

In this and the following terms, the database D is not specified when it is apparent from context.

The *support* of an itemset s is the proportion of records in dataset D of which s is a subset:

$$\sup(s, D) = |\operatorname{cov}(s, D)|/n. \tag{2}$$

We use $|\cdot|$ to denote the cardinality of a set. We define the *count* as the number of records of which s is a subset,

$$\#(s, D) = |\operatorname{cov}(s, D)|. \tag{3}$$

For notational convenience, we use

$$\#(s, t, D) = |\operatorname{cov}(s \cup t, D)| \tag{4}$$

to represent the number of records covered by both s and t,

$$\#(\neg s, D) = n - \#(s, D) \tag{5}$$

to represent the number of records of which s is not a subset, and

$$\#(\neg s, \neg t, D) = n - \#(s, D) - \#(t, D) + \#(s, t, D) \tag{6}$$

to represent the number of records that contain neither s nor t.

We use $\mathrm{P}(R \supseteq s)$ to represent the probability that a record drawn uniformly at random from the distribution from which D is drawn will contain itemset s and $\mathrm{P}(R \not\supseteq s)$ the probability that it will not contain s.

Two or more items are positively associated in any meaningful sense of the term only if they occur together more frequently than they would if they were statistically independent of one another. Thus, for any itemset x to be identified as a positive association, we might think that we should require

$$\mathrm{P}(R \supseteq x) > \prod_{i \in x} \mathrm{P}(R \supseteq \{i\}). \tag{7}$$

Webb [2010] argues that positive association between items in an itemset x is unlikely to be interesting, even if x satisfies (7), unless x is *self-sufficient*. To be self-sufficient, an itemset must be *productive*, *nonredundant* and *independently productive*.

2.1. Productivity

An itemset is only productive if all its partitions into two subsets of items are positively correlated with each other. This prevents massive inflation of the number of itemsets discovered due to the addition to each productive itemset x of items that are statistically independent of the items in x and due to joining multiple itemsets irrespective of whether they interact with one another. For example, if *smoking* is associated with *cancer* then *smoking* and *age-is-even* will also be associated with *cancer*, unless *age-is-even* actively reduces the probability of *smoking* or *cancer*. Further, being pregnant is associated with edema, so {*smoking, cancer, pregnant, edema*} satisfies (7) as long as *smoking* and *cancer* do not reduce the probability of *pregnant* and *edema*. Nonetheless, these types of itemsets are unlikely to be interesting. Thus, we should require itemsets be productive:

$$\mathrm{P}(R \supseteq x) > \max_{y \subsetneq x}(\mathrm{P}(R \supseteq y)\mathrm{P}(R \supseteq x \setminus y)). \tag{8}$$

This is the itemset equivalent of the productivity constraint on association rules [Webb 2006].

As we cannot directly determine the probabilities from the sample, we need to perform a statistical test. We use the Fisher Exact Test. When using dataset D to test whether to accept that itemsets x and y are positively correlated, the p-value is

$$p_{\mathrm{F}}(x, y, D) = \sum_{i=0}^{\omega} \frac{\binom{\#(x,D)}{\#(x,y,D)+i}\binom{\#(\neg x,D)}{\#(\neg x,y,D)+i}}{\binom{n}{\#(y,D)}}, \tag{9}$$

where $\omega = \min(\#(x, \neg y, D), \#(\neg x, \neg y, D))$.

To test for productivity, this test must be passed for every partition of itemset s,

$$p_P(s, D) = \max_{x \subsetneq s}(p_F(x, s \backslash x, D)). \tag{10}$$

2.2. Redundancy

Redundancy provides another filter that can discard itemsets that are unlikely to be interesting. An itemset x is redundant if and only if it contains a proper subset, y, that contains an item, i, that subsumes $y \setminus i$,

$$\exists i, y : i \in y \wedge y \subsetneq x \wedge \mathrm{cov}(\{i\}) \supseteq \mathrm{cov}(y \setminus i). \tag{11}$$

For example, *female* subsumes *pregnant*, so any association between these two items and any other item i, such as {*female, pregnant, edema*}, even though productive, is unlikely to be interesting, as it is entailed by the subsumption relationship and whatever association exists between the subsumed items and i (in this example {*pregnant, edema*}). The itemsets {*female, pregnant*} and {*pregnant, edema*} are both potentially interesting, but the productive itemset containing all three items is unlikely to be. This is the itemset equivalent of the redundancy constraint for association rules [Bastide et al. 2000; Zaki 2000].

Unfortunately, it is not possible to define a statistical hypothesis test for redundancy, as it is not possible to use $\mathrm{cov}(\{i\}) \subsetneq \mathrm{cov}(y \setminus i)$ in a null hypothesis.

2.3. Independent Productivity

An itemset is only independently productive with respect to a set of itemsets $\mathcal{S}$ if it is nonredundant and productive and its productivity cannot be explained by the productivity of its self-sufficient supersets in $\mathcal{S}$. For example, suppose that the presence of *fuel*, *oxygen*, and *heat* is necessary for *fire* to be present. In this case, every subset of the first three items will be associated with *fire*. But the six itemsets containing *fire* that are subsets of {*fuel*, *oxygen*, *heat*, *fire*}, such as {*oxygen*, *fire*} and {*fuel*, *heat*, *fire*}, are unlikely to be interesting given {*fuel*, *oxygen*, *heat*, *fire*}. We want to present the user with a single itemset as a summary of the key association rather than seven itemsets.

If there is only one productive nonredundant superset s of an itemset x, this assessment is relatively straightforward. We first find the set of items in s that are not in x, $y = s \setminus x$. We then assess whether x is productive when assessed only on transactions that do not include y. For example, to assess whether {*oxygen*, *heat*, *fire*} is independently productive given {*fuel*, *oxygen*, *heat*, *fire*}, we assess whether it is productive in the context of transactions that do not contain *fuel*.

However, we cannot simply apply this constraint separately with respect to each superset. Consider the possibility that rather than containing a single item for fuel, the data instead contain items representing each of several types of fuel, such as *wood* and *gas*. In this case, we might have supersets {*wood*, *oxygen*, *heat*, *fire*} and {*gas*, *oxygen*, *heat*, *fire*}. The itemset {*oxygen*, *heat*, *fire*} will still be productive in the context of transactions that do not include *wood* because of transactions that have alternative forms of fuel. What we need to do is to test that it is productive in the context of transactions that do not contain any form of fuel, in this case either *wood* or *gas*:

$$\mathrm{P}\left(R \supseteq x \mid \bigwedge_{\substack{z \in \mathcal{S} \\ z \supsetneq x}} R \not\supseteq z \backslash x\right) > \max_{y \subsetneq x}\left(\mathrm{P}\left(R \supseteq y \mid \bigwedge_{\substack{z \in \mathcal{S} \\ z \supsetneq x}} R \not\supseteq z \backslash x\right) \mathrm{P}\left(R \supseteq x \backslash y \mid \bigwedge_{\substack{z \in \mathcal{S} \\ z \supsetneq x}} R \not\supseteq z \backslash x\right)\right). \tag{12}$$

To create a statistical test for this, we define the *exclusive domain* of an itemset x with respect to a set of itemsets $\mathcal{S}$ as

$$\operatorname{edom}(x, \mathcal{S}, D) = \{1 \ldots n\} \setminus \bigcup_{\substack{y \in \mathcal{S} \\ y \supsetneq x}} \operatorname{cov}(y \setminus x, D). \tag{13}$$

This is the set of TIDs for records not covered by any of the sets of additional items in any superset of x in $\mathcal{S}$. For example, the exclusive domain of {*oxygen*, *heat*, *fire*} with respect to {{*wood*, *oxygen*, *heat*, *fire*}, {*gas*, *oxygen*, *heat*, *fire*}} is the set of all transactions that do not contain *wood* or *gas*.

We then apply the test for productivity with respect to this domain,

$$p_I(s, D) = \max_{x \subsetneq s}(p_F(x, s \backslash x, \operatorname{edom}(s, \mathcal{S}, D))). \tag{14}$$

Without an independent productivity constraint, a system typically discovers for every maximal interesting itemset large numbers of itemsets that are simply subsets of that one key itemset of interest.

2.4. Statistical Testing

Webb [2007] argues that the large search spaces explored in association discovery result in extreme risks of making false discoveries and that corrections for multiple testing should be applied.

In the current work, we address this issue by using layered critical values [Webb 2008], whereby a critical value of

$$\alpha_{|x|} = \min_{1 \leq i \leq |x|}\left(\frac{\alpha}{2^{i-1}\binom{|\mathcal{I}|}{i}}\right) \tag{15}$$

is used to assess an itemset of size $|x|$ for both the productivity and independent productivity tests. Note that this function is monotonically decreasing with respect to itemset size, which is required to allow pruning using the statistical test.

It is acknowledged that the use of statistical hypothesis testing, such as this, has an inherent limitation that it requires the selection of a *critical value*, an arbitrary bound on the risk that will be tolerated of accepting an association in error.

In the current work, we use the conventional critical value of 0.05.

2.5. Benefits of Self-Sufficient Itemsets

Self-sufficient itemsets have the attractive feature that it is straightforward to generate a simple and readily understood explanation of why any itemset is rejected. This is achieved by identifying which of the three constraints is violated and the itemset(s) with respect to which it is violated. For example, "{`male, poor-vision, prostate-cancer, glasses`} was rejected because it fails a test for independence with respect to {`male, prostate-cancer`} and {`poor-vision, glasses`}."

Associations are most commonly expressed as rules [Agrawal et al. 1993]. However, Webb [2011] argues that for many purposes, itemsets provide a more useful formalism for presenting discovered associations, if for no other reason than to prevent multiple expressions of a single association. If an itemset x is self-sufficient, every $y \subset x$ will be associated with $x \setminus y$. It is credible that any association rule $x \to y$ is unlikely to be interesting unless $x \cup y$ is a self-sufficient itemset. If this is so, then the primary circumstance where association rules may be more useful than itemsets may be when there are particular items of interest to the user. In this case, expressing rules that have consequents restricted to those items allows statistics that express the relationship between the other items and the items of interest to be easily highlighted [Novak et al. 2009].

3. QUANTIFYING INTERESTINGNESS

Many measures have been developed for quantifying association interestingness [Geng and Hamilton 2006; Tew et al. 2014]. Almost all of these relate to rules rather than itemsets. The majority assess the deviation of the support of a rule from the support that would be expected if the antecedent and consequent were independent. We denote such a measure by a function $\mathrm{M} : [0, 1], [0, 1], [0, 1] \to \mathbb{R}$ from the supports of the rule, the antecedent, and the consequent to a real.

Webb [2011] suggests that itemset measures should be developed from a rule measure by selecting the least extreme value that results from applying the measure to any rule that can be created by partitioning the itemset x into an antecedent y and consequent $z = x \setminus y$. We call such an interest measure a min partition measure (MPM). The MPM $\mathcal{M}_{\mathrm{M}}(x)$ of an itemset x using base measure M is

$$\mathcal{M}_{\mathrm{M}}(x) = \min_{y \subsetneq x} (\mathrm{M}(\mathrm{sup}(x), \mathrm{sup}(y), \mathrm{sup}(x \setminus y))). \tag{16}$$

In the current work, we use two such measures, *leverage*

$$\delta(x) = \min_{y \subsetneq x} (\mathrm{sup}\,(x) - \mathrm{sup}\,(y) \times \mathrm{sup}\,(x \setminus y)) \tag{17}$$

and *lift*

$$\gamma(x) = \min_{y \subsetneq x} (\mathrm{sup}(x)/[\mathrm{sup}\,(y) \times \mathrm{sup}\,(x \setminus y)]). \tag{18}$$

4. A BOUND ON MPMS WITH WELL-BEHAVING MEASURES OF INTEREST

We present here a bound on MPMs that use *well-behaving* measures of interest [Piatetsky-Shapiro 1991]. This bound will be useful in the algorithm that we develop in the next section.

We use Hämäläinen's [2010] reformulation of the axioms for well-behaving measures of interest, $\mathrm{M}(\mathrm{sup}(x), \mathrm{sup}(y), \mathrm{sup}(z))$, with respect to an itemset x, and a partition $y \subsetneq x$ and $z = x \setminus y$:

Axiom 1. $\mathrm{M}(\mathrm{sup}(x), \mathrm{sup}(y), \mathrm{sup}(z))$ is minimal, when $\mathrm{sup}(x) = \mathrm{sup}(y) \times \mathrm{sup}(z)$;

Axiom 2. $\mathrm{M}(\mathrm{sup}(x), \mathrm{sup}(y), \mathrm{sup}(z))$ is monotonically increasing with $\mathrm{sup}(x)$, when $\mathrm{sup}(y)$, $\mathrm{sup}(z)$, and n remain unchanged; and

Axiom 3. $\mathrm{M}(\mathrm{sup}(x), \mathrm{sup}(y), \mathrm{sup}(z))$ is monotonically decreasing with $\mathrm{sup}(y)$ (or $\mathrm{sup}(z)$), when $\mathrm{sup}(x)$ and $\mathrm{sup}(z)$ (or $\mathrm{sup}(y)$) remain unchanged.

We use only one upper bound for MPMs with well-behaving measures of interest. When exploring supersets of any itemset x, an upper bound on $\mathcal{M}_{\mathrm{M}}(x')$, where $x' \supseteq x$ is provided by

$$\mathrm{M}\left(\mathrm{sup}(x), \mathrm{sup}(x), \max_{i \in x} (\mathrm{sup}(\{i\}))\right). \tag{19}$$

This follows because

(1) $\mathrm{sup}(x')$ cannot have support greater than $\mathrm{sup}(x)$, and for any values of the other arguments, M is maximized when the support of the first argument is maximized;

(2) M is maximized when the supports of both y and z are minimized, neither of which can have support lower than sup(x'), which the first clause sets to sup(x); and
(3) For every $i \in x$, there must be a partition of x' into $x' \setminus i$ and $\{i\}$, and hence there must be a partition of x' whose support is no lower than $\max_{i \in x}(\text{sup}(\{i\}))$. As M is maximized when the third argument is minimized, this value sets a strong upper bound on $\mathcal{M}_{\text{M}}$.

This bound is undefined for lift (γ) when $\text{sup}(x) = 0$. In this context, we use the trivially derived upper bound of 0.

5. THE OPUS MINER ALGORITHM

There is a strong case for the desirability of finding self-sufficient itemsets, but at first consideration, search for them appears computationally intractable. The search space of potential itemsets is $2^{|\mathcal{I}|}$. For each itemset x considered, we must ensure that all $2^{|x|-1} - 1$ partitions of i pass a statistical test. To determine the value of any itemset x, we must find the maximum of a function with respect to all $2^{|x|-1} - 1$ partitions of i. It is clear that efficient search of this space will require very efficient pruning mechanisms.

OPUS Miner is a new branch-and-bound algorithm for efficient discovery of self-sufficient itemsets. For a user-specified k and interest measure, OPUS Miner finds the top-k productive nonredundant itemsets with respect to the specified measure. It is then straightforward to filter out those that are not independently productive with respect to that set, resulting in a set of self-sufficient itemsets. It can be applied to any *well-behaving* measure of interest [Piatetsky-Shapiro 1991].

OPUS Miner is based on the OPUS search algorithm [Webb 1995]. OPUS is a set enumeration algorithm [Rymon 1992] distinguished by a computationally efficient pruning mechanism that ensures that whenever an item is pruned, it is removed from the entire search space below the parent node.

OPUS Miner systematically traverses viable regions of the search space, maintaining a collection of the top-k productive nonredundant itemsets in the search space explored. When all of the viable regions have been explored, the top-k productive nonredundant itemsets in the search space explored must be the top-k for the entire search space.

OPUS can use any of depth, best, or breadth-first search. OPUS Miner uses depth-first search. This has the advantage that the number of open nodes at any one time is minimized. This allows extensive data (most importantly the relevant TIDset) to be maintained for every open node and ensures that only one such record will be stored at any given time for each level of depth that is explored. It also promotes locality of memory access, as most new nodes that are opened are minor variants of the most recently explored node. It has the disadvantage that larger numbers of deep nodes are explored than would be necessary if breadth-first search were employed. This presents a significant computational burden due to the complexity of node evaluation increasing exponentially as itemset size increases.

There is a choice between recalculating the TIDset that an itemset covers every time its support is required, or memoizing the support. This involves a trade-off between space and computation. OPUS Miner memoizes these support values. An itemset's support is not memoized if it is determined that no superset can be a top-k productive nonredundant itemset, either because the supersets must be redundant, cannot achieve higher values of interest measure $\mathcal{M}$ than the current k best value, or cannot pass the Fisher Exact Test. As ExpandItemset ensures that all subsets of an itemset must be traversed before the node for the itemset is opened, Apriori [Agrawal et al. 1993]-like pruning of supersets of unviable itemsets can be achieved by checking whether any immediate subset of a candidate itemset has not been memoized.

Algorithm 1 presents the top level of the algorithm, which simply establishes a queue of items ordered in descending order on the upper bound on the value of any itemset that can include the item and then calls the ExpandItemset search procedure (Algorithm 2) once for each item. This sets *topK* where it appears (see algorithm as well) to the top-*k* productive nonredundant itemsets. These are then scanned by checkIndepProductive to identify any itemsets that are not independently productive.

ALGORITHM 1: OPUS Miner

The top level of the OPUS Miner algorithm.

```
procedure OPUS MINER(dataset $D$, integer $k$, measure $\mathcal{M}(\bullet, \bullet, \bullet)$)
    $q \leftarrow \{i \mid \text{fisher}(\#(i), \#(i), \#(i)) \leq \alpha_2\}$
    initialize $q'$ to be an empty queue of items
    $topK \leftarrow \emptyset$
    for all $i \in q$ in descending order on $\mathcal{M}(\text{sup}(i), \text{sup}(i), \text{sup}(i))$ do
        if $\mathcal{M}(\text{sup}(i), \text{sup}(i), \text{sup}(i)) > \min_{k \in topK}(k.value)$ then
            ExpandItemset($\{i\}$, $q'$)
            insert $i$ into $q'$
        end if
    end for
    checkIndepProductive()
    return $topK$
end procedure
```

ALGORITHM 2: ExpandItemset

Explore all supersets of itemset X formed by adding items in item queue q.

```
procedure ExpandItemset(itemset $X$, queue $q$)
    initialize $q'$ to be an empty queue of items
    for all $i \in q$ do
        $X' \leftarrow X \cup \{i\}$
        $p' \leftarrow \text{fisher}(\#(X'),\ \max_{j \in X'}(\#(j)),\ \#(X'))$
        if $\mathcal{M}\left(\text{sup}(X'),\ \text{sup}(X'),\ \max_{j \in X'}(\text{sup}(\{j\}))\right) > \min_{k \in topK}(k.value)$ and $p' \leq \alpha_{|X'|}$ then
            checkImmediateSubsets($X'$, $block$, $apriori$)
            if $\neg apriori$ then
                checkPartitions($X'$, $val$, $p$)
                if $val > \min_{k \in topK}(k.value)$ and $p < \alpha_{|X'|}$ then
                    insertItemset($X'$, $\#(X')$, $val$, $p$)
                end if
                if $\neg block$ and $p' \leq \alpha_{|X'|+1}$ then
                    $memoize(X', \#(X'))$
                    ExpandItemset($X', q'$)
                    insert $i$ into $q'$
                end if
            end if
        end if
    end for
end procedure
```

The heart of the algorithm is the ExpandItemset search procedure. It takes as arguments an itemset to be expanded (often called the *head*), the set of TIDs that the itemset covers, and a queue of items available to expand the current itemset (often called the *tail*).

For each item in the queue, a new itemset, X', is formed. Line 5 calculates a lower bound on any application of the Fisher Exact Test to a superset of X'. We use a bound established by Hämäläinen [2010]. No superset of X' may cover more TIDs than X'. For every superset of X', we will need to perform a test against every partitioning, including $\{i\}, X' \setminus i$, where i is the highest support item in X'. For such a partitioning, the p-value can never be lower than that obtained if the other partition covers only the TIDs covered by the full itemset and if the full itemset covers as many TIDs as possible (which can be no more than the cover of the current itemset). Thus, we can obtain a bound on the value for the Fisher test by applying it with the equivalent arguments to those used to bound $\mathcal{M}$.

Itemset X' and all of its supersets are abandoned if either this lower bound on p is greater than $\alpha_{|X'|}$, or if k itemsets have already been found and an upper bound on the value of supersets of X' is not greater than the smallest value of one of those k itemsets (line 6). Note that withholding the current item from the queue of items to be passed to subsequent calls to ExpandItemset results in the additional OPUS pruning relative to standard set enumeration search.

Next, a check is performed of whether counts have been memoized for all immediate subsets of X' (Algorithm 4). If not, then it must have been determined that no superset of that subset could be in the solution, and so X' need not be explored. While doing this check, it is also possible to check whether any immediate subset of X' has the same count as X', in which case all supersets of X' must be redundant and should be blocked.

Next, checkPartitions (Algorithm 3) is called. This procedure iterates through all partitionings, x, y, of X', by iterating x through all subsets of X' that exclude an arbitrary fixed item, i, hence iterating y through each set of items that includes i. For each partitioning, it assesses the Fisher Exact Test p-value and the value of X'. The maximum p-value and minimum value are recorded. The search terminates early if the minimum value falls below the k best so far or the maximum p exceeds the current critical value.

ALGORITHM 3: checkPartitions

Check the binary partitions of itemset X. Return the minimum value v of $\mathcal{M}()$ and the maximum p-value p with respect to any partition, or a value of v or v that will result in X being abandoned.

```
procedure checkPartitions(itemset $X$, output value $v$, output p-value $p$)
    $p \leftarrow 0.0$
    $v' \leftarrow \infty$
    $i \leftarrow$ an arbitrary item in $X$
    for all $x \subsetneq X \setminus i$ do
        $y \leftarrow X \setminus x$  $p' \leftarrow \text{fisher}(\#(X), \#(x), \#(y))$
        if $p' > p$ then
            $p \leftarrow p'$
            if $p > \alpha_{|X|}$ then
                return
            end if
        end if
        $v' \leftarrow \mathcal{M}(\text{sup}(X), \text{sup}(x), \text{sup}(y))$
        if $v' < v$ then
            $v \leftarrow v'$
            if $v \leq \min_{k \in topK}(k.value)$ then
                return
            end if
        end if
    end for
end procedure
```

ALGORITHM 4: checkImmediateSubsets

Check all subsets of X formed by removing a single item to determine whether X fails the apriori test ($apriori$ set to **true***) and whether supersets of X must fail either the apriori or redundancy tests ($block$ set to* **true***).*

```
procedure checkImmediateSubsets(itemset $X$, output bool $block$, output bool $apriori$)
    $block \leftarrow$ false
    $apriori \leftarrow$ false
    for all $i \in X$ do
        if $\#(X \setminus i)$ has not been memoized then
            $block \leftarrow$ true
            $apriori \leftarrow$ true
            return
        end if
        if $\#(X \setminus i) = \#(X)$ then
            $block \leftarrow$ true
        end if
    end for
end procedure
```

ALGORITHM 5: checkIndepProductive

Post-process the top k itemsets to determine whether they are independently-productive.

```
procedure checkIndepProductive
    $a \leftarrow \emptyset$
    for all $X \in topK$ in ascending order on $|X|$ do
        for all $s \in topK$ do
            if $X \subsetneq s$ then
                $a \leftarrow a \cup \mathrm{cov}(s)$
            end if
        end for
        $i \leftarrow$ an arbitrary item in $X$
        for all $x \subsetneq X \setminus i$ do
            $y \leftarrow X \setminus x$
            $p \leftarrow \mathrm{fisher}(|\mathrm{cov}(X) \setminus a|, |\mathrm{cov}(x) \setminus a|, |\mathrm{cov}(y) \setminus a|)$
            if $p > \alpha_{|X|}$ then
                remove $X$ from $topK$
                break from for loop
            end if
        end for
    end for
end procedure
```

Itemset X' is added to the top-k so far if and only if its value is higher than the k highest so far and its p-value is no higher than $\alpha_{|X'|}$. Only if the upper bound on the p-value is no greater than the minimum critical value for itemsets larger than X' and it is not determined that all supersets must be redundant is a recursive call made to ExpandItemset and the item added to the queue for use at lower levels.

The process of detecting itemsets in *topK* that are not independently productive is shown in Algorithm 5. The itemsets in *topK* are processed in ascending order on size so as to ensure that only self-sufficient itemsets are used to constrain the self-sufficiency of other itemsets.

The function for the Fisher Exact Test is shown in Algorithm 6. For efficiency, factorial values are memoized so that each need be calculated once only.

6. COMPLEXITY

The worst-case time complexity of OPUS Miner with respect to the number of items is $\Omega(2^{|\mathcal{I}|})$, as in the worst case, it will not be able to do any effective pruning and will have to explore all itemsets.

ALGORITHM 6: fisher

Return the result of a Fisher Exact Test with respect to the count of an itemset sup and the counts of two subsets that partition it, $sup1$ and $sup2$.

```
function fisher(int $sup$, int $sup1$ int $sup2$)
    $p \leftarrow 0.0$
    $a \leftarrow n - sup1 - sup2 + sup$
    $b \leftarrow sup1 - sup$
    $c \leftarrow sup2 - sup$
    $d \leftarrow sup$
    while $b \geq 0$ and $c \geq 0$ do
        $p \leftarrow \frac{(a+b)!(c+d)!(a+c)!(b+d)!}{a!b!c!d!(a+b+c+d)!}$
        $a \leftarrow a + 1$
        $b \leftarrow b - 1$
        $c \leftarrow c - 1$
        $d \leftarrow d + 1$
    end while
    return $p$
end function
```

If the space of itemsets that need to be explored does not change, then the only operations that are affected by an increase in the number of examples are the evaluations of itemset support and of the Fisher Exact Test. Both of these scale linearly, and hence complexity with respect to the number of examples is O(n). The amount of the search space that is explored is affected primarily by the pruning mechanisms. The pruning for redundancy and on optimistic value is not affected by the data quantity. However, as data quantity increases, the Fisher test is able to pass itemsets with ever smaller support, as the count will increase as the data quantity increases. As a result, the amount of the search space explored will increase as data quantity increases due to the Fisher test pruning fewer alternatives.

In practice, the time requirements depend critically on the efficiency of the pruning mechanisms for a particular set of data, and vary greatly from dataset to dataset, as demonstrated by the experiments that follow.

Due to its use of depth-first search, the number of nodes in the search space that are open simultaneously is small, and the space requirement for the data they must store is dominated by the space requirements of the memoized itemset counts. This is also in the worst case $\Omega(2^{|\mathcal{I}|})$ if no pruning occurs. In practice, the space requirements also depend on the efficiency of the pruning mechanisms for a given dataset.

The effectiveness of the pruning mechanisms and the concomitant computational requirements of the OPUS Miner algorithm in practice are investigated in Section 8.5.

7. RELATED RESEARCH

Association discovery has been intensely researched for more than two decades [Han et al. 2007], and it is possible here to cover only a small amount of closely related research.

7.1. Alternative Approaches to Finding Succinct Summaries

The bodies of research into information theoretic techniques for finding succinct summaries of violations of the independence assumption in data [Siebes et al. 2006; Gallo et al. 2007; Mampaey et al. 2011; Vreeken et al. 2011] and randomization [Hanhijärvi et al. 2009] and statistical [Lijffijt et al. 2012] testing to incrementally build such a summary address much the same issues as self-sufficient itemsets. As these techniques aim to model the full joint distribution, they will tend to develop more succinct collections of itemsets than self-sufficient itemsets but will necessarily choose between one of many potential such collections without informing the user of the alternatives. In contrast, self-sufficient itemsets seek to identify all key associations in data that are likely to be of interest.

In Section 8, we investigate the relative performance of OPUS Miner and two exemplar information theoretic systems, which we describe in more detail here.

Observing that the overly large numbers of results of frequent pattern mining algorithms are mostly due to massive redundancy, Vreeken et al. [2011] proposed—instead of asking for all patterns that individually meet some conditions—to mine a small *set* of patterns that together optimize a global criterion. As the target criterion, Siebes et al. [2006] defined the best set of itemsets by the MDL principle [Rissanen 1978] as the set that provides the best lossless compression of the data. The heuristic Krimp algorithm typically discovers hundreds and up to a few thousand patterns. Its results have been shown to be highly useful in a wide range of data mining tasks.

Instead of using an ad hoc encoding scheme, Mampaey et al. [2012] proposed to model the full joint distribution of the data by means of a maximum entropy model [Jaynes 1982]. Essentially, its goal is to find the set of itemsets that best predicts the data. To this end, the mtv algorithm iteratively finds the itemset for which the frequency in the data most strongly deviates from the prediction given the current model, then adds this itemset to the model, updates its predictions, and so forth, until MDL tells it to stop. Experiments show that it finds highly informative patterns. Querying this maximum entropy model is NP-hard, and exponential in the number of included itemsets, however, and mtv can hence only be used to mine relatively small top-k summaries.

Maximum entropy approaches [Wang and Parthasarathy 2006; Tatti 2008; De Bie 2011] provide a powerful approach to filtering itemsets relative to their subsets. However, they do not have a counterpart to the independent-productivity constraint and do not facilitate the same type of simple explanations that self-sufficiency can provide for why any specific itemset is rejected. Moreover, in general, inferring maximum entropy distributions is NP-hard, whereas querying such models is even PP-hard [Tatti 2008].

7.2. Nonderivable Itemsets

As discussed in more detail elsewhere [Webb 2010], nonderivable itemsets [Calders and Goethals 2007], which provide a succinct summary of the full multinomial distribution in the data, are very different in nature to self-sufficient itemsets. For example, pairs of items that are statistically independent of one another may be nonderivable but will never be self-sufficient. The practical impact of this is illustrated in the experiments in which we mine associations in lottery results, presented later in Section 8.2.

A nonderivability constraint is similar in nature to the redundancy constraint in self-sufficient itemsets. Both are constraints that reject an itemset $\mathcal{I}$ if its support can be determined on the basis of knowledge about $\mathcal{I}$'s subsets. A nonderivability constraint is more powerful than a redundancy constraint. It will reject all itemsets rejected by an redundancy constraint and more. Based on personal experience, the type of inference that underlies the redundancy constraint is readily applied by nonexpert association-discovery users, and failure to apply such a constraint often leads to user frustration. It is not clear to what extent this is true of the inferences that underlie a nonderivability constraint. An interesting topic for future research would be to explore replacing the redundancy constraint in self-sufficient itemset discovery by a nonderivability constraint, or perhaps some intermediate constraint.

7.3. Other Related Approaches

Hämäläinen [2010] has provided a deep analysis of the efficient use of a Fisher Exact Test to find association rules where the antecedent and consequent are statistically associated with one another. The current work generalizes those techniques to the context of itemsets.

A number of OPUS and related branch-and-bound algorithms have been developed for rule discovery [Webb 1995; Bayardo et al. 2000; Webb 2000; Webb and Zhang 2005; Hämäläinen 2012]. These differ quite substantially from OPUS Miner, most significantly because none of them supports measures of interest based on the minimal degree of deviation of the observed frequency of an itemset from that expected under independence between any binary partition, and none supports statistical testing of a null hypothesis that the subsets in at least one partition of the itemset are not positively correlated. So, for example, if there is a strong association between pregnant and edema and the strength of that association is strengthened only infinitesimally by adding factor X, then all of these previous approaches will assess {pregnant, X} → edema as more interesting than {pregnant} → edema and {edema, X} → pregnant as more interesting than {edema} → pregnant. In contrast, OPUS Miner will only assess the degree of interest of {edema, X, pregnant} with respect to the amount of the *increase* in the strength of association that arises from adding X and requires further that the increase be statistically significant.

Another point of difference is that OPUS Miner finds itemsets rather than rules, thereby avoiding the extraordinary amount of duplication in representations of each single core underlying association that is often evidenced in rule discovery.

Our approaches consider only positive associations. For many applications, it will also be useful to find negative associations [Wu et al. 2004]. The six principles for identifying associations that are unlikely to be of interest outlined in the Introduction apply equally to positive and negative associations but have been applied here only to positive associations. For attribute-value data with binary attributes, negative associations are found by the current framework, as a negative association involving one value of a binary association is equivalent to a positive association involving the other value. For other cases, negative associations can be found by augmenting the data with items corresponding to the negations of all nonbinary values. It is likely, however, that it is possible to develop much more computationally efficient approaches than this, and these remain an interesting direction for future research.

Previous research [Fu et al. 2000] has called the k itemsets with highest support the k “most interesting itemsets.” We argue that high-support itemsets are often not interesting, merely representing pairs of items that are each individually frequent and hence co-occur frequently despite having low, if any, association.

Our approaches are designed for application in the absence of information about a user’s background knowledge and beliefs. Approaches that take account of background knowledge provide an important and closely related field of research [Jaroszewicz et al. 2009; Tatti and Mampaey 2010; De Bie 2011].

8. EXPERIMENTS

Here we empirically evaluate OPUS Miner and investigate the quality and usefulness of self-sufficient itemsets. We start our evaluation by analyzing the associations discovered from synthetic data with known ground truth. We then use two case studies to qualitatively evaluate our results and compare them to five alternative itemset mining techniques. Finally, we investigate the algorithm’s computational efficiency.

8.1. Synthetic Data

First, we consider the performance of OPUS Miner on synthetic data, as we can then compare the results to the ground truth. We evaluate on three datasets, respectively generated by the independence model, an itemset-based model, and a Bayes net.

Table I. Results on Synthetic Transaction Data

		Results Breakdown				
# Rows	# Results	# Exact	# Intersects	# Subsets	# Union Subsets	# Unrelated
1,000	117	4	1	112	—	—
10,000	186	7	0	109	70	—
100,000	750	9	1	62	678	—
1,000,000	972	11	0	27	934	—

8.1.1. Independence Model. In this first experiment, we consider datasets of 1,000 up to 1,000,000 rows over 100 items, for which the frequencies were drawn uniform randomly between [0.05, 0.25].

For each of these datasets, OPUS Miner correctly finds no itemset to be productive.

8.1.2. Itemsets. Next, we generate synthetic data in which we plant itemsets.

First, we generate $\mathcal{G}$, a set of 15 itemsets. For each $g \in \mathcal{G}$, we choose the cardinality from [3, 7], support from [0.01, 0.20], and the items from [1, 50]—all uniformly at random. We generate the data over an alphabet of 100 items, of respectively, 1,000, 10,000, 100,000, and 1,000,000 transactions, and plant itemsets at random with probability proportional to their support. As a final step, we add noise by flipping the values of a random 2% of items. This may alter the empirical supports of the embedded itemsets. All items in [51, 100] are statistically independent of all other items.

We run OPUS Miner on each of these datasets. For all but the largest, we could find *all* self-sufficient itemsets by setting k to a large number. Due to the large memory requirements of memoizing itemsets during the search, this was not possible for 1,000,000 transactions, and for this dataset we present results with respect to the top-10,000-leverage (δ) nonredundant productive itemsets, of which only 972 are self-sufficient.

We analyze the results using a protocol based on that of Zimmermann [2013]. We use the following properties. An itemset x is an *exact match* if and only if

$$x \in \mathcal{G}. \tag{20}$$

An itemset x is an *intersection* if and only if

$$x \notin \mathcal{G} \wedge \exists y \in \mathcal{G} \exists z \in \mathcal{G}, y \neq z \wedge x \subsetneq y \wedge x \subsetneq z. \tag{21}$$

An itemset x is a *subset* if and only if

$$x \notin \mathcal{G} \wedge \exists y \in \mathcal{G}, x \subsetneq y \wedge \neg \exists z \in \mathcal{G}, y \neq z \wedge x \subsetneq z. \tag{22}$$

An itemset x is a *union subset* if and only if

$$x \notin \mathcal{G} \wedge [\neg \exists y \in \mathcal{G}, x \subsetneq y] \wedge x \subseteq \bigcup_{\substack{z \in G: \\ |z \cap x| > 1}} z. \tag{23}$$

In Equation (23), we include the condition $|z \cap x| > 1$, requiring an overlap of at least two items between x and each z, as we want to consider the union of itemsets in $\mathcal{G}$ for which x describes at least part of the association.

An itemset x is *unrelated* to $\mathcal{G}$ if and only if it is none of an exact match, an intersection, a subset, or a union subset.

We give counts of the number of itemsets that fall into each of these mutually exclusive classes in Table I. Only modest numbers of self-sufficient itemsets are found, and importantly, none are spurious: all can be related directly to the itemsets planted in the data. Further, as the quantity of data increases, so too does the number of planted itemsets that are recovered exactly—and, importantly, found at the top of the ranking. The planted itemsets that are not recovered exactly are all of such high cardinality that a lot of data is required to pass the very strict test on productivity with its strong correction for multiple testing. For each of these itemsets, we do find direct subsets.

Many of the planted itemsets overlap. In this case, for every partition of the union of the planted itemsets, the observed frequency will exceed the frequency predicted under an assumption of independence. In consequence, the union will be assessed as productive. Where the union is small enough to be found, it will render its subsets not independently productive. Where it is too large, instead many of its subsets will be found.

8.1.3. Bayes Nets. As third and final evaluation on synthetic data, we consider data generated from a Bayes net. This is a more difficult setup than shown previously, as a Bayes net can encode negative correlations: item B may have higher probability for being generated when item A is absent. Self-sufficient itemsets model positive correlations only, and hence it is interesting to see how informative these are with regard to the generating model.

We first generate a Bayes net over 100 items, drawing 250 edges at random, while keeping the maximum number of parents at 5. For each item, per conditioning, we draw the probability uniformly at random from [0.05, 0.5]. We then use the Bayes net to generate datasets of respectively 1,000, 10,000, 100,000, and 1,000,000 rows. We apply OPUS Miner to mine up to the top-100 nonredundant and productive itemsets—for the smallest dataset only 30 itemsets are nonredundant and independently productive.

Our main analysis is to see whether the discovered itemsets correctly identify part of the Bayes net structure. As the results are highly similar between the four datasets, we only consider those for the 1,000,000 row database. Out of the top-100 nonredundant and productive itemsets, 91 are also independently productive.

First, we observe that each of these self-sufficient itemsets match a single connected component of the Bayes net. That is, no discovered itemset was generated by independent components. More in depth, we find that all itemsets indeed identify local positive associations. Most of these are in the form of chains. That is, for an itemset $\{A, B, C\}$, we see the Bayes net to be connected as $A \rightarrow B \rightarrow C$, where each of the dependencies are positive correlations. For 21 of the 91 itemsets, there are no intermediate influences factors—that is, items B and C have no other incoming edges.

For 69 discovered self-sufficient itemsets, the intermediate nodes do have additional incoming edges. Investigating the corresponding probability tables reveals that these itemsets identify the strongest positive correlations within the local part of the Bayes net. For example, say that item B is additionally positively correlated with item D. When we find that itemset $\{A, B, C\}$ is self-sufficient, we see that the association $A \rightarrow B$ is much stronger than either $D \rightarrow B$ or $A \wedge D \rightarrow B$.

Finally, we find one itemset for which we find intermediate negative correlations. That is, for a reported self-sufficient itemset $\{A, B, C\}$, we may find that A correlates negatively with parents of B (and/or C). That is, $A \rightarrow (\neg X, \neg Y)$, with $\neg X \rightarrow B$ and $\neg Y \rightarrow C$. Essentially, self-sufficient itemsets are able to correctly identify positive correlations arising from a chain of negative correlations.

8.1.4. Summary. Overall, the evaluation on synthetic data shows that no spurious patterns are returned. The results on the itemset data show that the top self-sufficient itemsets all represent itemsets embedded in the data. The remaining itemsets represent partial recovery of more subtle patterns. By the nature of the strong statistical tests that we employ, to detect (true) associations between many items requires in the order of tens of thousands or more rows. The experiments on the Bayes net generated data show that self-sufficient itemsets are very well suited for identifying positive associations, even if these are generated through a chain of negative associations.

8.2. Lottery Case Study

Next, we assess the practical usefulness of OPUS Miner through two case studies. In this first case study, we compare the results of OPUS Miner to those of three standard itemset mining techniques: frequent itemsets [Agrawal et al. 1993], closed itemsets [Pasquier et al. 1999b], and nonderivable itemsets [Calders and Goethals 2007], as well as to two modern *pattern set* mining techniques, Krimp [Vreeken et al. 2011] and mtv [Mampaey et al. 2012]. Although many pattern set mining methods have been proposed, these two in particular have been shown to provide small and nonredundant sets of high-quality itemsets [Mampaey et al. 2012; Kontonasios and De Bie 2010; Tatti and Vreeken 2012].

In this first case study, we consider—mostly as a sanity check—mining the results of the 2,482 draws of the Belgian National Lottery between 1983 and late 2011.[1] Each record consists of the ids of seven draws without replacement from 42 balls. Assuming that the lottery is fair, no true patterns will exist. When we mine this data for frequent patterns, however, very many patterns are returned. That is, when we mine frequent itemsets [Agrawal et al. 1993], closed itemsets [Pasquier et al. 1999b], or nonderivable itemsets [Calders and Goethals 2007] with minimal support of 1%, we discover 902 itemsets, whereas for a minimum count of 2, respectively 33,382, 29,828, and 33,338 'patterns' are found. This aptly illustrates the problem of using frequent pattern mining for finding associations.

Krimp, on the other hand, for a minimum count of 1, only selects 232 patterns. Each of these occur relatively frequently in the data, respectfully around 80 and 15 times for the selected itemsets of 2 and 3 items, and as such, these itemsets may be considered interesting. However, after statistical correction for the number of patterns considered, none of these frequencies deviate significantly from the independence model, and hence by our definition, these patterns do not identify interesting associations. That Krimp returns these patterns indicates that its prior is too weak—its encoding scheme does not exploit all available information optimally.

In contrast, assuming that the lottery is fair, both mtv and OPUS Miner discover the correct result. OPUS Miner finds no itemsets to be self-sufficient. Even with a minimum count of 1, mtv finds no frequent patterns to be significant.

8.3. NIPS Case Study

Assessing the relative usefulness of alternative collections of associations is difficult, as this can only be assessed by experts in a field, and can only really be assessed relative to an application for which the associations are to be employed. To circumvent these problems, here we use one of the few datasets for which many of this article's readers will be relative experts and hence able to assess the meaningfulness of the resulting associations. The docword.nips dataset comprises 1,500 records, each containing the set of stemmed words found in an NIPS conference paper out of a total vocabulary of 12,375 distinct stemmed words.

8.3.1. Self-Sufficient Itemsets. In this first part of the case study, we systematically investigate each of the aspects of self-sufficiency and the value of itemsets relative to rules.

By setting k to a large value (we used 100,000), we are able to discover all self-sufficient itemsets for this dataset. Doing so takes 7 CPU hours and 15 minutes on a single processor on a Sun Fire X4600 server.

There are 43,210 nonredundant productive itemsets. Of these, 25,618 are independently productive relative to the rest and hence are self-sufficient. Table II presents

[1] www.nationaleloterij.be.

Table II. Top 25 Self-Sufficient Itemsets with Respect to Leverage (δ)

kaufmann, morgan	The publisher Morgan Kaufmann.
cambridge, mit	MIT and its address.
san, morgan, mateo	Morgan Kaufmann's address is San Mateo. However, "Kaufmann" is frequently misspelled and as a result {san, morgan, mateo} is independently productive relative to {san, kaufmann, morgan, mateo}, which appears at rank 10.
mit, press	The publisher MIT Press.
grant, supported	Part of a frequent acknowledgement.
springer, verlag	The publisher Springer-Verlag.
bottom, top	Two related words.
conference, international	A frequent word pair that is independently productive relative to its self- sufficient supersets {conference, international, proceeding} (rank 37) and {artificial, conference, international} (rank 1,241).
conference, proceeding	Two related words.
san,kaufmann,morgan,mateo	The publisher and its address.
node, nodes	Two related words.
algo, rithm	The prefix and suffix of a frequent hyphenation.
negative, positive	Two related words.
pages, editor	Two related words. With a count of 231, this pair is independently productive relative to its self-sufficient supersets {pages, editor, advances} (count 150, rank 836) and {pages, volume, editor} (count 103, rank 1,402).
feature, features	Two related words.
class, classes	Two related words.
artificial, intelligence	Two related words.
thesis, phd	Two related words.
estimate, estimated	Two related words.
san, mateo, advances	The address of the publisher Morgan Kaufmann and a name that appears in the title of several of its publications. Interestingly, {kaufmann, morgan, advances} is not independently productive relative to a number of its supersets that relate to individual books such as {san, kaufmann, morgan, touretzky, advances} (rank 159).
probability, probabilities	Two related words.
role, play	Two related words.
cambridge, mit, press	The publisher MIT Press and its address.
high, low	Two related words.
ieee, tran	These words appear in the titles of many conference proceedings.

Table III. Top 25 Itemsets with Repect to Leverage (δ) with Those That Are Not Self-Sufficient Italicized

kaufmann,morgan	conference,international	morgan,advances
technical,report	hidden,trained	top,bottom
mit,cambridge	learn,learned	*kaufmann,mateo*
mateo,morgan	hidden,training	distribution,probability
san,mateo,morgan	trained,training	conference,proceeding
grant,supported	*san,mateo*	*kaufmann,mateo,morgan*
springer,verlag	descent,gradient	san,kaufmann,mateo,morgan
san,morgan	image,images	
san,kaufmann,mateo	mit,press	

the top 25 self-sufficient itemsets with respect to leverage, together with commentary on why they are selected and their meaning.

We next illustrate the difference between simply finding itemsets with the highest leverage and finding self-sufficient itemsets with the highest leverage. Table III presents the top 25 itemsets with respect to leverage. Those that are not self-sufficient

Table IV. Top 25 Rules with Repect to Leverage (δ)

kaufmann → morgan
morgan → kaufmann
abstract, morgan → kaufmann
abstract, kaufmann → morgan
references, morgan → kaufmann
references, kaufmann → morgan
abstract, references, morgan → kaufmann
abstract, references, kaufmann → morgan
system, morgan → kaufmann
system, kaufmann → morgan
neural, kaufmann → morgan
neural, morgan → kaufmann
abstract, system, kaufmann → morgan
abstract, system, morgan → kaufmann
abstract, neural, kaufmann → morgan
abstract, neural, morgan → kaufmann
result, kaufmann → morgan
result, morgan → kaufmann
references, system, morgan → kaufmann
neural, references, kaufmann → morgan
neural, references, morgan → kaufmann
abstract, references, system, morgan → kaufmann
abstract, references, system, kaufmann → morgan
abstract, result, kaufmann → morgan
abstract, neural, references, kaufmann → morgan

are set in italics. As can be seen, these are all subsets of {san, kaufmann, mateo, morgan}, relating to the publisher Morgan Kaufmann and their address, San Mateo. Both the full itemset {san, kaufmann, mateo, morgan} and its two subsets {kaufmann, morgan} and {san, morgan, mateo} are self-sufficient, the former subset because the name sometimes appears without the address and the latter because *Kaufmann* is frequently misspelled. This is a positive result for self-sufficient itemsets, which in this case cleanly identify appropriate itemsets as being unlikely to be interesting.

We next contrast finding itemsets to finding rules. Table IV presents the 25 rules with highest leverage. The top two rules provide alternative representations of the top itemset, {kaufmann, morgan}. The remaining rules are all formed by adding high-support words to the antecedent of one or the other of these rules.

We believe that the contrast between Tables II and IV illustrates nicely some of the advantages of itemsets as a representation for associations, relative to rules. It also illustrates the importance of assessing the value of associations based on all partitions into two subsets, as advocated by Principles 1 and 6. For example, consider abstract, morgan → kaufmann. The word *abstract* appears in 97% of all documents. It is almost independent of either *morgan* or *kaufmann*, as indicated by the leverage of kaufmann, morgan → abstract being just 0.0065. It has support of 0.281 (count 421), confidence of 0.814, and leverage of 0.1807. The rule morgan → kaufmann has support of 0.283 (count 424), confidence of 0.812, and leverage of 0.1817. The increase in confidence is a result of there being one less document containing abstract among those that contain both *morgan* and *kaufmann* than one would expect if its presence had no effect on the association between the two. This does not seem like compelling evidence for considering it to be the third most interesting association in the data. It is hard to see why anyone would consider it a more interesting association than the second of the

Table V. Top 25 Self-Sufficient Itemsets with Respect to Lift (γ)

duane,leapfrog
crippa,ghiselli
artery,coronary
brasher,krug
zag,zig
petek,schmidbauer
harmonet,harmonization
ornstein,uhlenbeck
taube,postsubiculum
americana,periplaneta
chorale,harmonization
kerszberg,linster
mizumori,postsubiculum
ekman,hager
chorale,harmonet
fodor,pylyshyn
nakashima,satoshi
iceg,implantable
alessandro,sperduti
iiiiiiii,iiiiiiiiii
nuno,vasconcelos
implantable,pickard
lpnn,petek
deerwester,dumais
jeremy,bonet

Table VI. Top 25 Itemsets with Respect to Lift (γ)

debris,rectal
multiset,zly
mm32k,mm_vector
inducer,sdti
labeller,volcano
inducer,tdsi
sdti,tdsi
contextualized,flee
gopal,sdh
eptesicus,ferragamo
apobayesian,sasb
eptesicus,glint
mtb,stb
apobayesian,sbsb
muesli,nlist
e13b,i1000
ferragamo,glint
semenov,unlearning
aisb,strawman
sasb,sbsb
pm1,zly
g4f3,neurochess
flee,forbus
contextualized,forbus
multiset,pm1

self-sufficient itemsets, {technical, report}. By ignoring all but one of the partitions of an itemset when assessing interestingness, rule-based techniques inflate the apparent value of many minor variants of a single core association.

The manner in which conventional association rules can be dominated by trivial variants of a small number of key associations is further illustrated by the fact that the highest leverage (0.1468) rule that does not contain either *morgan* or *kaufmann* is abstract, input, training → trained. This does not appear in the top 25 because 712 rules each containing *morgan* and/or *kaufmann* have higher leverage. This rule, while not a trivial variant of the association between *morgan* and *kaufmann*, is instead a trivial variant of the association between *training* and *trained*, as *abstract* and *input* are both very frequent items, neither of which is strongly associated with either *training* or *trained*.

We next consider how applying self-sufficient itemsets with different preference functions allows them to be used for different analytic objectives. Table V lists the 25 top self-sufficient itemsets on lift (γ). These turn out to be all pairs of words that each appear in only four to six papers and both appear in exactly or almost exactly the same papers. The requirement that the words appear in at least four papers arises from this being the minimum number of examples required for an itemset to become statistically significant in the context of OPUS Miner's very strict correction for multiple comparisons. They provide an interesting contrast to the self-sufficient itemsets in the top 25 itemsets on leverage (Table III), illustrating how different measures of interest can highlight qualitatively different forms of association within data. These associations include authors who publish together (e.g., *petek* and *schmidbauer*), authors and the topics in which they specialize (e.g., *taube* and *postsubiculum*), the two words in a hyphenated surname (*crippa* and *ghiselli*), an author's first name and surname (*jeremy* and *bonet*), and words that relate to one another (e.g., *artery* and *coronary*).

By way of contrast, Table VI lists the 25 top itemsets on lift without a self-sufficiency constraint. These turn out to be all pairs of words that each appear in only one paper and both appear in the same paper. Some of these are clearly meaningful

Table VII. Top 25 Frequent (Closed) Itemsets

abstract,references
references,result
abstract,references,result
abstract,system
abstract,function,references
neural,references
abstract,neural,references
references,set
abstract,function,result
abstract,references,system
abstract,references,set
abstract,neural,result
abstract,number
abstract,result
abstract,function
abstract,neural
function,references
abstract,set
function,result
references,system
neural,result
abstract,introduction
result,system
result,set
abstract,network

Table VIII. Closure of {duane, leapfrog}

abstract, according, algorithm, approach, approximation, bayesian, carlo, case, cases, computation, computer, defined, department, discarded, distribution, duane, dynamic, dynamical, energy, equation, error, estimate, exp, form, found, framework, function, gaussian, general, gradient, hamiltonian, hidden, hybrid, input, integral, iteration, keeping, kinetic, large, leapfrog, learning, letter, level, linear, log, low, mackay, marginal, mean, method, metropolis, model, momentum, monte, neal, network, neural, noise, non, number, obtained, output, parameter, performance, phase, physic, point, posterior, prediction, prior, probability, problem, references, rejection, required, result, run, sample, sampling, science, set, simulating, simulation, small, space, squared, step, system, task, term, test, training, uniformly, unit, university, values, vol, weight, zero

({apobayesian, sbsb}, {mm32k, mm_vector}, {e13b, i1000}, {inducer, sdti}, {semenov, unlearning}, {inducer, tdsi}, {sdti, tdsi}, {sasb, sbsb}, {eptesicus, ferragamo}, {apobayesian, sasb}, {mtb, stb}). However, at least as many appear to be merely chance co-occurrences ({debris, rectal}, {multiset, zly}, {muesli, nlist}, {ferragamo, glint}, {labeller, volcano}, {aisb, strawman}, {contextualized, flee}, {pm1, zly}, {gopal, sdh}, {g4f3, neurochess}, {flee, forbus}, {contextualized, forbus}, {eptesicus, glint}, {multiset, pm1}). While some of the pairs that appear only once do appear to have meaning as associations, there is a clear qualitative difference in the associations that pass the statistical test for productivity and hence are deemed self-sufficient.

Next, we contrast the self-sufficient itemsets approach to classical frequent itemset mining. Table VII presents the 25 most frequent itemsets. All these turn out to be closed. As can be seen, these itemsets are simply collections of words that all appear in most NIPS papers. We believe that this illustrates the widely perceived low relative value of support as a measure of interest [Han et al. 2007].

It might be thought that self-sufficient itemsets are similar to closed itemsets [Pasquier et al. 1999a]. In fact, to the contrary, the redundancy constraint requires that the itemsets be no more than one item larger than a generator. To illustrate the difference, consider the first self-sufficient itemset in Table V, {duane, leapfrog}. This itemset comprises the names of an author and of the algorithm that he developed. These names each appear in only four NIPS papers, those being in both cases the same four papers. The closure of this itemset, listed in Table VIII, is the set of all 99 words that appear in all four papers that contain those two words.

The two longest self-sufficient itemsets are {error, hidden, input, output, trained, training, unit} ($\delta = 0.0397$, $p = 5.85E{-}32$) and {error, hidden, input, layer, output, trained, training} ($\delta = 0.0380$, $p = 2.08E{-}30$), each containing seven items. Their union, {error, hidden, input, layer, output, trained, training, unit} ($\delta = 0.0328$, $p = 9.2E{-}26$) is not accepted as productive due to the very strict adjusted critical value $\alpha_8 = 2.87E{-}32$. Otherwise, it would have rendered both the seven itemsets not independently productive. This illustrates a limitation of the algorithm with respect to its capacity to find very long itemsets.

8.4. Comparison with Minimum Description Length Approaches

The major alternative approach to finding key associations in high-dimensional data is provided by the information theoretic minimum description length (MDL) techniques [Siebes et al. 2006; Gallo et al. 2007; Mampaey et al. 2011; Vreeken et al. 2011]. Here we seek to assess how the key associations identified by OPUS Miner compare to those identified by exemplar MDL approaches KRIMP and MTV.

We ran KRIMP with a minimum support of 20%, and it selected 1,369 itemsets out of the 207,419,059 frequent itemsets. Of these 1,369 itemsets, 41 are self-sufficient, 38 are subsets of self-sufficient itemsets where only one item is missing, and 76 are supersets of self-sufficient itemsets where we find only one item extra.

We show the top 25 itemsets, ordered by the area of the data that they cover in the KRIMP encoding process, in Table IX. KRIMP's top 25 includes seven self-sufficient itemsets such as {computer, department, science, university} and {cambridge, mit, press}. It also includes four nonredundant and productive but not independently productive itemsets. An example is {kaufmann, morgan, san}. OPUS Miner assesses this as not independently productive relative to self-sufficient itemsets {san, kaufmann, morgan, mateo}, {san, kaufmann, morgan, touretzky, mateo}, {san, kaufmann, morgan, touretzky, advances}, {san, processing, kaufmann, morgan, advances}, {san, kaufmann, moody, morgan, advances}, {san, kaufmann, morgan, publisher}, and {san, lippmann, kaufmann, morgan}. As discussed in Section 8.3.1, OPUS Miner's top 25 includes instead {kaufmann, morgan} (the publisher's name), {san, morgan, mateo} (included because the name *Kaufmann* is often misspelled) and {san, kaufmann, morgan, mateo} (the name and address).

In the KRIMP results, we also find a number of very long itemsets (here at ranks 1, 8, 13, 15, and 24) that seem to group frequently occurring words together without necessarily identifying a clear concept. We also see that relatively many itemsets combine a key concept with one or more less strongly related items, such as {pattern, recognition, simple} and {dimensional, mean, order, space}, for which we respectively find that {pattern, recognition} and {dimensional, space} are the key nonredundant productive itemsets.

Another interesting contrast is provided by {com, tion}. These are a common prefix and a common suffix for hyphenated words. It is plausible that these are associated because documents that use hyphenation are relatively likely to have each of these word parts. With $p = 1.57E{-}6$, this itemset fails OPUS Miner's strict significance test. By contrast, OPUS Miner's top 25 includes the directly related prefix-suffix pair {algo, rithm}, $\delta = 0.1042$, the support of which is too low (12%) to be considered by KRIMP.

On a positive note, KRIMP can detect complex concepts in much smaller amounts of data than OPUS Miner's strict statistical testing for productivity requires. For example, {algorithm, approach, error, learning, method, problem} ($\delta = 0.036$) is the type of collection of associated items that OPUS Miner seeks. If this itemset had not been excluded by OPUS Miner's very strong statistical testing, it would probably render many of its subsets not independently productive.

Next, we mine the top 25 itemsets with MTV using a minimum support of 20%. As running time is exponential in the number of selected overlapping itemsets,we impose a max group length of five overlapping sets. We give the results in Table X.

Overall, we see that MTV identifies meaningful concepts, with very little redundancy in between, although the very frequent word "abstract" appears only very weakly related in three of the discovered itemsets and the equally frequent "references" only very weakly related in another. We find 8 out of the 25 to be self-sufficient, and a

Table IX. Top 25 KRIMP Itemsets, with Respect to Covered Area
Itemsets that are not self-sufficient are italicized. Asterisks denote the itemsets that are nonredundant and productive but not independently productive.

abstract, _function_, _input_, _introduction_, _network_, _neural_, _number_, _output_, _references_, _result_, _set_, _system_, _weight_	Includes the 10 most frequent words (_abstract_, _references_, _result_, _function_, _neural_, _system_, _set_, _network_, _introduction_, _number_) together with the very frequent (count 743) productive but not independently productive itemset {`input,output,weight`}. Many subsets of {`input, network, neural, output, weight`} are self-sufficient, but with $p = 6.8E{-}18$, it fails to pass the adjusted $\alpha = 1.3E{-}21$ for itemsets of size 5.
algorithm, _approach_, _error_, _learning_, _method_, _problem_	Despite relatively high leverage (0.0360), OPUS Miner's strict statistical testing excludes it ($p = 8.3E{-}012$, adjusted α for size 6 itemsets $6.3E{-}24$). The closest self-sufficient itemset is {`algorithm, approach, learning, method`}.
advances, _data_, _information_, _model_, _processing_	$\delta = 0.0116$.
hidden, _layer_, _trained_, _training_ *	Although nonredundant and productive, this itemset is not independently productive due to four itemsets with support lower than the minimum at which KRIMP was run, among which are {`hidden, layer, trained, training, propagation`} (14%) and {`hidden, layer, trained, training, train`} (13%).
`computer,department,science,university`	
`cambridge,mit,press`	
high, _low_, _single_	$\delta = 0.0309$. $p = 2.3E{-}8 > \alpha = 4.0E{-}14$ for itemsets of size 3. Subset {`high, low`} is self-sufficient.
abstract, _conclusion_, _function_, _input_, _introduction_, _network_, _neural_, _number_, _output_, _references_, _result_, _set_, _system_	Very similar to the top-ranked itemset, this itemset also mostly consists of very frequent words; in comparison, _weight_ is missing and _conclusion_ is added.
pattern, _recognition_, _simple_	Not productive. The subset {`pattern, recognition`} is productive, but not independently productive, due to, among others, {`object, pattern, recognition`}.
equation, _parameter_, _zero_	$\delta = 0.0254$.
dimensional, _mean_, _order_, _space_	$\delta = 0.0126$.
distribution, _probability_, _statistical_ *	This itemset is not independently productive due to 15 self-sufficient supersets, including {`distribution, estimation, statistical, probability`}.
abstract, _case_, _defined_, _function_, _introduction_, _network_, _neural_, _number_, _references_, _result_, _set_, _system_	As with the itemsets at rank 1 and 8, this itemset is a collection of frequent words.
kaufmann, _morgan_, _san_ *	Not independently productive due to the self-sufficient superset {`morgan, kaufmann, san, mateo`}.
abstract, _function_, _input_, _introduction_, _network_, _neural_, _number_, _order_, _references_, _result_, _set_, _system_	See rank 1, 8, and 13.
`discussion, thank`	
`control, current`	
`experiment, experimental`	
architecture, _net_, _unit_ *	Not independently productive due to {`architecture, net, unit, weight`} and {`architecture, net, input, unit`}.
com, _tion_	$\delta = 0.0280$. Common hyphenation prefix and suffix. By contrast, OPUS Miner's top 25 includes {`algo, rithm`}, $\delta = 0.1042$, with support (12%) below KRIMP'S minimum.
`grant, supported`	
`feature, features`	
factor, _real_	$\delta = 0.0223$.
abstract, _error_, _function_, _input_, _network_, _neural_, _number_, _output_, _references_, _result_, _set_, _system_, _weight_	See rank 1, 8, 13, and 15.
field, _local_	$\delta = 0.0360$. $p = 7.0E{-}9 > \alpha = 3.3E{-}10$ for size 2 itemsets.

Table X. Top 25 Itemsets According to MTV

Itemsets that are not self-sufficient are italicized. Asterisks denote the itemsets that are nonredundant and productive but not independently productive.

abstract, *error*, *hidden*, *input*, *network*, *neural*, *output*, *trained*, *training*, *unit*	$\delta = 0.0060$. OPUS Miner finds several subsets of this itemset, but it fails the strict statistical tests for productivity, as do the next six itemsets.
abstract, *neural*, *system*, *processing*, *kaufmann*, *morgan*, *advances*	$\delta = 0.0040$.
hidden, *input*, *layer*, *learning*, *network*, *neural*, *problem*, *set*, *training*, *weight*	$\delta = 0.0154$.
abstract, *data*, *number*, *performance*, *result*, *set*, *test*, *trained*, *training*	$\delta = 0.0064$.
references, *system*, *mit*, *press*, *processing*, *cambridge*	$\delta = 0.0102$.
distribution, *function*, *parameter*, *probability*, *statistical*	$\delta = 0.0097$.
san, *mateo* *	Not independently productive relative to {`morgan, kaufmann, san, mateo`}.
descent, *function*, *gradient*	$\delta = 0.0137$. Extensions of nonredundant productive itemset {`gradient, descent`} that are preferred by OPUS Miner include {`gradient, descent, error, training`} ($\delta = 0.0393$), {`gradient, descent, minimize`} ($\delta = 0.0385$), and {`gradient, descent, training, weight`} ($\delta = 0.0384$).
technical, report, university	
computer, department, science, university	
algorithm, *approach*, *function*, *parameter*, *point*, *vector*, *method*	$\delta = 0.0112$. OPUS Miner finds subsets of this itemset, such as {`algorithm, approach, vector, method`} (lev. 0.0507). At 0.0270, the leverage of adding *point* to this itemset is too low to pass OPUS Miner's strict statistical tests. The leverage of adding the very frequent word *function* to it is even lower (0.0112).
grant, supported	
learn, *learned*, *information*	{`learn, learned`} is self-sufficient (lev. 0.1094), but addition of the frequent word *information* decreases leverage to 0.0161.
conference, proceeding	
algorithm, optimal, method, approximation	
distribution, *function*, *parameter*, *mean*, *gaussian*	Leverage 0.0112. OPUS Miner finds {`distribution, parameter, mean`} (lev. 0.0471).
positive, negative	
consider, *define*, *defined* *	This itemset is not independently productive relative to {`consider, define, defined, finite`} (count 152) and {`consider, define, defined, definition`} (count 150).
case, *consider*, *general*, *paper*, *note*	$\delta = 0.0214$.
information, *feature*, *features*	$\delta = 0.0123$. OPUS Miner finds {`feature, features`} (lev. 0.0937), and eight of its supersets, such as {`feature, features, image, images`} (lev. 0.0471), but the addition of the frequent (count 1200) word *information* results in low leverage.
ieee, tran	
high, *large*, *single*, *low*	$\delta = 0.0199$. OPUS Miner finds {`high, low`} (lev. 0.0871), but adding either of the frequent words *large* (count 1,090) and *single* (count 1,038) greatly lowers leverage.
pattern, *recognition*, *task*	$\delta = 0.0405$. This itemset fails OPUS Miner's strict statistical test for productivity, $p = 7.96E{-}12$, adjusted α for size 3 itemsets $3.96E{-}14$.
classification, *recognition* *	While this itemset is nonredundant and productive, it is not independently productive with respect to 23 of its supersets, including {`classification,pattern,recognition,training`}.
rate,rates	

further 3 to be nonredundant and productive but not independently productive. For example, although {consider, define, defined} is nonredundant and productive, OPUS Miner finds that it is not independently productive relative to {consider, define, defined, finite}.

From these one-to-one comparisons, we find that all three methods identify key concepts in the data. Krimp and mtv detect more long itemsets, but we also see that they tend to more easily include less strongly related items. For this data, Krimp and mtv require strong minimum support constraints, whereas OPUS Miner finds itemsets with support as low as 0.003.

By modeling the data by the MaxEnt principle, mtv employs a prior that considers all available information *optimally*. Moreover, its search strategy is to iteratively find the optimal, in terms of likelihood, addition to the current model. We see here that this approach can provide particularly clean results. However, as querying this MaxEnt model requires exponential time in k, the size of the model, mtv is hence only applicable for mining relatively high-level summaries of data. As OPUS Miner does not model the full joint distribution, but instead considers local marginal distributions with respect to subsets and supersets, it can mine equally clean, yet much larger top-k self-sufficient itemsets. Where the key interactions in the data cannot be described in just 20 or so itemsets, OPUS Miner can provide much more detail.

Even though it was proposed first, Krimp can be regarded as a faster, more greedy, version of mtv. Instead of using an optimal MaxEnt model, it encodes the data heuristically, thus avoiding the exponential runtime in the size of the model. As such, like OPUS Miner, Krimp can provide much more detail about the data. However, although the Krimp results include comprehensible and informative itemsets, we do find relatively many variants of each returned itemset, often mixing different concepts into very long itemsets—and, we see that many of the itemsets Krimp returns have low leverage. An example of such an itemset is {abstract, function, input, introduction, network, neural, number, output, references, result, set, system, weight} (leverage−0.0001). This behavior seems to follow from its heuristics. In particular, when calculating likelihoods, it assumes independence between the itemsets in the model while there is strict dependence (nonoverlap, large itemsets first) in its encoding scheme. As a result, Krimp may underestimate the likelihood of the data under the model—essentially, for larger code tables, its prior becomes more and more uninformative—and so itemsets providing only small additions in likelihood may still be accepted into the model. As OPUS Miner does not aim to model the full joint distribution, it does not face this problem.

As a final comment in this comparison, we note that the choice between top-k mining (as OPUS Miner does), or (k-)pattern set mining (as Krimp and mtv perform) is subtle, and there is no general best solution. If one is after the best description of the data in k nonredundant terms, the latter approach makes the most sense. However, one has to appreciate that choices in the search process may affect which itemsets are reported in the final model. If one is not so much interested in a model of the data, but rather wants to obtain a more general overview of the top-most associations, self-sufficient itemsets are a natural choice. We leave it to the individual reader to judge for this case study the qualitative differences between the itemsets found by each of Krimp and mtv that are accepted as self-sufficient by OPUS Miner and those that are not (set in italics in Tables IX and X), and between those that are selected as the top 25 by each approach (Tables II, IX, and X).

8.5. Efficiency

Next, we seek to assess the computational efficiency of OPUS Miner, including how it scales with increasing k and increasing data quantity, and the contribution to

Table XI. Datasets

Dataset	Transactions	Items
accidents	340,183	468
chess	3,196	75
connect	67,557	129
kosarak	990,002	41,270
mushroom	8,124	119
pumsb	49,046	2,113
pumsb_star	49,046	2,088
retail	88,162	16,470
T10I4D100K	100,000	870
T40I10D100K	100,000	940
webdocs	1,692,082	5,267,656

Table XII. Time in Seconds for Top 50 Itemset Discovery

Dataset	Lev	Lift
accidents	122.46	0.03
chess	0.71	0.02
connect	12.77	0.04
kosarak	4.26	0.46
mushroom	0.27	0.01
pumsb	3.00	0.06
pumsb_star	3.01	0.04
retail	0.73	5.78
T10I4D100K	2.50	0.09
T40I10D100K	18.46	1.33
webdocs	117.57	1.88

computational efficiency of its key mechanisms. To this end, we ran it on each of the datasets from the FIMI Repository [Goethals 2012], described in Table XI. The accidents data is due to Geurts et al. [2003] and the retail data to Brijs et al. [1999]. All results are averages over five runs. These experiments were performed on a virtual dual single-core CPU 8Gb RAM Linux machine running on a Dell PowerEdge 1950 with dual quad-core Intel Xeon E5410 processors running at 2,333Mhz with 32Gb of RAM.

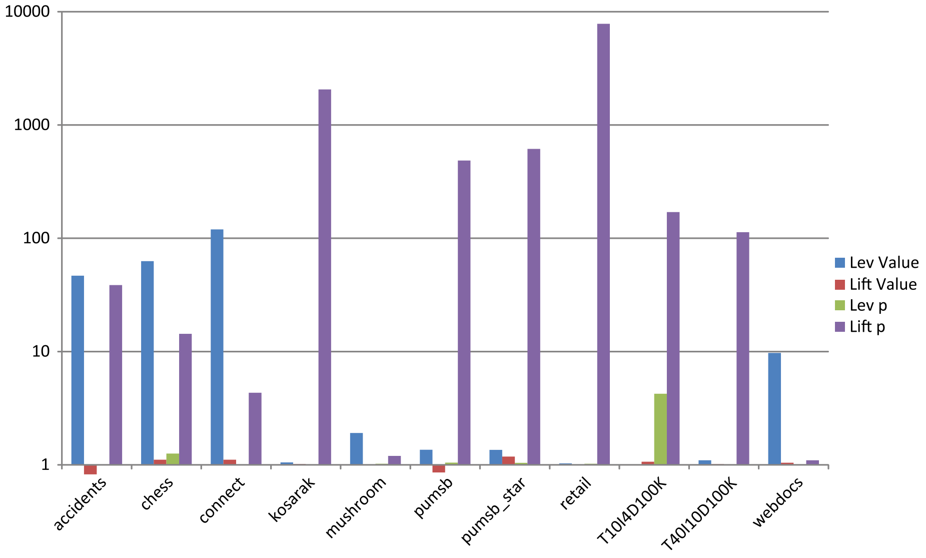
Running times for individual runs that registered less than the minimum registerable time of 1 tick (1/100 of a second) were rounded up to 1 tick. The base value for k was set to 50, as this was the largest round number for which the ablation studies could be completed within the memory constraints of the experimental machine.

Table XII presents the average time for discovering the top 50 itemsets for each dataset. As can be seen, these times range from just 1/100 of a second for the mushroom data searching on lift to a little over 2 minutes for the accidents data searching on leverage. The time of less than 2 minutes for webdocs demonstrates that OPUS Miner can scale very effectively to datasets that contain both millions of transactions and millions of items.

A surprise result is that search by lift is always faster than search by leverage, and often many orders of magnitude so. The reasons for this are revealed in an ablation study in which we disable each of the pruning mechanisms, one using the bound on the value of an itemset and the other on the Fisher p-value.

Figure 1 presents the relative average time (the average time divided by the average time taken for standard search) for the ablation studies, plotted on a log scale. For each dataset, the leftmost bar represents the time taken for search by leverage when the bound on value is disabled divided by the time taken with all pruning enabled. The next bar represents the respective ratio for lift. The final two bars represent the respective ratios for when the bound on Fisher p-value is disabled. As can be seen, pruning on value often has high impact for search by leverage but has little impact on search by lift. Indeed, on two datasets, the average times actually decrease slightly, but we believe that this is an artifact of the inherent inaccuracy of the timing mechanism for extremely short time intervals. In contrast to the results for pruning using upper limits on value, pruning on Fisher p-value has little impact on search by leverage but large impact on search by lift.

We believe that the reason the effect of the pruning mechanisms is reversed for leverage and lift is because the upper bound on value is tight for leverage but not so tight for lift and that high leverage itemsets will usually pass a Fisher Exact Test but



This figure is plotted on a log scale due to the variability in values. Each bar represents the ratio of time taken without the pruning mechanism to the time taken with it.

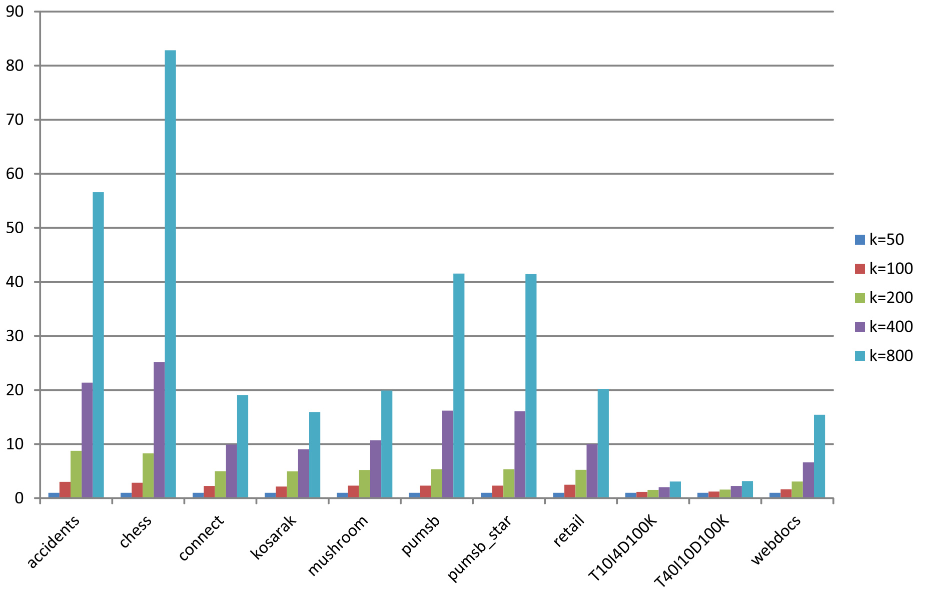
Fig. 1. Relative times without each pruning mechanism.

high lift itemsets often will not. For lift, the upper bound on value is not affected by the support of the current itemset and varies only with the support of the individual items in the itemset. Hence, it has little effect. In contrast, the tight upper bound for leverage allows very effective pruning of the search space. On the other hand, the individual items with the highest potential lift have very low support, and the upper bound on Fisher *p*-value can efficiently remove these from the search space, greatly reducing compute time. The resulting search by lift is extremely fast, because the items being considered first have low support (typically count of 3 or higher), and hence the set intersection operations and Fisher Exact Test computations are both very efficient.

Figure 2 shows the relative average time as k is increased for leverage. There is a 16-fold increase in k from the base level of 50 to the highest level of 800. As can be seen, for four of the datasets (accidents, chess, pumsb, and pumsb_star), the increase in computation is superlinear relative to the increase in k; for another three (connect, mushroom, and retail), the increase is approximately linear (16-fold); and for the remaining four, it is sublinear. We believe that this behavior depends on the distribution of top values in the search space. As the difference between the 50th and 800th value increases, the relative amount of computation required to find the additional itemsets will also increase.

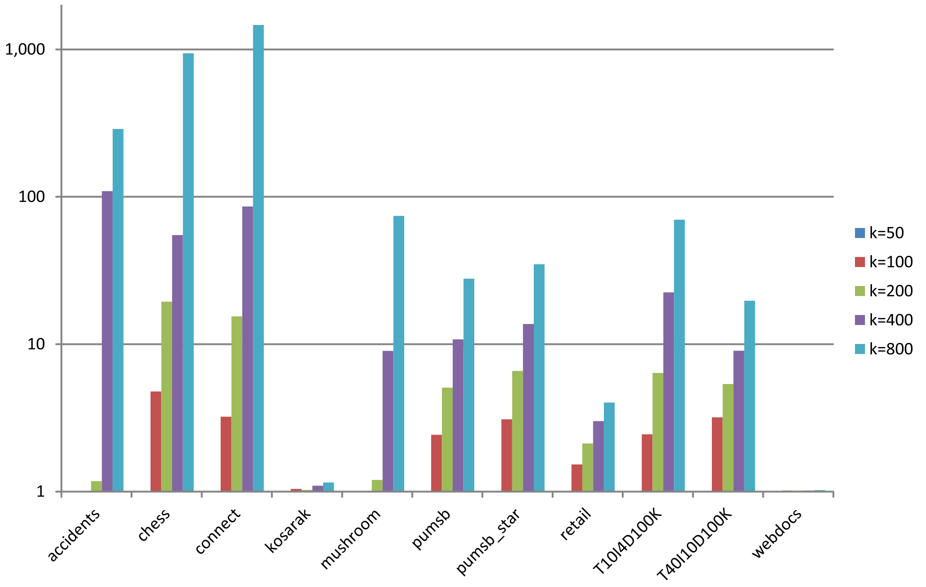
Figure 3 presents the equivalent results for lift, but with the relative average time plotted on a log scale. In the context of search by lift, the increases are much greater. This is because lower values for lift will be associated with higher support items and itemsets, and hence the computation for both set intersection and the Fisher Exact Test increases greatly, as does the size of the itemsets that must be investigated.

Figure 4 shows how OPUS Miner scales as data quantity increases. For each of the three types of synthetic data examined in Section 8.1, CPU time in seconds is plotted as data quantity increases. These experiments were conducted on a heterogeneous grid system, and hence the times should be treated as indicative only as different runs may have been executed on slightly different hardware under different operating conditions.



Each bar represents the time taken for association discovery with the given value of k divided by the time taken for $k = 50$.

Fig. 2. Relative times as k increases, leverage (δ).



Each bar represents the time taken for association discovery with the given value of k divided by the time taken for $k = 50$. This Figure is plotted on a log scale due to the variability in values. For webdocs no value of k takes more than twice as long as $k = 50$, which takes on average just 1.88 seconds.

Fig. 3. Relative times as k increases, lift.

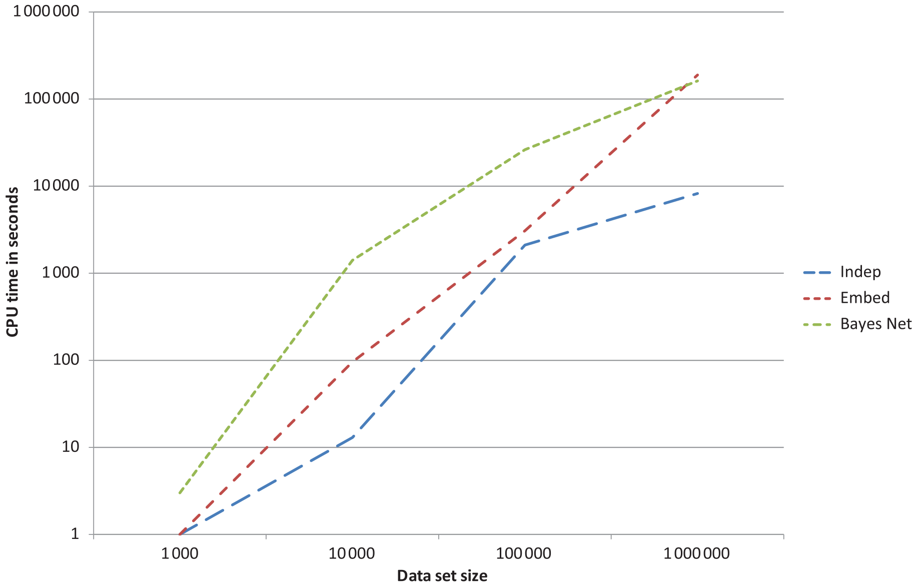


Fig. 4. Scalability with respect to data size. The figure is plotted on a log scale as the dataset sizes grow at a polynomial rate.

These plots do reveal that the time grows at a polynomial rate. It is superlinear because, as discussed in Section 6, greater data size allows itemsets with lower support to pass the statistical tests, and thus the number of itemsets that must be explored grows as the data quantity grows.

In summary, with k up to 100, OPUS Miner can find self-sufficient itemsets in no more than a few minutes, even for a dataset containing more than a million records and more than five million items. Of the two pruning mechanisms, pruning on optimistic value is more effective when searching by leverage, whereas pruning on statistical significance is more effective when searching by lift. Finding the top-k itemsets by lift is more efficient than finding the top-k itemsets by leverage. OPUS Miner's scalability as k increases varies greatly from dataset to dataset. In the best case, compute time scales sublinearly with k. In the worst case, a 16-fold increase in k results in more than a 1,000-fold increase in compute time. OPUS Miner scales at a polynomial rate with respect to data quantity, as greater numbers of examples allow more subtle patterns to pass its strict statistical testing regime, thus increasing the size of the search space that must be explored.

9. CONCLUSIONS

We have presented an algorithm for discovering top-k productive and nonredundant itemsets, with postprocessing to identify those that are not independently productive. This algorithm can be used with any well-behaving measure of interest. It is highly efficient for two measures, lift and leverage, despite both requiring very computationally intensive evaluation of all binary partitions of an itemset.

We also present a new upper bound on well-behaving measures of interest. We have shown how this bound and Hämäläinen's [2010] lower bound on the p-value of a Fisher Exact Test can both be used to greatly prune the search space. Pruning on the value of the measure of interest is more effective than pruning on the Fisher p-value for measures like leverage for which itemsets with high values are likely to pass the Fisher test. Pruning on the Fisher p-value is more effective than pruning on the measure of interest for measures like lift, for which itemsets with high values are likely to fail the Fisher test.

It would be valuable to push the constraints on independent productivity into the core search process in order to support search for top-*k* self-sufficient itemsets. On the face of it, this appears infeasible, as it implies that for every potential itemset, all immediate supersets should be assessed for self-sufficiency. More investigation is warranted as to whether it is possible to prune this massive search space down to a feasible computational task.

There is potential for speed-up and reducing memory use by varying some of OPUS Miner's design choices. Either best-first or a mix of depth and breadth-first search might provide advantages relative to OPUS Miner's depth-first strategy. Other alternatives worth exploring include more sophisticated strategies for selecting which itemset counts to memoize. However, the greatest potential for improving performance probably lies in the development of further bounds on itemset value and Fisher *p*-value and pruning mechanisms that utilize them.

OPUS Miner uses a very strict set of statistical tests with a strong correction for the multiple comparisons problem. This makes it very conservative in deciding whether to accept an itemset as productive. It is possible that if this could be relaxed, more large itemsets would be found that would allow more of the smaller itemsets to be suppressed as not independently productive.

It is clear from our case studies that the statistical approach of OPUS Miner and the information theoretic alternatives each have relative strengths and weaknesses. OPUS Miner is more conservative, which is an advantage in some contexts and a disadvantage in others. It would be useful to investigate these relative strengths and weaknesses in more detail and to consider where there are any opportunities for each approach to borrow key elements from the other.

The source code for the system can be downloaded from http://sourceforge.net/projects/opusminer. We believe that this software serves as a practical demonstration of the feasibility of using self-sufficient itemsets to find succinct summaries of the key associations in data.

ACKNOWLEDGMENTS

We are grateful to Tijl De Bie, Wilhelmiina Hämäläinen, Michael Mampaey, Francois Petitjean, and Nikolaj Tatti for valuable comments and suggestions on drafts of this work. We are further indebted to Michael Mampaey for his assistance with the experiments using MTV.

REFERENCES

R. Agrawal, T. Imielinski, and A. Swami. 1993. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM-SIGMOD International Conference on Management of Data*. 207–216.

Y. Bastide, N. Pasquier, R. Taouil, G. Stumme, and L. Lakhal. 2000. Mining minimal non-redundant association rules using frequent closed itemsets. In *Proceedings of the 1st International Conference on Computational Logic (CL'00)*. Springer-Verlag, Berlin, 972–986.

R. J. Bayardo, Jr., R. Agrawal, and D. Gunopulos. 2000. Constraint-based rule mining in large, dense databases. *Data Mining and Knowledge Discovery* 4, 2–3, 217–240.

R. Brijs, G. Swinnen, K. Vanhoof, and G.Wets. 1999. Using association rules for product assortment decisions: A case study. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM Press, New York, NY, 254–260.

T. Calders and B. Goethals. 2002. Mining all non-derivable frequent itemsets. In *Proceedings of the 6th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKKD'02)*. Springer, Berlin, 74–85.

T. Calders and B. Goethals. 2007. Non-derivable itemset mining. *Data Mining and Knowledge Discovery* 14, 1, 171–206.

T. De Bie. 2011. Maximum entropy models and subjective interestingness: An application to tiles in binary databases. *Data Mining and Knowledge Discovery* 23, 3, 407–446.

A. W. C. Fu, W. K. Renfrew, and J. Tang. 2000. Mining N-most interesting itemsets. In *Proceedings of the 12th International Symposium on Foundations of Intelligent Systems*. 59–67.

A. Gallo, T. De Bie, and N. Cristianini. 2007. MINI: Mining informative non-redundant itemsets. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD'07). Lecture Notes in Computer Science*, Joost Kok, Jacek Koronacki, Ramon Lopez de Mantaras, Stan Matwin, Dunja Mladenic, and Andrzej Skowron (Eds.), Vol. 4702. Springer, Berlin/Heidelberg, 438–445.

L. Geng and H. J. Hamilton. 2006. Interestingness measures for data mining: A survey. *Computing Surveys* 38, 3, 9.

K. Geurts, G. Wets, T. Brijs, and K. Vanhoof. 2003. Profiling high frequency accident locations using association rules. In *Proceedings of the 82nd Annual Transportation Research Board*.

B. Goethals. 2012. Frequent Itemset Mining Implementations Repository. Retrieved April 26, 2014, from http://fimi.ua.ac.be/.

W. Hämäläinen. 2010. *Efficient Search for Statistically Significant Dependency Rules in Binary Data*. Ph.D. Dissertation. Department of Computer Science, University of Helsinki.

W. Hämäläinen. 2012. Kingfisher: An efficient algorithm for searching for both positive and negative dependency rules with statistical significance measures. *Knowledge and Information Systems* 32, 2, 383–414.

J. Han, H. Cheng, D. Xin, and X. Yan. 2007. Frequent pattern mining: Current status and future directions. *Data Mining and Knowledge Discovery* 15, 1, 55–86.

S. Hanhijärvi, M. Ojala, N. Vuokko, K. Puolamäki, N. Tatti, and H. Mannila. 2009. Tell me something I don't know: Randomization strategies for iterative data mining. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, 379–388.

S. Jaroszewicz, T. Scheffer, and D. A. Simovici. 2009. Scalable pattern mining with Bayesian networks as background knowledge. *Data Mining and Knowledge Discovery* 18, 1, 56–100.

E. T. Jaynes. 1982. On the rationale of maximum-entropy methods. *Proceedings of the IEEE* 70, 9, 939–952.

K.-N. Kontonasios and T. De Bie. 2010. An information-theoretic approach to finding noisy tiles in binary databases. In *Proceedings of the 10th SIAM International Conference on Data Mining (SDM'10)*. SIAM, Columbus, OH, 153–164.

J. Lijffijt, P. Papapetrou, and K. Puolamaki. 2012. A statistical significance testing approach to mining the most informative set of patterns. *Data Mining and Knowledge Discovery* 28, 1, 238–263. DOI:http://dx.doi.org/10.1007/s10618-012-0298-2

M. Mampaey, N. Tatti, and J. Vreeken. 2011. Tell me what I need to know: Succinctly summarizing data with itemsets. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, 573–581.

M. Mampaey, J. Vreeken, and N. Tatti. 2012. Summarizing data succinctly with the most informative itemsets. *ACM Transactions on Knowledge Discovery from Data* 6, 4, 1–44.

P. K. Novak, N. Lavrac, and G. I. Webb. 2009. Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup discovery. *Journal of Machine Learning Research* 10, 377–403.

N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. 1999a. Discovering frequent closed itemsets for association rules. In *Proceedings of the 7th International Conference on Database Theory (ICDT'99)*. 398–416.

N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. 1999b. Efficient mining of association rules using closed itemset lattices. *Information Systems* 24, 1, 25–46.

G. Piatetsky-Shapiro. 1991. Discovery, analysis, and presentation of strong rules. In *Knowledge Discovery in Databases*, Gregory Piatetsky-Shapiro and J. Frawley (Eds.). AAAI/MIT Press, Menlo Park, CA, 229–248.

J. Rissanen. 1978. Modeling by shortest data description. *Automatica* 14, 1, 465–471.

R. Rymon. 1992. Search through systematic set enumeration. In *Proceedings of KR-92*. 268–275.

A. Siebes, J. Vreeken, and M. van Leeuwen. 2006. Item sets that compress. In *Proceedings of the 6th SIAM International Conference on Data Mining (SDM'06)*. SIAM, Bethesda, MD, 393–404.

N. Tatti. 2008. Maximum entropy based significance of itemsets. *Knowledge and Information Systems* 17, 1, 57–77.

N. Tatti and M. Mampaey. 2010. Using background knowledge to rank itemsets. *Data Mining and Knowledge Discovery* 21, 2, 293–309.

N. Tatti and J. Vreeken. 2012. Comparing apples and oranges—measuring differences between exploratory data mining results. *Data Mining and Knowledge Discovery* 25, 2, 173–207.

C. Tew, C. Giraud-Carrier, K. Tanner, and S. Burton. 2014. Behavior-based clustering and analysis of interestingness measures for association rule mining. *Data Mining and Knowledge Discovery* 28, 4, 1004–1045. DOI:http://dx.doi.org/10.1007/s10618-013-0326-x

J. Vreeken, M. van Leeuwen, and A. Siebes. 2011. Krimp: Mining itemsets that compress. *Data Mining and Knowledge Discovery* 23, 1, 169–214.

C. Wang and S. Parthasarathy. 2006. Summarizing itemset patterns using probabilistic models. In *Proceedings of the 12th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD'06)*. 730–735.

G. I. Webb. 1995. OPUS: An efficient admissible algorithm for unordered search. *Journal of Artificial Intelligence Research* 3, 431–465.

G. I. Webb. 2000. Efficient search for association rules. In *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'00)*. ACM, New York, NY, 99–107.

G. I. Webb. 2006. Discovering significant rules. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'06).* ACM, New York, NY, 434–443.

G. I. Webb. 2007. Discovering significant patterns. *Machine Learning* 68, 1, 1–33.

G. I. Webb. 2008. Layered critical values: A powerful direct-adjustment approach to discovering significant patterns. *Machine Learning* 71, 2–3, 307–323.

G. I. Webb. 2010. Self-sufficient itemsets: An approach to screening potentially interesting associations between items. *Transactions on Knowledge Discovery from Data* 4, 3:1–3:20.

G. I. Webb. 2011. Filtered-top-k association discovery. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 1, 3, 183–192. DOI:http://dx.doi.org/10.1002/widm.28

G. I. Webb and S. Zhang. 2005. K-Optimal rule discovery. *Data Mining and Knowledge Discovery* 10, 1, 39–79.

X. Wu, C. Zhang, and S. Zhang. 2004. Efficient mining of both positive and negative association rules. *ACM Transactions on Information Systems* 22, 3, 381–405.

M. J. Zaki. 2000. Generating non-redundant association rules. In *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'00)*. ACM, New York, NY, 34–43.

M. J. Zaki and C. J. Hsiao. 2002. CHARM: An efficient algorithm for closed itemset mining. In *Proceedings of the 2nd SIAM International Conference on Data Mining*. 457–473.

A. Zimmermann. 2013. Objectively evaluating interestingness measures for frequent itemset mining. In *Proceedings of the Emerging Trends in Knowledge Discovery and Data Mining International Workshops (PAKDD'13)*, 354–366. http://link.springer.com/chapter/10.1007%2F978-3-642-40319-4_31.

Received September 2012; revised September 2013; accepted September 2013