Finding the most interesting correlations in a database: how hard can it be? ☆

Christopher Jermaine*

Computer and Information Sciences and Engineering Department, University of Florida, Gainesville, FL 32611-6120, USA

Received 23 December 2002; received in revised form 22 August 2003; accepted 26 August 2003

Abstract

This paper addresses some of the foundational issues associated with discovering the best few correlations from a database. Specifically, we consider the computational complexity of various definitions of the “top- k correlation problem,” where the goal is to discover the few sets of events whose co-occurrence exhibits the smallest degree of independence. Our results show that many rigorous definitions of correlation lead to intractable and strongly inapproximable problems. Proof of this inapproximability is significant, since similar problems studied by the computer science theory community have resisted such analysis. One goal of the paper (and for future research) is to develop alternative correlation metrics whose use will both allow efficient search and produce results that are satisfactory for users.

© 2003 Elsevier Ltd. All rights reserved.

1. Introduction

Perhaps the most useful tactic for measuring the level of *correlation* or *interestingness* or *surprise* of a set of events is by instead measuring the lack of independence of the events. One of the most straightforward examples of this strategy is the popular χ^2 test for independence [1] from statistics. An example of the application of this test is as follows. Say we want to check for correlation between the purchase of beer and the purchase of diapers at a supermarket. We first produce a contingency matrix for the four possible events as follows, based on our observations of the actual data:

	Diapers	No diapers	$\sum$ Row
Beer	1734	4589	6323
No beer	3474	45,722	49,196
$\sum$ Column	5208	50,311	55,519

For example, in the above table, beer appeared without diapers 4589 times, and beer appeared with diapers 1734 times. Then, we produce a similar matrix of *expected* values, based on an assumption of independence of the different events:

	Diapers	No diapers	$\sum$ Row
Beer	593.1	5729.9	6323
No beer	4614.9	44,581.1	49,196
$\sum$ Column	5208	50,311	55,519

☆ Recommended by Nick Koudas

*Tel.: +1-352-3922691; fax: +1-352-3921220.

E-mail address: cjermain@cise.ufl.edu (C. Jermaine).

For example, the *expected* number of purchases of beer without diapers (given our assumption of independence) is $5729.9 = 6323 \times 50,311/55,519$; the expected number of purchases of beer with diapers is 593.1. Next, we compute the χ^2 statistic, which is a sum over the four different cells in the contingency matrix of observed values:

$$\chi^2 = \sum_{r \in \text{matrix}} \frac{(\text{Obs}(r) - \text{Ex}(r))^2}{\text{Ex}(r)}.$$

In our case, this sum is 2732.8. Finally, by comparing with the χ^2 distribution, we obtain the probability that we would see such a large sum, were the events truly independent. In our case, this value is nearly zero, rejecting the *NULL* hypothesis and testifying to the correlation between the purchase of beer and diapers.

In this paper, we consider the computational implications of integrating statistical expectation of this type into the data-mining process. Specifically, we consider the so-called *iceberg cube* queries [2], where we allow the function in the HAVING clause of the proposed SQL CUBE-BY query [3] to measure deviation from independence. The queries we consider are basically data cube queries of the form:

```
CREATE CUBE Iceberg AS
  SELECT  $a_1, a_2, a_3, \dots, a_n$ 
  FROM SalesInfo
  CUBE BY  $a_1, a_2, a_3, \dots, a_n$ 
  HAVING INTEREST (*) IN TOP  $k$ 
```

where we will allow INTEREST to be some (potentially complex) statistical function that uses a notion of deviation from independence of attribute values to decide how interesting the subcubes are. We remind the reader that the CUBE-BY operator essentially iterates over all combinations of the attributes in the CUBE-BY clause, and that p , the number of data dimensions, may be very large in practice: on the order of 10^5 or more. Ideally, we will choose a formulation for the INTEREST function so that it is meaningful to report back only the few most important correlations or relationships from the database.

Formulations of this type of problem have been studied previously. For example, if INTEREST is

simply the COUNT (*) function, then the above query is equivalent to the frequent itemset mining problem [4,5]. Most work in this domain considered simple aggregate functions, like COUNT [2] (equivalent to *support* in market-basket analysis) and AVERAGE [6]. While clearly useful, researchers have commented on the limitations of such functions, precisely because they lack any notion of expectation [7]. Using support, for example, the level of correlation of two events is simply the fraction of times that they co-occur. As a metric, support facilitates fast search, but it has its drawbacks. For example, the finding that values A and B occur together in 98% of the database tuples is uninteresting, if A and B both occur in 99% of the tuples when considered in isolation. This would be expected since $P(A) = P(A|B)$ and $P(B) = P(B|A)$. Most simple statistical tests over A and B (including the χ^2 statistic) would reflect this lack of true correlation of A and B, even though A and B together would have very high support.

In the remainder of this paper, we consider the computational implications of moving beyond simple aggregate functions like COUNT or AVERAGE, to more rigorous analysis. For example, one could imagine basing the INTEREST function on the χ^2 statistic, or on both the χ^2 statistic and the COUNT function, which would render the query very similar to the problem addressed previously by Brin et al. [8]. Such a query might be useful, in that the user could choose a statistically meaningful correlation level, and receive back only the most significant subcubes. In this paper, we will explore the computational implications of choosing different bases for the INTEREST function, in order to determine how practical it is to integrate expectation into this framework.

1.1. The problem with river flows and stock prices

Correlation search was the first data-mining problem to be tackled by the database community, and in the last 10 years it has likely been the most widely studied. For evidence of this, one need only look at the hundreds of papers written in the last decade describing and evaluating algorithms to perform *association rule (AR) mining* and *frequent itemset (FI) mining*. For example, *Citeseer* [9] lists

more than 600 distinct papers that reference the AR mining paper by Agrawal and Srikant [5], and the *DBLP most referenced list* [10] shows 100 references to that paper, with an additional 111 references to the earlier paper by Agrawal et al. [4]. The use of correlation metrics that incorporate some notion of deviation from expectation has also received some research attention. Some of the papers taking this approach are listed in the References section [11–14]. However, despite this work and the wealth of algorithms now available for performing correlation search, the complexity analysis presented in this paper is motivated by the observation that there are many data sets that seem to defy useful analysis by any and all of the available correlation-search algorithms.

For example, we have spent considerable time with two data sets that we call the *Rivers* and *Stocks* data sets. The *Rivers* data set consists of many years of streamflow measurements for several hundred waterways located in California. At a given instant in time, the flow for each of the waterways is measured by the US Geological Survey, and each measurement becomes an attribute in a database tuple with several hundred numerical attributes (one for each river). The data can be discretized by creating several discrete buckets for the different flow levels on each river. The *Stocks* data set simply records whether the price of each of the stocks listed on the Standard and Poor's 500 rose or fell each day over the last 10 years. Thus, the *Stocks* data set has 500 binary (or 1000 unary) attributes, and one record for each day. The data sets can be accessed at <http://www.cise.ufl.edu/~cjermain/RiversStocks.html>.

For both of these data sets, we would like to be able to discover the most interesting or most powerful correlations present in the data. This type of problem is fundamental in the field of knowledge discovery in databases (KDD). In the *Rivers* data set, this means that we want to discover the few sets of rivers that are most closely correlated in terms of their flow levels. In the *Stocks* data set, we want to discover a few sets of stocks that tend to move up or down together.

Given this sort of problem, there are two classes of existing algorithms that could be used to attempt a solution: those based on generating all

patterns that exceed a specific *support level* (we refer to this technique as *enumeration*), and those that use a heuristic to find only the few most interesting rules from the data (we refer to this as *heuristic pruning*). Both approaches seem to fail on the *Stocks* and *Rivers* data.

For example, there are literally hundreds of *enumeration* algorithms in existence. These algorithms use various clever tricks to quickly enumerate all sets of events which co-occur with frequency that is greater than a certain threshold (known as a *support level*). Well-known examples include the Apriori [5], Max-Miner [15], and FP-Growth algorithms [16]. After the patterns have all been enumerated by the algorithm, some sort of correlation metric can be applied to each set, and the sets can be sorted based on the metric in order to report back only the best few. The problem with such an approach for the *Stocks* and *Rivers* data sets is that the number of patterns tends to increase exponentially with decreasing support. Because each record from each database contains so many events, and because there are so many correlations present, it turns out that whatever feasible support level is chosen, the best correlations all seem to be those with the lowest support. In other words, there are simply too many patterns to enumerate and check all of them exhaustively, no matter how fast the enumeration.

There are several other algorithms that rely on *heuristic pruning*. These algorithms ignore the search for all correlations meeting some threshold level, and instead search for only the best few rules in a database. There are several examples of these algorithms, including those of Bayardo and Agrawal [17], Webb [18], and Morishita and Sese [19]. However, there are two problems with applying such algorithms to our particular task.

First, these algorithms generally search for the best few *rules*, instead of the best few *correlations*, which is what we are more interested in. Rule induction is really a separate problem from correlation search. A set of events A might be considered highly correlated if each subset of A lacks independence with respect to every other subset of A . On the other hand, a rule $A \rightarrow B$ is interesting if the occurrence of B is not independent from the occurrence of A , or if there is some

evidence that an occurrence of A causes the occurrence of B . The fact that rule induction is closely associated with the notion of causation probably means that it should be used very carefully. Causal analysis is a difficult and often misleading undertaking,¹ and it is debatable whether causation can even be inferred from after-the-fact data analysis. For the *Rivers* and *Stocks* data sets, we are thoroughly uninterested in rule induction or in any notion of causation; we simply want to know the most correlated rivers in California, and the few sets of stocks whose members seem to move together.

A second problem with the heuristic pruning algorithms is that these algorithms use heuristics only to the extent that the heuristics can reduce the search space; the algorithms still typically report back an exact answer. If the heuristics cannot reduce the search space, then the algorithms will not terminate in a reasonable time period. It is our experience that the density and numerous correlations of both the *Stocks* and *Rivers* data sets can easily defeat this type of algorithm. For example, we have used the commercially available implementation of the well-known Magnum Opus algorithm [19] to try to find the 100 rules with the greatest *lift* from the *Stocks* data set, without a minimum support threshold. After 7 days of waiting, with the progress meter never having moved from 3%, we decided to abandon our attempt.

1.2. Foundational questions

The *Stocks* and *Rivers* data sets are not the only ones that tend to cause problems for existing algorithms. In fact, such “problem” databases appear to be common.

¹For a classic example of the perils of causal analysis, consider the migration of farmers from the eastern United States to Kansas in the late 1800s [20]. It was long known that Kansas was generally too dry to support farming without irrigation, but when a rainy decade coincided with the arrival of great numbers of new farmers to Kansas, it was assumed that cultivation of the land had altered weather patterns. The real causal relationship was the other way around: unusually wet weather caused more people to move to Kansas and begin farming. The eventual return to normal rainfall was then disastrous.

The unsuitability of the existing methods for use with such data has led us to become interested in some of the basic, theoretical issues associated with this type of problem. First, what is it about these particular problems that make them so hard? Are they really that difficult? Can we alter our definition of what a “correlation” is, and by so doing, arrive at an easier problem? If so, do those easier problems produce usable results? If not, can the problems that we really want to solve be approximated? These are some of the issues addressed in the paper.

The results of this paper are of interest beyond the database and knowledge discovery communities. For example, it may be interesting to theoreticians that the abstract problem most closely related to correlation search is not a famous, widely-studied problem like VERTEX-COVER or BIN-PACKING. Instead, correlation search seems to be most closely related to an obscure but elegant little problem called MAXIMUM-BICLIQUE (the problem of finding the biclique having the most edges in a bipartite graph). Though it is an elegant and frequently encountered problem, it is a difficult problem to study, and partially as a result of this difficulty, it is often ignored (it was not even proven NP-Hard until the year 2000 [21]). The number of relevant references in the theory literature is very small. Future work on the MAXIMUM-BICLIQUE problem would be greatly useful in addressing the problem of correlation search, and the results of this paper underscore the importance of the problem.

1.3. Related work

We close this introduction with a quick review of the related work.

Surprisingly, little existing work addresses foundational issues in data mining and knowledge discovery. Most of the published results that do address theoretic computational issues associated with AR, FI, and correlation search are somewhat incidental, in that they are results that were proven to show the difficulty associated with the specific problem addressed by the authors of the paper. We now survey the results published by the

database and KDD communities regarding the difficulty of various correlation-search problems. To our knowledge, the survey is sufficiently complete despite the very small number of papers mentioned.

The first paper to mention the complexity of the FI mining problem was authored by Gunopulos et al. [22]. They describe the problem of deciding whether there is a pattern of length t that occurs with frequency σ as being NP-complete. The hardness is easily proven by a reduction from the BALANCED-BICLIQUE problem [23]. A very similar result was described by Zaki and Ogihara [24], where they consider the complexity of different variations of the same problem. However, the results of these papers differ substantially from the contents of this paper. These papers address only FI mining, which considers the co-occurrence of a set of events to be equivalent to the correlation of a set of events (in other words, there is no notion of expectation built into co-occurrence problems; we will come back to this particular point in Section 1). Omission of expectation makes the results both easier to prove and less widely applicable.

All of the other complexity results that we know of in this particular area are concerned with the problem of inducing various types of rules from data. As mentioned in Section 1.1, mining for rules in a database is a very different problem from discovering correlations. From a complexity-theoretic point of view, rules are much more expressive than correlations, since they have left- and right-hand sides. More information can be encoded in a rule, and so the hardness of various forms of rule induction described in the literature is proven quite differently than in the proofs that we give. Intuitively, because of this expressiveness and added power, we might expect rules to be harder to find.

The most well-known result in this area is probably due to Morishita [25], who proved that finding a set of tests in order to minimize the entropy or maximize the variance with respect to a target variable when constructing a decision tree is NP-hard (Morishita also presents a very similar result in a second paper with co-author Jun Sese [18]). This is the rule-based work that is closest to

ours. On one hand, Morishita's result is somewhat related to the contents of this paper since the entropy metric he considers does make use of expectation. But again, the problem he studies is prediction of a user-specified target variable (such as a person's sex, or whether she responded positively to a credit card solicitation), which is quite different from our particular problem.

There are several other papers dealing with the complexity of various rule-based problems. Rastogi and Shim have proven that the problem of determining whether there are any optimized association rules that exceed given support and confidence levels is NP-hard [26]. Optimized rule mining is the problem of instantiating a rule template of the form $U \wedge C_1 \rightarrow C_2$, where C_1 and C_2 are events or conditions chosen beforehand by the user; the events present in U are then chosen by the algorithm. In another paper, Fujisawa et al. have proven that mining optimized two-dimensional association rules is NP-hard [1]. This is the problem of partitioning the domains of two categorical attributes so as to predict the value of a target variable with maximum confidence, given a minimum support bound. The results presented in this paper address one of the problems left open by Fujisawa et al., namely, the approximability of biclique problems. In a third paper, Wijssen and Meersman have studied the complexity of mining general quantitative association rules [27] (rules over data with quantitative attributes). Again, a fundamental difference between these papers and our own work is that they deal with the problem of discovering rules from data, rather than simply discovering correlations.

1.4. Paper organization and contributions

The remainder of the paper is organized as follows. In Section 2, we discuss the notion of correlation as lack of independence. In Section 3, we present a bit of notation. In Sections 4 and 5, we consider the computational complexity of finding correlations based upon metrics that make use of probabilistic expectation. Several complexity results are given for various formulations of the problem. In Section 6, we consider basing search on the loglikelihood ratio, a parametric test from

statistics. In Section 7, we consider an alternative to the loglikelihood ratio which results in a problem that can be solved efficiently. Also, in Section 7, we present a hardness-of-approximation result for the MAXIMUM-BICLIQUE problem (and thus the related correlation search problems) that is stronger than any previously developed in the theory literature, as the MAXIMUM-BICLIQUE problem has previously resisted attempts at proving similar results (see Feige [28]). The paper is concluded in Section 8.

2. Notation

We next turn to some notation.

We assume that a database DB is composed of p attributes (n denotes the size, in tuples, of the database). We assume that each attribute is binary. This is not too restrictive since numerical attributes can be discretized (mapped into buckets) and encoded in binary; categorical attributes can be encoded as well. A database tuple is then a p -dimensional tuple over the database domain (or feature space) 2^p .

We define the notion of a *subcube*, which can be thought of as nothing more than an itemset from association rule mining, with the addition that we allow itemsets characterized by the *absence*, as well as presence, of items. A subcube can also be thought of as one of the subsets of attributes which are iterated over during the evaluation of a CUBE-BY query.

Definition 1. A *subcube* ς is any hyper-rectangle (or rectilinear region) from the binary data

space 2^p . It is specified by a p -dimensional vector of items from the domain $\{0, 1, *\}$. “*” is a wild card matching any value. We define $Obs(\varsigma)$ to be the number of tuples from DB falling in ς , and $D(\varsigma)$ (the “dimensionality” of ς) is the number of non-“*” values in the specification of ς .

For example, consider the database of Fig. 1. Let $\varsigma_1 = (1, *, *, *, 0, *, *, 0, *, *, 0, *, *)$. This subcube is a specification for all tuples where Item 1 was purchased by a customer more than 20 years old, more than 20 dollars was spent in total, and the purchase happened in the years 2000 or 2001. Since tuples 7 and 8 both fall in this subcube, $Obs(\varsigma_1) = 2$. Since ς_1 contains limitations in 4 dimensions, $D(\varsigma_1)$ is 4. Note that as the dimensionality of a subcube increases, the portion of the data space covered by the subcube decreases.

Definition 2. If a subcube has only positive (or ‘1’ values) in its specification, we will refer to it as a *restricted* subcube.

We note that AR and FI mining have traditionally been concerned with the search only for restricted subcubes.

If we project the subcube along a dimension, we measure the observed support in that one dimension:

Definition 3. The projected support of a subcube ς along dimension i is defined as

$$P(\varsigma, i) = \frac{|\{t | t \in DB, t[i] = \varsigma[i]\}|}{n}$$

Tuple	Items Purchased				Customer Age			Total \$\$ Spent			Purchase Year		
	#1	#2	#3	#4	0-20	21-40	41-80	0-20	21-30	31+	2000	2001	2002
1	1	0	1	1	1	0	0	0	0	1	1	0	0
2	0	1	0	0	0	1	0	1	0	0	1	0	0
3	1	0	0	0	0	1	0	1	0	0	1	0	0
4	1	0	0	1	1	0	0	0	1	0	0	1	0
5	0	1	1	0	1	0	0	0	1	0	0	1	0
6	1	1	1	1	1	0	0	0	0	1	0	0	1
7	1	1	0	0	0	0	1	0	1	0	0	0	1
8	1	0	0	1	0	0	1	0	1	0	0	0	1

Fig. 1. A hypothetical, 13-attribute binary database.

For example, $P(\varsigma_1, 1) = 3/4$. That is, Item 1 was purchased in 3/4 of the tuples in the database. If we assume no relationships among attributes (as in the χ^2 test; this assumption is known as AVI, or attribute value independence), then the expected support of a subcube can be defined using the projected support:

Definition 4. The *expected support* or *probability* of a subcube ς is $P(\varsigma) = \prod_i P(\varsigma, i)$.

In our example, $P(\varsigma_1) = \frac{3}{4} \times \frac{1}{2} \times \frac{3}{4} \times \frac{5}{8} = 0.176$. That is, given AVI, we expect this subcube to contain 17.6% of the tuples from the database.

Finally, we define our notion of a *correlation function*. This is a general class of functions whose purpose is to measure whether or not a given subcube is considered dense.

Definition 5. A function $f(n, p, L) \rightarrow \mathbf{R}$ is a *correlation function* if, given a database size n , and a subcube probability $P(\varsigma) = p$:

- (a) For fixed L , f is a non-decreasing function with respect to p , and
- (b) $f(n, p, L_1) < f(n, p, L_2)$ for $L_1 < L_2$.

This is meant to represent a very broad class of functions. Intuitively, a correlation function returns the minimum value for $Obs(\varsigma)$ for which a subcube is to be considered interesting at some level L . The higher the level L , the more interesting the subcube must be to satisfy the level, and the higher the required $Obs(\varsigma)$ value.

For example, the support function $f_{Sup}(n, pr, s) = s \times n$ is an extremely simple correlation function, accepting all subcubes having a support greater than s . This is, in fact, exactly the correlation measure used in classical AR mining.

3. Use of the most obvious correlation function

Our purpose in this work is to explore the complexity of solving problems of the form:

“Find for me any subcube ς for which $Obs(\varsigma) \geq f(n, P(\varsigma), L)$ for correlation level L .”

Note that this is essentially equivalent to the CUBE-BY query we described in Section 2, though it is a somewhat relaxed version of the problem since we are looking for only one subcube which has INTEREST at level L (rather than the best few). However, being able to solve this relaxed problem is a necessary and sufficient condition for being able to solve the top- k problem.

We begin our analysis at an obvious starting point. Perhaps, the simplest correlation function that would take into account expectation is as follows. Let $f_{Rat}(n, pr, L) = n \times pr \times L$. f_{Rat} is very simple: using it in the above problem will find some subcube which contains more than L times as many tuples as expected. However, it may be somewhat surprising that solving even this simple problem is probably not practical:

Theorem 3.1. *Given a level L , the problem of finding any restricted subcube where $Obs(\varsigma) \geq f_{Rat}(n, P(\varsigma), L)$ is NP-hard.*

Proof. All proofs are included in Appendix A. □

In a sense, this is a rather extraordinary result: simply finding whether there is a single itemset in a database which has fractionally more tuples than expected is not possible in polynomial time. Does this mean that any search through high-dimensional data based on expectation is impractical? Not necessarily.

Part of the difficulty of this problem stems from having to identify subcubes which contain attribute values with high expected support. Not coincidentally, in traditional AR mining, such attributes have largely been ignored, or assumed not to exist. AR mining is usually used only to look for the presence (and not absence) of items in market baskets, and except for some of the work on mining maximal itemsets [15], there has usually been an assumption that a given item (or at least most items) is/are typically found in only a fraction of the total market baskets.

What happens if we take a hint from AR mining and restrict ourselves to look only for subcubes containing no attribute values with a projected probability of greater than α ? We will call such subcubes α -subcubes. For example, the subcube $(*, 1, 1, 1, *, *, *, 1, *, *, *, *, *)$, which corresponds to baskets purchased by youngsters that contain items 2, 3, and 4, is an α -subcube for an α -value of 0.5. That is, attributes 2, 3, 4, and 8 all have positive values less than 50% of the time. If we restrict the problem in this way, then the computation of a solution is more reasonable:

Theorem 3.2. *Let $d_{\max}$ be the maximal value for $D(\zeta)$, evaluated over all occupied, restricted subcubes. Given a level L , finding some restricted α -subcube where $Obs(\zeta) \geq f_{Rat}(n, P(\zeta), L)$ is possible in $O(n^2 \times d_{\max}^{-1 + \log_{1/\alpha} n})$ time.*

A simple corollary of Theorem 3.2 is:

Corollary 3.3. *For any fixed value of α , the problem of finding some restricted α -subcube where $Obs(\zeta) \geq f_{Rat}(n, P(\zeta), L)$ falls in the class $\tilde{P}$.² Equivalently, if we assume that $NP \not\subseteq \tilde{P}$, then the problem is not NP-hard for fixed α .*

The significance of this is that while the problem may still require exponential time with respect to database size, it is not necessarily intractable for reasonable values of α , n and L . The fact that the problem falls in $\tilde{P}$ indicates that it might be considerably easier than the hardest problems in NP (like SAT, CLIQUE, VERTEX-COVER, and so on). In fact, the time required to solve this problem on modern computer hardware is probably small enough to be acceptable, especially when combined with techniques like random sampling. For example, consider the use of f_{Rat} in a market-basket setting. For a given retailer, we might expect that the most frequently purchased items are encountered in less than 10% of the register purchases, and the largest transaction

contains 100 different items. According to Theorem 3.2, we could process a sample of 10,000 market baskets, and we would require only around 10^{14} computational steps. With a well-designed algorithm, the number of steps required might be orders of magnitude less. On modern computer hardware, and with careful attention paid to algorithm design, we might expect such a search to complete in only a matter of minutes or seconds.

4. More reasonable correlation functions

4.1. Problems with f_{Rat}

While use of f_{Rat} might well be computationally feasible, it might not still be a reasonable correlation function to use. The problem is that it can favor anomalies rather than significant trends in the data. Consider this: every tuple from the database is contained in a cell in the data space 2^p . That cell defines a subcube. But the correlation level corresponding to that tiny cell is usually incredibly high. For example, consider again Fig. 1, and the subcube exactly enclosing tuple 1. The P value of this subcube is $\frac{5}{8} \times \frac{1}{2} \times \frac{3}{8} \times \frac{1}{2} \times \frac{1}{2} \times \frac{3}{4} \times \frac{3}{4} \times \frac{1}{2} \times \frac{1}{4} \times \frac{3}{8} \times \frac{3}{4} \times \frac{5}{8} = 2.7 \times 10^{-4}$. Yet, the f_{Rat} requirement is satisfied at a correlation level which is proportional to the inverse of this product, or 3682. This is clearly a very high correlation level; in fact, every tuple in the database will likely be considered interesting and surprising by this test, and the eight most interesting subcubes based on this metric correspond to single-tuple subcubes holding each of the eight tuples from the example database. The propensity of f_{Rat} to accept singleton tuples is closely related to the following, fairly simple result:

Observation 4.1. *For any occupied α -subcube ζ , it holds that $Obs(\zeta) \geq f_{Rat}(n, P(\zeta), L)$ for any value of $L < 1/n(1/\alpha)^{D(\alpha)}$.*

The implication here is that, for most reasonable correlation levels, finding the most interesting restricted subcube is quite easily done. Specifically, we simply report back the occupied, restricted α -subcube with the smallest value for $P(\zeta)$ over all

² $\tilde{P}$ is the set of problems solvable in *quasipolynomial* time, which is $O(2^{poly(\log n)})$. Though there is currently some debate on the issue, it is usually assumed that these are easier than the hardest problems in NP.

subcubes in the database, which can be done in a single, linear scan. If we consider a market-basket setting, we can expect that this subcube will correspond to the largest (or one of the largest) register transactions. For example, imagine that the largest register transaction contained in a database consists of 150 items that were purchased at one time by a single customer, and each of the individual items purchased in this transaction appeared in less than 25% of the database transactions. If the database size was 2^{30} , this single register transaction will correspond to a subcube accepted by f_{Rat} for any correlation level of less than 2^{270} . However, a single register transaction is hardly something that most people are interested in.

An equally serious problem with f_{Rat} is that it becomes totally unusable if we do not limit our search to restricted subcubes (that is, we allow our search to range over subcubes that are characterized by both the presence or absence of attribute values). This is demonstrated by the following:

Theorem 4.2. *Given a database, let ς be the subcube such that $Obs(\varsigma) \geq f_{Rat}(n, P(\varsigma), L)$ is satisfied for the largest possible value of L . Then ς is an occupied subcube where $D(\varsigma) = p$.*

This theorem says that according to f_{Rat} , the most interesting, unrestricted subcube is *always* limited in all of the database attributes. The problem that is made very clear by this theorem is that since this subcube will correspond to a single event or register transaction in most real-life situations, it is hardly an interesting correlation to report back to the user. Thus, Theorem 4.2 creates doubt about the utility of the correlation function for use with unrestricted subcubes.

4.2. Improving f_{Rat}

To fix these problems with f_{Rat} , we again consider the classical AR mining problem:

“Find all subcubes where $Obs(\varsigma) \geq f_{Sup}(n, P(\varsigma), s)$.”

What if we combine f_{Sup} and f_{Rat} ? That is, let:

$$f_{SupAndRat}(n, P(\varsigma), (s, L)) \\ = \max\{f_{Sup}(n, P(\varsigma), s), f_{Rat}(n, P(\varsigma), L)\}.$$

The function is shown above in Fig. 2. This would give us a simple variation on AR mining, where we integrate expectation into the definition. A subcube (or itemset) is interesting if it both attains a minimum support level and has fractionally more tuples than expected. The minimum support level will help to discount the statistically insignificant subcubes that are favored by f_{Rat} , and facilitate the search for both restricted and unrestricted subcubes (in contrast to f_{Rat} , which, according to Theorem 4.2, is unusable when searching for unrestricted subcubes). In fact, no result similar to Theorem 4.2 holds for $f_{RatAndSup}$. However,

Theorem 4.3. *Given a level L and a minimum support s , the problem of finding any unrestricted subcube where $Obs(\varsigma) \geq f_{RatAndSup}(n, P(\varsigma), (s, L))$ is NP-hard.*

An important corollary follows immediately from the proof of Theorem 4.3:

Corollary 4.4. *Even for a fixed value of α , search for unrestricted α -subcubes based upon $f_{RatAndSup}$ is still NP-hard.*

Thus, adding a minimum support threshold to f_{Rat} in order to add a notion of statistical

Fig. 2. A plot of the $f_{SupAndRat}$ correlation function.

significance accomplishes several things. First, it will likely improve the quality of the result of the search. Second, it seems to make search for subcubes characterized by the presence or absence of attribute values meaningful. Finally, it renders the search intractable in the general case, even when searching for α -subcubes at a fixed value of α (unlike f_{Rat} , which is tractable for fixed α). To summarize, $f_{RatAndSup}$ is a superior correlation function to f_{Rat} , but from a computational perspective, it is likely much harder to use.

Of course, there are other ways to ignore small subcubes. For example, what if we attempt to discount uninteresting subcubes by adding a minimum subcube size, or expected support? In other words, let:

$$f_{RatAndMinP}(n, P(\zeta), (MinP, L)) = \infty$$

if $P(\zeta) < MinP$; $f_{Rat}(n, P(\zeta), L)$ otherwise.

This correlation function is shown above in Fig. 3. Again, the attempt here is to avoid reporting subcubes which are trivial. This function is probably somewhat inferior to $f_{RatAndSup}$ in that it discounts complex patterns only because they are complex, and not based upon any characteristic of the data. Still, it does handle the problem of statistical significance to a certain extent, and it does not seem to render search for unrestricted subcubes meaningless (as f_{Rat} does). However, search based upon this function is still intractable: **Theorem 4.5.** *Given a level L and a minimum subcube probability $MinP$, the problem of find-*

Fig. 3. A plot of the $f_{SupAndMinP}$ correlation function.

ing any unrestricted subcube where $Obs(\zeta) \geq f_{RatAndMinP}(n, P(\zeta), (MinP, L))$ is NP-hard.

We close this section by noting that our proof of Theorem 4.5 does not suggest that a similar result to Corollary 4.4 holds for $f_{RatAndMinP}$. In other words, our proof of the hardness of search based upon $f_{RatAndMinP}$ relies on the ability to search for unrestricted α -subcubes at an arbitrary value of α . Thus, we cannot rule out the possibility that we can efficiently search for unrestricted subcubes considered interesting by $f_{RatAndMinP}$, if we consider only α -subcubes for a fixed value of α . The existence of such a search algorithm might make search based upon $f_{RatAndMinP}$ practical in some situations. However, in the general case, both of these simple augmentations of f_{Rat} result in intractable problems.

5. Loglikelihood ratio

While more practical than f_{Rat} , the correlation functions of the last section still leave something to be desired.

Consider two subcubes ζ_1 and ζ_2 , where both subcubes have exactly twice the expected number of tuples. However, ζ_1 contains 1,000,000 tuples, while ζ_2 contains only 40 tuples. Since ζ_2 is relatively small (only 20 tuples are expected), it is quite possible that finding twice as many tuples as expected in ζ_2 is purely a chance occurrence.³ However, the density of the much larger region ζ_1 is surely not an accident: the probability of finding 1,000,000 tuples given that the true, underlying probability distribution would predict only 500,000 is extremely small. As such, it makes sense that we would prefer to report ζ_1 over ζ_2 . However, if we use $f_{RatAndSup}$ and set s to be 40, both subcubes are reported back, or both are ignored. It is not acceptable to simply raise s in order to discount ζ_2 . Say that there were 10^{10} times

³We point out that there are an exponential number of subcubes in a given database. Due to this large number, many smaller subcubes of somewhat surprising density are likely to occur purely by chance.

as many tuples in ς_2 as expected (not unreasonable in correlated data). Would we want to lose such a dense subcube, simply because it contained only 40 tuples? Probably not.

Because of these limitations, in this section we describe a metric from statistics that would take into account the issues raised above, and remove the need for choosing several arbitrary parameters. We briefly consider (and dismiss) use of the popular χ^2 test in this domain, and settle instead on a test based on the *loglikelihood ratio* as an appropriate test. We show that, unfortunately, the use of the loglikelihood ratio to guide search through a data cube leads to an NP-hard problem definition. In response, in the next section we will introduce a relaxed variant on the log-likelihood ratio that produces a tractable problem definition.

5.1. The χ^2 test

The χ^2 test is arguably the most popular statistical check for correlation, and the fact that it is specifically designed for use with binned, categorical data (the data we consider in this paper) makes it, at least superficially, an attractive possibility for integration into a correlation search algorithm. A widely referenced paper [8] did just that, though their framework was a bit different than the one presented here.

We now briefly consider the use of the χ^2 statistic for high-dimensional correlation search. First, we wish to stress that, in general, use of the Pearson χ^2 statistic (as was described in Section 2) is not feasible in conjunction with this type of problem. The problem with this metric is that evaluation of the χ^2 statistic over a subcube ς requires time that is exponential with respect to $D(\varsigma)$, since there are $2^{D(\varsigma)}$ cells in the contingency table that must be considered. The result is that computation of the statistic itself is intractable for high-dimensional data. However, we can still use the basic idea behind the χ^2 statistic to create a reasonable correlation function. Specifically, we will define a function $f_{\text{ChiSquare}}$ that is equivalent to evaluation of the χ^2 statistic over only the one cell in the contingency table

associated with ς . That is, let:

$$f_{\text{ChiSquare}}(n, P(\varsigma), (L)) = \sqrt{nLP(\varsigma)} + nP(\varsigma).$$

The following theorem then describes the complexity of search based upon $f_{\text{ChiSquare}}$:

Theorem 5.1. *Given a level L , the problem of finding a single restricted subcube where $\text{Obs}(\varsigma) \geq f_{\text{ChiSquare}}(n, P(\varsigma), L)$ is NP-hard.*

However, it is encouraging that just like search based upon f_{Rat} , for many practical situations, $f_{\text{ChiSquare}}$ is in fact tractable.

Theorem 5.2. *For fixed α , the problem of finding a single, restricted α -subcube where $\text{Obs}(\varsigma) \geq f_{\text{ChiSquare}}(n, P(\varsigma), L)$ for some L is a member of the class $\tilde{P}$.*

Furthermore, while search based upon $f_{\text{ChiSquare}}$ is likely computationally feasible (just as for f_{Rat}), one advantage of the use of $f_{\text{ChiSquare}}$ compared to f_{Rat} is that it does not seem to break down when we expand our search to include unrestricted subcubes ($f_{\text{RatAndMinP}}$ and $f_{\text{SupandRat}}$ also both seem to share this desirable characteristic). Recall that when we expand our search to unrestricted subcubes, Theorem 4.2 stated that f_{Rat} will always favor individual cells from the data space and will be unable to capture general trends. From a purely mathematical point of view, $f_{\text{ChiSquare}}$ does not seem to share this undesirable property.

However, even given these benefits, the χ^2 statistic's general applicability for checking for correlation within the itemset framework is still very doubtful. The problem with this correlation function stems from the fact that a standard rule of thumb from statistics is that, for the χ^2 test of independence to be valid, *each possible event should expectedly occur at least five times* [1]. This requirement is simply unrealistic in many data-mining domains.

For example, in market basket applications, we can anticipate an upper limit of the order of 10^5 items (or more) that may be purchased (consider a large discount retailer like K-Mart or Wal-Mart). Imagine database recording market baskets in this domain, where each basket averages 20 items in

size. Say we want to apply the χ^2 test to an arbitrary, three-itemset from this domain, specified by the subcube ς . In the best case (without a skewed distribution of item purchases), to ensure that we would *expect* to find at least one such three-itemset in the database (that is, to ensure that $P(\varsigma) \times n \geq 1$), we would need a database with at least $((10^5)^3)/20 \approx 10^{14}$ transactions. That is, we would require that every person on Earth purchases tens of thousands of baskets! In many other realistic situations, we would need an even larger database.

The result is that for the vast majority of items, we rarely see them, and hence would not expect them to be purchased together. The problem is that the χ^2 test is understood to be invalid under such conditions. The situation deteriorates quickly (exponentially) when we move beyond three-way correlations to four-way, five-way, and higher degree correlations. For the test to be valid on itemsets containing even a handful of items, we would effectively need a database containing billions of billions of billions of transactions.

This problem cannot simply be ignored. In an excellent paper from computational linguistics, Ted Dunning [29] shows how the application of the χ^2 test to the domain of co-occurrence analysis in text can produce poor results. This domain is closely related to the market basket domain in that we have a relatively large number of possible “items” (words in the English language) and a database which is not large enough to validate the test. Using the χ^2 test on such textual data, Dunning shows how the test chooses a rather silly set of word-co-occurrences as being among the most important bigrams in a small corpus when the assumptions underlying the test are not met (for example: “instance 280”, “scanner cash”, “maturity hovered” and so on). Similar problems are likely to result in other problem domains as well.

5.2. Likelihood ratio

For this reason, some statistical texts recommend the use of a parametric *likelihood ratio* test under such circumstances. Briefly, we describe how it can be used to define a correlation function, f_{Lik} .

Our description of f_{Lik} is heavily influenced by a similar statistical test based on the loglikelihood ratio described by Dunning [29].

In general, a *likelihood function* is a function which is used to measure the goodness of fit of a statistical model to actual data. It is written as:

$$H(p_1, p_2, \dots; k_1, k_2, \dots),$$

where the variables $p_1, p_2, \dots$ describe the statistical model, and $k_1, k_2, \dots$ describe the data. A famous example of the use of a likelihood function is the Gaussian EM clustering algorithm, where the statistical model is a multi-dimensional normal distribution, and the task is to maximize the value of the likelihood function by altering the model parameters to fit a specific data set.

The concept of a likelihood function can easily be used to statistically test a given hypothesis, by applying the *likelihood ratio* test. Essentially, we take the ratio of the greatest likelihood possible given our hypothesis, to the likelihood of the best “explanation” overall. The greater the value of the ratio, the stronger our hypothesis is said to be.

To apply the likelihood ratio test to our subcube/itemset domain to produce a correlation function, it is useful to consider the binomial probability distribution. This is a function of three variables:

$$\text{Pr}_{bin}(p, k, n) \rightarrow [0:1].$$

Intuitively, the binomial function models the following situation. Suppose we try n separate and independent times to toss a ball through a hoop, and each attempt has a probability of success p . The probability of succeeding k times in those n trials is exactly $\text{Pr}_{bin}(p, k, n)$.

This is relevant to our problem definition, because we can consider a subcube ς as a “hoop” and tuple as a “ball”. Then, a database of size n is n attempts to hit the subcube (hoop) with a tuple (ball). Given our assumption of independence of all attributes or items, we predict that each trial has a probability of success $P(\varsigma)$.

The correlation of ς can then be measured by quantifying to what extent our assumption of attribute value independence was violated in practice. To do this, we perform the likelihood ratio test, comparing the binomial likelihood of

the fact that we observed $Obs(\varsigma)$ tuples in ς (when we thought the probability of ς was $P(\varsigma)$) with the best possible binomial explanation. The best possible explanation is that the probability of ς was *not* $P(\varsigma)$, but was instead simply $Obs(\varsigma)/n$.

Formally, the likelihood ratio in this case is

$$\frac{L(P(\varsigma), n, Obs(\varsigma))}{L\left(\frac{Obs(\varsigma)}{n}, n, Obs(\varsigma)\right)},$$

where L is derived from the formula for the binomial distribution, and is $L(p, n, k) = p^k(1-p)^{n-k}$. Commonly, this test is referred to as the *loglikelihood ratio* ($-2 \log \lambda$) and is computed as

$$\begin{aligned} -2 \log \lambda &= \log(L(P(\varsigma), n, Obs(\varsigma))) \\ &\quad - \log\left(L\left(\frac{Obs(\varsigma)}{n}, n, Obs(\varsigma)\right)\right). \end{aligned}$$

If we rewrite $-2 \log \lambda$ (the level of correlation) as L_{Lik} to conform with the notation of this paper, and solve for $Obs(\varsigma)$, we obtain a mathematical expression for the function $f_{Lik}(n, P(\varsigma), L_{Lik})$.

We argue that given the severe limitations of the χ^2 test in such a setting, f_{Lik} is a far preferable statistical metric for use in this framework.

5.3. Computational complexity

In the preceding section, we wished to present an accepted statistical measure that could be used to measure the level of correlation of a subcube, in order to explore the computational complexity of the application of such a measure to the itemset framework. Given the results presented earlier in the paper, the following results are probably not surprising.

Theorem 5.3. *Given a level L , the problem of finding a restricted subcube where $Obs(\varsigma) \geq f_{Lik}(n, P(\varsigma), L_{Lik})$ is NP-hard.*

Thus, according to Theorem 5.3, search based on a commonly accepted parametric statistical test is an intractable task in this setting.

6. The support-biased correlation function

At this point, search based upon nearly every reasonable correlation metric that we have considered has been intractable. Can we do anything about this complexity? One solution is to develop some sort of approximation algorithm, which *attempts* to find subcubes of high likelihood ratio, and perhaps finds them within some probabilistic or approximation guarantees. However, we can prove that finding and developing such approximation is probably not possible, as we will discuss shortly. A second solution is to somehow “tweak” the definition of the loglikelihood ratio correlation function, relaxing it somewhat (while trying to maintain many of its original characteristics) to arrive at a tractable problem definition. We will explore this second option briefly in this section.

To tackle these issues, we first discuss the nature of the f_{Lik} function. Consider the contours of the f_{Lik} function, for differing correlation levels. These contours are shown above in Fig. 4, for a 100-tuple, 100-dimensional database. We show the contours with respect to subcube observed support and subcube dimensionality. For simplicity, we assume that each of the 100 database dimensions has the same projected support ($\frac{1}{5}$).

Note that in the plot, with increasing dimensionality or pattern complexity, the observed

Fig. 4. Contours of the loglikelihood ratio correlation function. Contours are shown with respect to increasing subcube dimensionality. For example, a subcube of dimensionality 75 and support 75 is accepted by the second highest contour (or correlation level), but not by the highest.

support required for a subcube to be accepted is always decreasing. However, the *rate* of decrease continually slows, and the required support never reaches zero. It is significant that the contours of the f_{Lik} function, when plotted in this way, closely resemble a set of quadratic curves governed by the equation $Obs(\zeta) \times D(\zeta) = c$, where the constant c is a function of the correlation level L .

It turns out that because of the quadratic nature of these contours, search based upon this test (and any other, related test) is not only intractable, but is strongly inapproximable as well. To describe this property formally, we describe a simplified version of the f_{Lik} correlation function called f_{Quad} , which is characterized by a simple, quadratic relationship between the support of a pattern, and the pattern's complexity. That is, we will let

$$f_{Quad}(n, P(\zeta), (L)) = \frac{L}{\log P(\zeta)}.$$

Then, let $L_{Quad}(\zeta) = Obs(\zeta) \log P(\zeta)$. $L_{Quad}(\zeta)$ simply denotes the highest correlation level for which f_{Quad} considers a subcube ζ to be interesting. Given this function, we can show that finding a subcube that is anything *close* to the most interesting subcube in a database is likely intractable.

Theorem 6.1. *Given a database of size n , let ζ_{Opt} be the restricted subcube which maximizes L_{Quad} . Assume that $NP \not\subseteq \tilde{P}$. Then there exists some $\varepsilon > 0$ such that no polynomial time algorithm exists that can be guaranteed to find a restricted subcube ζ'_{Opt} where $L_{Quad}(\zeta'_{Opt}) \geq L_{Quad}(\zeta_{Opt})/n^\varepsilon$.*

This is a significant result because it implies that if we make the reasonable assumption that $NP \not\subseteq \tilde{P}$, then approximate search based on f_{Quad} (and those functions which, like most of those considered in this paper, can be used to solve for f_{Quad}) cannot be accomplished. In other words, any sort of reasonably accurate, approximate search based on f_{Quad} is “almost” NP-hard. It is also not too much of a stretch to conjecture that such an approximate search is actually NP-hard as well (see the proof for a discussion of this). Also, we conjecture that the largest value of ε for which Theorem 6.1 holds is not insignificant; in reality, it may be on the order of 0.5 or even more (though proof of a tight

lower bound on any approximation for f_{Quad} is beyond the scope of this paper).

This inapproximability of the search seems to leave us with two options. One option is to forgo any sort of guarantee, and simply rely on heuristic. As a matter of fact, it is our strong belief that this option must be seriously considered in future data-mining research, and is one of the motivations behind our work. A second option is to try to carefully design correlation metrics that *do* allow efficient search or approximation, but that maintain many desirable characteristics of a rigorous metric like f_{Lik} . We give an example of this second tactic now.

The quadratic nature of the f_{Lik} function is the source of the computational complexity associated with the use of f_{Lik} in iceberg cube queries. Were the curves not quadratic (and instead were linear), the problem would be tractable. This suggests a simple alternative to the loglikelihood-based correlation function. What if we alter it so that the contours are *linear* with respect to increasing subcube dimensionality? Consider the following definition:

Definition 6. The *support-biased correlation function* is

$$f_{SupBiased}(n, P(\zeta), (L, \alpha)) = \alpha n - L_g^{-1} \log P(\zeta).$$

When possible contours of the $f_{SupBiased}$ function are plotted in the same way as the plot of f_{Lik} in Fig. 4, we can see the difference between the two metrics. The plot of the support-biased correlation function (Fig. 5) is linear with respect to increasing subcube dimensionality. This clearly inhibits its value as an approximation of f_{Lik} at lower required levels of correlation, as the quadratic contours f_{Lik} drop far below the linear contours of $f_{SupBiased}$. But at higher levels, the approximation is clearly an adequate one. A particularly attractive characteristic of the support-biased correlation function is given in the following theorem:

Theorem 6.2. *Given a correlation level (L, α) , the problem of finding a restricted subcube where $Obs(\zeta) \geq f_{SupBiased}(n, P(\zeta), (L, \alpha))$ is solvable in polynomial time.*

Fig. 5. Contours of the support-biased correlation function, which provides a tractable approximation to the loglikelihood ratio-based search.

Thus, by changing the definition of the problem (weakening the definition of f_{Lik} somewhat), we can arrive at a tractable problem definition. Especially in a real database where correlations are often numerous and strong, this may be an acceptable modification to the problem, and is a good example of one tactic for handling the intractability of these problems.

7. Conclusions

This paper has been a work of theory, exploring the computational complexity of correlation search. Specifically, we considered the search for trends in the data which defy a simple assumption of attribute value independence. Metrics for measuring correlation strength that rely on lack of independence are very attractive for several reasons:

- First, such metrics are consistent with long-accepted notions of correlation and dependence from statistics.
- Second, they can be designed to naturally balance such factors as pattern complexity and the “support” of the pattern, which reduces or eliminates algorithmic parameter tuning.
- Third, by using a rigorous metric, it is meaningful to ask for the k strongest correlations in

the database, where the choice of k is independent of any of the characteristics of the data. Instead, the choice of k can be made according to the willingness and ability of an end-user to sort through the results of a search. This has the potential to transform the task of choosing algorithm parameters into a business-oriented decision, rather than the difficult, technical decision it is now.

As we have shown, the biggest drawback of this type of problem is that it likely does not lend itself to an efficient solution. This paper has shown that it is likely impossible to head “straight toward” the best few correlations. Furthermore, it is likely impossible to guarantee that we can head “straight toward” anything close to the best few correlations. Given this, we strongly advocate two future research directions:

- First, we advocate an emphasis on *heuristic* search for the types of correlations that we have discussed, whether or not the search is actually correct. This work should focus on methods that quickly find a small, high-quality set of answers that seem to make most users happy, most of the time (even if the users must live without any guarantees).
- A second research direction is careful development of correlation metrics that *do not* lead to intractability and inapproximation, but *do* still facilitate a small, precise, and fast answer. While these metrics might not be as rigorous as we would like, they would ideally still produce concise answers that would seem to be useful to the end user. An example of this type of metric is the “support-biased” correlation function from Section 7.

Considering the inherent computational hurdles discussed in this paper, these two research directions should present difficult and challenging problems for future work.

Appendix A

Proof of Theorem 3.1. The proof is a reduction from the MAXIMUM-BICLIQUE decision

problem. The maximum biclique problem is as follows: we are given a bipartite graph $G = (V, E)$ and an integer k , with two halves $A \subset V$, $B \subset V$ where $A \cup B = V$ and $A \cap B = \emptyset$. The problem is to determine if G contains two sets of vertices V_1 and V_2 such that $|V_1| \times |V_2| \geq k$ and $(v_1, v_2) \in E \forall v_1 \in V_1, v_2 \in V_2$. Long conjectured to be NP-complete, this fact was recently proven [21], and facilitates the proof of Theorem 3.1. In our proof, we will show that if we can check for an appropriate subspace in polynomial time, then we can check for a biclique in polynomial time.

We begin by asserting that the hardness of the maximum biclique problem is unchanged if we add the constraint that we only search for $|V_2|$ larger than $|B| \times \frac{3}{4}$. We call this restricted problem 3/4-RIGHT-EDGE-BICLIQUE. $\square$

Lemma A.1. *Solving 3/4-RIGHT-EDGE-BICLIQUE is NP-hard.*

Proof. To prove that 3/4-RIGHT-EDGE-BICLIQUE is NP-hard, we will use the reduction from the original proof of the NP-completeness of the maximum-edge biclique problem [21]. This proof, in turn, is a version of the well-known, “subtle” proof by David S. Johnson that BALANCED-COMplete-BIPARTITE-SUBGRAPH is NP-hard [23]. Both of these proofs use the same reduction from the problem of finding a 1/2 clique in a graph (a clique containing 1/2 of the graph’s edges). We will subsequently refer to this reduction as R .

To show how R can still be used to prove that 3/4-RIGHT-EDGE-BICLIQUE is NP-hard, we begin by assuming that we have a graph G where the predicate τ evaluates to true. Specifically, $\tau(G)$ is true if G does not have a 1/2 clique, and it is also true if G has a 1/2 clique that contains at least 3/4 of G ’s edges. Otherwise, $\tau(G)$ is false.

We begin with a claim that given a graph G such that $\tau(G)$ is true, the reduction R can still be used to produce a bipartite graph G' such that G' has a 3/4-RIGHT-EDGE-BICLIQUE if and only if G had a 1/2 clique. Why is this? Using R on G will produce a bipartite graph where the vertices of G form the “left” side of the bipartite graph, and the

edges from G form the “right” side of the bipartite graph (with a few additional “dummy” edges also appearing on the right side of the graph). As Peeters shows [21], G has a clique C containing 1/2 of G ’s vertices if and only if there is a biclique in the resulting bipartite graph that has more than j edges (for a certain number j). Specifically, this biclique will connect all of the vertices *not* in C with all of the edges that *are* in C , and also all of the additional “dummy” edges. R still works in conjunction with 3/4-RIGHT-EDGE-BICLIQUE as long as $\tau(G)$ is true, because in this case, we know that if C exists in G , then C must have 3/4 of G ’s edges.

We next assert that our requirement that $\tau(G)$ be true does not prevent us from checking for a 1/2 clique in a general graph. Given a graph G having n vertices, we can easily produce a new graph G' such that $\tau(G')$ is true and G' has a 1/2 clique if and only if G has one. We simply add two large sets of vertices (say, of size n^3) to G . One of those sets should be fully connected (that is, it should constitute a clique of size n^3). For that clique-forming set, we connect all of its vertices to all of the vertices that were originally in G via a set of $n \times n^3$ additional edges. The other set of additional vertices will have no associated edges. This will transform G into a new graph G' with the desired property.

Given this machinery, we can then give the proof that 3/4-RIGHT-EDGE-BICLIQUE is NP-hard. If we could solve this problem in polynomial time, then we could check for a 1/2 clique in a graph G in polynomial time as well. Simply use G to produce a graph G' such that $\tau(G')$ is true, as described above. Then, use R on G' to produce a bipartite graph. Use the algorithm that solves 3/4-RIGHT-EDGE-BICLIQUE to see if the resulting graph has a large enough biclique; if it does, accept G ; if not, then reject it. Thus, 3/4-RIGHT-EDGE-BICLIQUE is NP-hard.

We now return to our original goal: proving that search based on f_{Rat} is NP-hard. To do this, we will show how to reduce 3/4-MAX-EDGE-BICLIQUE to search based upon f_{Rat} . $\square$

Lemma A.2. *3/4-MAX-EDGE-BICLIQUE is polynomial-time reducible to search based upon f_{Rat} .*

Fig. 6. An example bipartite graph (left) and its simple database encoding (right).

Proof. The reduction relies on the following transformation. Given a bipartite graph G , we can encode it as a database with binary attributes as follows. Create a p -dimensional database ($p = |B|$) with $|A|$ tuples, with one tuple for each vertex in A . Each tuple in B corresponds to a dimension (attribute) in the database. The i th tuple in the database, t_i , has $t_i[j] = 1$ if $(v_i, v_j) \in E$ for $v_i \in A$, $v_j \in B$. $t_i[j] = 0$ otherwise. An example graph and corresponding database is shown above in Fig. 6.

The strategy will be to check for a subcube having a higher number of tuples than expected in the database; this subcube will correspond to a biclique in G . In order to force all d -dimensional subcubes to have the same probability (or expected support), we will normalize (equalize) the projected support of each dimension. To perform the normalization, determine the node of the greatest degree in B (call this degree m). Then for each node in v_i in B , determine its degree d . Add $m - d$ identical tuples to the database having $t[i] = 1$, and $t[j] = 0$ for any $i \neq j$. In our example, since the greatest degree in B is 3, we would add one tuple of the form $(1,0,0,0,0,0,0,0)$ (add one such tuple since the first node in B has degree 2), one tuple of the form $(0,1,0,0,0,0,0,0)$, one of the form $(0,0,1,0,0,0,0,0)$, and so on.

Note that at this point, any subcube of dimensionality d from the database corresponds to a subset of d nodes from B , and has a probability of $(m/|DB|)^d$.

We can then determine if G has a maximal biclique of size k , by performing the following simple loop:

Fig. 7. Basic algorithm to decide a maximal biclique.

For $i = 1$ to $|A|$ do:

see if G has a biclique V_1, V_2 where $|V_1| \times |V_2| = k$ to perform this check we:

construct a database as described above

see if DB has a subcube in the “cone” above the coordinates $(|V_2|, |V_1|)$ using f_{Rat}

The process is illustrated above in Fig. 7, for a bipartite graph where $|A| = |B| = 100$. We iterate along the points on the curve $|V_1| \times |V_2| = k$, checking the cone above each point. If any subcube within some cone exists in the database, there was an acceptable biclique in the original graph.

To check for subcubes occupying the cone above the coordinate (V_1, V_2) , we begin by adding p^2 additional tuples to the database, where for the i th tuple t_i that is added, $t_i[i \bmod p] = 0$ and $t_i[j] = 1$ for $j \neq i \bmod p$. This guarantees that if there was a biclique of size $|V_1|$ by $|V_2|$ in the original graph,

there is now a $|V_2|$ -dimensional subcube with $|V_1| + (p^2 - p|V_2|)$ tuples contained in the database. Next, we add a tuples to the database where for each tuple t that is added, $t[j] = 1$ for all j (we describe how to choose a below). Now, if there was a biclique of size $|V_1|$ by $|V_2|$ in the original graph, there is a $|V_2|$ -dimensional subcube with $|V_1| + \beta$ tuples contained in the database, where $\beta = (p^2 - p|V_2| + a)$.

Let $|DB|$ denote the size of the resulting database, and let g be the fraction of tuples in each dimension with positive (“yes”) attribute values. At this point, if we perform a search based upon f_{Rat} in the resulting database at a level L , we will accept the database if there were any bicliques of size $|V_1|$ by $|V_2|$ in the original graph, where:

$$|V_1| \geq g^{|V_2|} |DB|L - \beta,$$

or equivalently

$$|V_1| \geq g^{|V_2|} |DB|L - p^2 + p|V_2| - a.$$

Let x denote the target value of $|V_2|$ that we are searching for, and let y denote the target value of $|V_1|$ that we search for in this iteration of the loop. We must then demonstrate how to force the above inequality to correspond to the acceptance of only the cone above some arbitrary point $(x, k/x)$ on the curve $xy = k$, as shown in Fig. 7. To do this, we will show that it is possible to choose appropriate values for g , L , and a so as to guarantee the following:

- (1) The base of the resulting cone will intersect the curve exactly at the point $(x, k/x)$.
- (2) The slope of the lower boundary of the cone will be 0 at the point $(x, k/x)$.
- (3) The slope of the cone’s boundary will be greater than 0 to the right of $(x, k/x)$.
- (4) The cone does not intersect the line $xy = k$ to the left of the point $(x, k/x)$.

To guarantee (1), we must have

$$g^x |DB|L - p^2 + px - a = \frac{k}{x}. \quad (\text{A.1})$$

To guarantee (2), we must have

$$\frac{\partial}{\partial |V_2|} g^{|V_2|} |DB|L - p^2 + p|V_2| - a$$

evaluated at $|V_2| = x$ is 0, or equivalently,

$$g^x \ln(g) |DB|L + p = 0. \quad (\text{A.2})$$

To guarantee (3), we need only show that (2) holds, since the slope of the cone is always increasing; that is:

$$\begin{aligned} \frac{\partial^2}{\partial |V_2|^2} g^{|V_2|} |DB|L - p^2 + p|V_2| - a \\ = g^{|V_2|} \ln^2(g) |DB|L, \end{aligned}$$

which is greater than 0 for any $|V_2|$ larger than 0.

Finally, we will guarantee (4) by requiring

$$g^{x-1} \ln(g) |DB|L + p = -3. \quad (\text{A.3})$$

Note that (A.3) will also guarantee that the slope at $x - z$ is always steeper than $-3z$, and so for sufficiently large x (recall that we are concerned only with $x \geq \frac{3}{4}p$), it is possible to show that the left side of the cone will not intersect the line $|V_1||V_2| = k$ (we omit a proof of this for brevity).

Given the constraints (A.1)–(A.3), we can now solve for appropriate values of g , L , and a . We begin by solving for g . From (A.3), we know that:

$$g^x \ln(g) |DB|L = (-3 - p)g.$$

Substituting back into (A.2), we have our target value for g

$$g = \frac{p}{3 + p}. \quad (\text{A.4})$$

Note that our construction thus far guarantees that the value of g is nearly 1: it is at least $p(p - 1)/(p^2 + 1)$. Since this is always larger than the target value for g given in (A.4) above, we can achieve the target for g by simply adding an appropriate number of *NULL* tuples to the database, which lowers g as is needed. Once we have g , we can then compute L . From (A.2), we have

$$L = -\frac{p}{g^x \ln g |DB|}.$$

Next, we can use L and g to compute a using (A.1).

$$\begin{aligned} a &= -\frac{k}{x} + g^x |DB|L - p^2 + xp \\ &= -\frac{k}{x} - \frac{p}{\ln g} - p^2 + xp. \end{aligned}$$

The final question we must answer is: is a a positive number? Since a represents the number of tuples with all “yes” values that we must add to the database to check for the cone, a must be positive for the reduction to be possible. Recall that since we are reducing 1/2-VERT-3/4-EDGE-CLIQUE to 3/4-MAX-EDGE-BICLIQUE, we can assume that $x \geq \frac{3}{4}p$. We also know that $g = p/(3+p)$ and so $p/(\ln g) = p(p+\varepsilon)/3$. Since we also know that $k < p^2$, a is indeed positive.

Thus, by adding a “yes” tuples to the database and by forcing g to be $p/(1+p)$ by adding additional *NULL* tuples to the database, we can search for the cone above the point $(x, k/x)$ by searching based on f_{Rat} with $L = -p/(g^x \ln g |DB|)$. We then have a polynomial time reduction of 3/4-MAX-EDGE-BICLIQUE to search based upon f_{Rat} and Lemma A.2 is proven. Theorem 3.1 then follows directly from Lemma A.1 and Lemma A.2. $\square$

Proof of Theorem 3.2. A d_{max} -dimensional α -subcube can have an expected support of at most $\alpha^{d_{max}}$. Thus, an occupied subcube that is limited in d_{max} dimensions has a $Obs(\varsigma)/P(\varsigma)$ value that is very high: at least $(1/\alpha)^{d_{max}}$. Imagine that there existed some subcube ς_{all} of dimensionality d_{all} containing *all* of the tuples in the database. Could it be accepted when a subcube limited in d_{max} dimensions is not? Only if

$$\left(\frac{1}{\alpha}\right)^{d_{max}} < n \left(\frac{1}{\alpha}\right)^{d_{all}}.$$

Simplifying,

$$\left(\frac{1}{\alpha}\right)^{d_{max}-d_{all}} < n.$$

This is only possible if $\log_{1/\alpha} n > d_{max} - d_{all}$, or equivalently, ς_{all} has removed limitations in at most $\log_{1/\alpha} n$ dimensions more than our original

subcube. This means that no subcube with restrictions on less than $d_{max} - \log_{1/\alpha} n$ dimensions can have a greater $Obs(\varsigma)/P(\varsigma)$ value than an occupied cell limited in d_{max} dimensions. Thus, a simple algorithm to find the most interesting subcube based upon f_{Rat} would be to check all of the possible ways to remove less than $\log_{1/\alpha} n$ dimensions from the subcube corresponding exactly to each database transaction. If one of the subcubes checked has a high enough ratio, report it back. There are $O(n(d_{max} \text{ choose } (\log_{1/\alpha} n - 1)))$ such subcubes; each requires n time to check. Simplifying, this brute-force check may be done in $O(n^2 \times d_{max}^{-\log_{1/\alpha} n})$ time, and the theorem is proven. $\square$

Proof of Observation 4.1. The greatest possible expected support for an α -subcube limited is $(\alpha)^{D(\varsigma)}$. An occupied α -subcube contains at least one tuple. Thus, any occupied α -subcube ς has $Obs(\varsigma) \geq f_{Rat}(n, P(\varsigma), L)$ for $L < (1/n)(1/\alpha)^{D(\alpha)}$. $\square$

Proof of Theorem 4.2. Assume that we have a subcube ς where $D(\varsigma) = i$. That is, ς is limited in i dimensions. Furthermore, say we have two subcubes ς_0 and ς_1 , where $\varsigma_0 \cup \varsigma_1 = \varsigma$. In other words, both ς_0 and ς_1 make up exactly one-half of ς , so each is limited in one dimension that ς is not limited in (thus $D(\varsigma_0) = D(\varsigma_1) = i + 1$). Specifically, ς_1 specifies a ‘1’ value for that dimension, and ς_0 specifies a ‘0’ for that dimension. Then we know that $Obs(\varsigma_1) + Obs(\varsigma_0) = Obs(\varsigma)$. Let Pr_1 denote the probability that an arbitrary database tuple has a ‘1’ for that dimension or attribute. Then, since ς_0 and ς_1 make up ς , we also know that $P(\varsigma_1)/Pr_1 = P(\varsigma_0)/(1 - Pr_1) = P(\varsigma)$.

Given all of this, the proof will be by contradiction. Assume that Theorem 4.2 does not hold. Then, it must be possible for ς to be considered interesting by f_{Rat} at a greater value of L than either ς_0 or ς_1 . In other words, both of the following hold:

$$\begin{aligned} \frac{Obs(\varsigma_1) + Obs(\varsigma_0)}{P(\varsigma)} &> \frac{Obs(\varsigma_1)}{P(\varsigma_1)}, \\ \frac{Obs(\varsigma_1) + Obs(\varsigma_0)}{P(\varsigma)} &> \frac{Obs(\varsigma_0)}{P(\varsigma_0)}. \end{aligned}$$

Equivalently,

$$Obs(\varsigma_1) + Obs(\varsigma_0) > \frac{Obs(\varsigma_1)}{Pr_1},$$

$$Obs(\varsigma_1) + Obs(\varsigma_0) > \frac{Obs(\varsigma_0)}{1 - Pr_1}.$$

By algebraic manipulation of the second relationship,

$$Obs(\varsigma_1) + Obs(\varsigma_0) - Pr_1 Obs(\varsigma_1) - Pr_1 Obs(\varsigma_0) > Obs(\varsigma_0)$$

$$\Rightarrow -Pr_1 Obs(\varsigma_1) - Pr_1 Obs(\varsigma_0) > -Obs(\varsigma_1)$$

$$\Rightarrow Obs(\varsigma_1) + Obs(\varsigma_0) < \frac{Obs(\varsigma_1)}{Pr_1}.$$

However, this contradicts the first relationship. Hence, ς cannot be considered more interesting than *both* ς_0 and ς_1 , and the theorem is proven. $\square$

Proof of Theorem 4.3. The proof uses a reduction very similar to the proof of Theorem 3.1, though somewhat less convoluted. To decide the maximum edge biclique problem using $f_{SupAndRat}$, we execute the following algorithm:

For $i = 1$ to $|A|$ do:

see if G has a biclique V_1, V_2 where $V_1 \subset A$, $V_2 \subset B$ and $|V_1| = i, |V_2| = k/i$

to perform this check we:

construct a database as described in the proof of Theorem 1

add $|DB|^2 - |DB|$ *NULL* tuples (tuples of all 0's) to the database

check for a subcube having $Obs(\varsigma) \geq$

$$f_{RatAndSup}\left(n, P(\varsigma), \left(k/i, \frac{k}{i} \left(\frac{|DB|^2}{m}\right)^i\right)\right)$$

if the database has such a subcube, accept the graph.

Effectively, this reduction looks for maximal bicliques using a series of nearly right angles, as opposed to the cones of Theorem 1. This is shown above in Fig. 8. The vertical lines correspond to subcubes $Obs(\varsigma)/P(\varsigma) \geq L$. As we reduced the projected support so radically using the *NULL* tuples, these contours have a very steep negative slope and will not intersect the line $|V_1||V_2| = k$ until the point $|V_1| = i, |V_2| = k/i$. At this point, the minimum support is bounded by the horizontal line corresponding to the minimum support threshold $|V_1| = i$.

To complete the proof, we turn to the question: will the possible acceptance of negative or '0' values (and not only positive '1' values) affect the reduction? This is important to determine, because if we accept an unrestricted subcube ς' requiring '0' values for certain attributes, where ς' is not contained in some restricted subcube ς that is itself accepted, then the reduction is incorrect. The incorrectness results from the fact the negative values in ς' correspond to the absence of edges in our original graph, and should not cause ς' to be accepted.

To show that this cannot happen, we assume that we have a restricted subcube ς that is not

Fig. 8. Basic algorithm to decide a maximal biclique using $f_{RatAndSup}$ (left) and $f_{RatAndMinP}$.

considered interesting by the reduction (and hence does not correspond to a maximal biclique in the original graph). Then, we add an arbitrary number of restrictions to ς , where each additional restriction requires the absence of the specified attribute value (i.e., a ‘0’). Let ς' denote the resulting unrestricted subcube. We assert that $P(\varsigma') \geq P(\varsigma)/e$ where $e = 2.71828\dots$ (this is true since $((|DB|^2 - |DB|)/|DB|^2)^{|DB|} \approx 1/e$). Clearly, $Obs(\varsigma') \leq Obs(\varsigma)$ (because we have made ς' more restrictive than ς). Thus, we observe that $P(\varsigma')Obs(\varsigma') \geq P(\varsigma)Obs(\varsigma)/e$.

The question is, given this observation, could ς' now be accepted by our reduction, when obviously it does not correspond to a maximal biclique in the original graph? The answer is no. If ς was not accepted in the first place because it did not meet the minimal support requirement, then because $Obs(\varsigma') \leq Obs(\varsigma)$, ς' will not be accepted either. If ς *did* meet the minimal support requirement, then we know that it was not accepted because it was limited in too few dimensions (it was limited in at most $i - 1$ dimensions). Since the greatest possible support for ς is $Obs(\varsigma) = m$, ς must be rejected at any correlation level $L \geq m/n(|DB|^2/m)^{i-1}$. Since we know that $P(\varsigma')Obs(\varsigma') \geq P(\varsigma)Obs(\varsigma)/e$, ς' must be *still* rejected at any correlation level $L \geq e \times (m/n)(|DB|^2/m)^{i-1}$. Since this is necessarily less than the level $L = (k/ni)(|DB|^2/m)^i$, that is checked for by the reduction, ς' could not have been accepted if ς was not. Thus, the reduction is complete, and the problem of finding any subcube (restricted or not) where $Obs(\varsigma) \geq f_{RatAndSup}(n, P(\varsigma), (s, L))$ is NP-hard. $\square$

Proof of Corollary 4.4. This follows immediately from the above proof, since as the projected support of each dimension is decreased by some arbitrary factor, we can still decide the maximum biclique problem using $f_{RatAndSup}$ by increasing the correlation level that we search for accordingly. $\square$

Proof of Theorem 4.5. For this proof, we use virtually the reduction used in the proof of Theorem 3.1. Recall that in that proof, we showed how to check for the existence of a subcube in the cone above the coordinate $(x, k/x)$ by forcing a g

value of $g = p/(3 + p)$. If we remove the restriction that our search be only over restricted subcubes, we can still use the reduction if we perform a search using $f_{RatAndMinP}$ exactly as we did with f_{Rat} as long as we set a minimum subcube probability of $(p/(3 + p))^x$. As in the proof of intractability of search based upon $f_{RatAndSup}$, we must then complete our proof by answering the question: will the possible acceptance of negative or ‘0’ values (and not only positive ‘1’ values) affect the reduction? Again, as in that proof, the answer is no. Any subcube which specifies any negative values will have an associated probability of at most $1 - p/(3 + p)$. For large p , this is a small value, and is clearly less than the minimum subcube probability of $(p/(3 + p))^x$ that is checked for, since $(p/(3 + p))^x \approx 1/e^3$ as $x \rightarrow 1 + p(1/e^3)$ is around 0.049787. Thus, the reduction is complete, and the problem of finding any subcube (restricted or not) where $Obs(\varsigma) \geq f_{RatAndMinP}(n, P(\varsigma), (s, MinP))$ is NP-hard. $\square$

Proof of Theorem 5.1. The proof is based upon a construction very similar to the proof of Theorem 3.1, which showed a polynomial time reduction of 3/4-MAX-EDGE-BICLIQUE to search based upon f_{Rat} . Virtually the same reduction is used for $f_{ChiSquare}$, so we give only a brief argument of its correctness here.

To use $f_{ChiSquare}$ to detect biclique in the same sort of cone that we tested for in the proof of the hardness of search based upon f_{Rat} , we first convert the graph into a database and then normalize the projected support of each dimension, just as in the construction used in that proof. We next add the p^2 additional tuples, just as in the previous reduction. After all of this, $f_{ChiSquare}$ will accept the resulting database if there was a biclique of size $|V_1| \times |V_2|$ for which

$$|V_1| \geq \sqrt{L} \sqrt{g^{|V_2|} |DB|} + g^{|V_2|} |DB| - p^2 + p|V_2| + a.$$

As in the proof of Theorem 3.1, we again wish to show that it is possible to choose g , L , and a so as to construct a cone above the point $(x, k/x)$, so that the slope of the cone boundary is exactly 0 at that point. Equivalently, we need to show that it is

possible to choose g' , L' and a to guarantee the acceptance of subcubes in the cone

$$|V_1| \geq L'|DB|g'^{|V_2|} + g'^{2|V_2|}|DB| - p^2 + p|V_2| + a,$$

where $g' = \sqrt{g}$ and $L' = \sqrt{L/|DB|}$. In a way very similar to the proof of Theorem 3.1, we can show that this is done for the cone above the point $(x, k/x)$ if we choose L' so that

$$L' = \frac{-p - 2|DB|g'^{2x} \ln g'}{|DB|g'^x \ln g'}.$$

In addition, it is possible to show that if we choose $g' = p/(3+p)$ (as in the proof of Theorem 3.1), we will force the slope of the cone to have a slope of *at most* -3 at $|V_2| = x - 1$. This contrasts with the proof of Theorem 3.1, where the same g forces a slope of *exactly* -3 . However, this additional steepness is not a problem, since our goal in choosing g is to guarantee that the cone does not intersect the line $|V_1| = k/|V_2|$ at any point to the left of $|V_2| = x - 1$.

The last thing that we need to show is that this new cone can still be made to intersect the curve $|V_1| = k/|V_2|$ at *exactly* the point $(x, k/x)$. In other words, we need to show that the value of a is positive, where

$$a = L'|DB|g'^x + g'^{2x}|DB| - p^2 + px - \frac{k}{x}.$$

To show that a is positive, we plug the formula for L' into the above, and have

$$a = \frac{-p - 2|DB|g'^{2x} \ln g'}{\ln g} + g'^{2x}|DB| - p^2 + px - \frac{k}{x}.$$

Simplifying, we have

$$a = \frac{-p - \varepsilon_1}{\ln g} + \varepsilon_2 - p^2 + px - \frac{k}{x}$$

for some $\varepsilon_1, \varepsilon_2 > 0$. This expression is almost identical to the one from the end of the proof of Theorem 3.1. Then, by the same argument given at the end of the proof of Theorem 3.1, a does indeed have a positive value, and the cone can be made to intersect $|V_1| = k/|V_2|$ at exactly the point $(x, k/x)$. Thus, we can construct a cone with $f_{ChiSquare}$ just as we could with f_{Rat} , and the reduction is complete. $\square$

Proof of Theorem 5.2. Our proof will be analogous to the proof of Theorem 3.2. Assume that our most restricted, occupied subcube is limited in d_{max} dimensions. This α -subcube can have an expected support of at most $\alpha^{d_{max}}$. Thus, this subcube is accepted by $f_{ChiSquare}$ at an L level of

$$\frac{(nP(\zeta) - Obs(\zeta))^2}{P(\zeta)} = \frac{(n\alpha^{d_{max}} - 1)^2}{\alpha^{d_{max}}}.$$

Imagine that there existed some subcube ζ_{all} of dimensionality d_{all} containing *all* of the tuples in the database. Could it be accepted when a simple occupied cell is not? Only if

$$\begin{aligned} \frac{(n\alpha^{d_{max}} - 1)^2}{\alpha^{d_{max}}} &< \frac{(n\alpha^{d_{all}} - n)^2}{\alpha^{d_{all}}} \\ \Rightarrow abs(n\alpha^{d_{max}} - 1)\alpha^{(d_{all}-d_{max})/2} &< abs(n\alpha^{d_{all}} - n) \\ \Rightarrow abs(n\alpha^{(d_{all}+d_{max})/2} - \alpha^{(d_{all}-d_{max})/2}) &< abs(n\alpha^{d_{all}} - n). \end{aligned}$$

Assume that $n\alpha^{d_{max}} < 1$. If this is not the case, then the theorem holds because d_{max} is then $O(\log_{1/\alpha} n)$, and so we can check all of the occupied subcubes (which must have d_{max} or less restrictions) in pseudopolynomial time with respect to database size. If $n\alpha^{d_{max}} < 1$, we have

$$\begin{aligned} \Rightarrow \alpha^{(d_{all}-d_{max})/2} &< n \\ \Rightarrow \left(\frac{1}{\alpha}\right)^{(d_{max}-d_{all})/2} &< n. \end{aligned}$$

This is only possible if $\log_{1/\alpha} n > (d_{max} - d_{all})/2$; or equivalently, ζ_{all} has removed limitations in at most $2 \log_{1/\alpha} n$ dimensions. Then, following an argument identical to that for f_{Rat} , $f_{ChiSquare}$ can be used to search for α -subcubes in quasipolynomial time for fixed α , and the theorem is proven. $\square$

Proof of Theorem 5.3. Again, the proof is via a reduction to the maximal biclique problem. This reduction is very similar to the others presented, except that likelihood can be used to check for a maximal biclique almost directly. To argue that this can be done, we first expand out the formula for likelihood given in Section 5. Let $i = Obs(\zeta)$. Expanding the formula, at a correlation level of L ,

we will accept all subcubes where

$$L < -i \log P(\zeta) + (i - n) \log(1 - P(\zeta)) \\ + i \log\left(\frac{i}{n}\right) + (n - i) \log\left(1 - \frac{i}{n}\right).$$

In order to detect a biclique in a graph, we perform the same transformation into a database as was done in the proof of Theorem 3.1, and then we normalize the projected support of each dimension so that it is $m/|DB|$. Assume that the size of the resulting database is n . The next step in the reduction is to replicate each of the database dimensions a large number of times (say, n^2 times). At this point, a search at a correlation level of L in our database will accept subcubes corresponding to bicliques where

$$L < -n^2|V_1||V_2| \log \frac{m}{n} + (i - n) \log\left(1 - \frac{m^{|V_2|}}{n}\right) \\ + i \log\left(\frac{|V_1|}{n}\right) + (n - |V_1|) \log\left(1 - \frac{|V_1|}{n}\right).$$

The first term will dominate the value of L , so that we will effectively accept bicliques where

$$L < -n^2|V_1||V_2| \log \frac{m}{n}.$$

By substituting an appropriate value for L , we can then easily accept bicliques where

$$|V_1||V_2| > k$$

for some constant k . Thus, we solve precisely the maximal biclique problem, and search based upon f_{Lik} is NP-hard. $\square$

Proof of Theorem 6.1. We begin by asserting that after normalization (as was done in the previous proofs), any instance of the maximum biclique problem can be solved directly by search based upon f_{Quad} .

The question then becomes, how hard is the maximum edge biclique problem to approximate? We begin by showing that the problem is MAX SNP-hard. $\square$

Lemma A.3. *The maximum edge biclique problem is MAX SNP-hard.*

Proof. First, we assert that the corresponding edge deletion problem (the problem of deleting the fewest number of edges from a bipartite graph so as to leave a biclique) is MAX SNP-hard. This follows from the work on phrasing certain minimization problems as the so-called *IP2 minimization problems* [30,31]. According to this work, all NP-hard IP2 problems are as hard to approximate as the vertex cover problem, and hence are MAX SNP-hard. The edge deletion problem can be phrased as an IP2 optimization problem (a version of the edge deletion problem was phrased as an IP2 problem previously [30]) and it is clearly NP-hard, since a solution to the deletion problem would in turn solve the maximal edge biclique problem. Thus, if $\bar{\varphi}(G)$ denotes the optimal number of edges in the solution to the minimal edge deletion problem on G , then it is NP-hard to find an approximate solution edge deletion problem that removes $(1 + \beta)\bar{\varphi}(G)$ edges for some number $\beta > 0$.

Now, consider the maximal edge biclique problem. Let $\varphi(G)$ denote the number of edges in an optimal solution to this problem. For a graph G , it should be clear that $\varphi(G) + \bar{\varphi}(G) = |E|$, where $|E|$ is the number of edges in G . Thus, an exact solution to $\bar{\varphi}(G)$ can be used to provide an exact solution to $\varphi(G)$, and vice versa. In the remainder of the proof, we will show a similar relationship: how an approximate solution to $\varphi(G)$ (we will refer to this approximate answer as $\varphi(G)'$) can be used to approximate a solution to $\bar{\varphi}(G)$ (referred to as $\bar{\varphi}(G)'$). Specifically, we will show that if we can find some approximation scheme for $\varphi(G)$ such that we can guarantee that for every fixed value of $\varepsilon > 0$, $\varphi(G)'$ is greater than $\varphi(G)(1 - \varepsilon)$, but less than $\varphi(G)$, then we can guarantee that for every fixed value of $\beta > 0$ that $\bar{\varphi}(G)'$ is greater than $\bar{\varphi}(G)$, but is less than $(1 + \beta)\bar{\varphi}(G)$. This leads to a contradiction, since approximation of $\bar{\varphi}(G)$ is MAX SNP-hard, and so approximation of $\varphi(G)$ is MAX SNP-hard as well.

In order to use $\varphi(G)'$ to approximate $\bar{\varphi}(G)$ accurately, we must consider two separate cases.

Case 1. If $\varphi(G)$ is not very large relative to $|E|$. Pick any value $\gamma > 0$. As long as $\varphi(G)$ is less than $(1 - \gamma)|E|$, the approximability of $\varphi(G)$ implies

approximability of $\bar{\varphi}(G)$ to within β for any $\beta > 0$. How? If $\varphi(G)$ can be approximated to within a factor of ε for every fixed value of $\varepsilon > 0$, then we know that we can approximate $\varphi(G)$ to within $\varepsilon = \beta \times \gamma$. Then, we let $\bar{\varphi}(G)' = |E| - \varphi(G)'$. Since $\varphi(G)'$ is accurate to within $\beta\gamma$, $\bar{\varphi}(G)'$ will be accurate to a factor of less than β , and we have an accurate approximation of $\bar{\varphi}(G)$.

Case 2. If $\varphi(G)$ is very large relative to $|E|$. The problem with the above approximation is that as $\varphi(G)$ approaches $|E|$ for unbounded $|E|$, the error in $\varphi(G)'$ must tend to 0 in order to compute $\bar{\varphi}(G)'$ accurately (even if computing $\varphi(G)'$ is not MAX SNP-hard, it does not imply that we can compute $\varphi(G)'$ with arbitrary accuracy). However, the algorithm implied by Theorem 6.2 *can* be used to approximate $\bar{\varphi}(G)$ with arbitrary accuracy as $\varphi(G)$ approaches $|E|$. For brevity, we will not describe the mathematics in detail here. However, the idea is that since Theorem 6.2 says that we can detect with no inaccuracy the existence of a subcube (and hence, a biclique) above the straight lines of Fig. 5, and because those straight lines are an arbitrarily accurate approximation for the quadratic contours of $\varphi(G)$ as $\varphi(G)$ tends to $|E|$, we can in fact approximate $\varphi(G)$ with arbitrary accuracy as $\varphi(G)$ tends to $|E|$. Then, we can use this approximation to compute $\bar{\varphi}(G)'$ with arbitrary accuracy for small $\bar{\varphi}(G)'$ relative to $|E|$.

Thus, we have described an accurate approximation of $\bar{\varphi}(G)'$. However, computation of $\bar{\varphi}(G)'$ is MAX SNP-hard, and so computation of $\varphi(G)'$ must be MAX SNP-hard as well. We now turn to proving the main theorem. $\square$

Lemma A.4. *If the maximum edge biclique problem is MAX SNP-hard, then it cannot be approximated to within a factor of n^β for some $\beta > 0$, unless $NP \subseteq \tilde{P}$.*

Proof. The proof is analogous to the relatively simple proof of given by Arora and Lund for the difficulty of the approximation of the clique problem [32].

Since the maximum edge biclique problem is MAX SNP-hard, it follows that there is some reduction from SAT such that if the resulting

bipartite graph has a biclique of size $k(1 - \varepsilon)$, then the original formula is not satisfiable, and if the graph has a biclique of size k , the formula is satisfiable. Thus, it is NP-hard to determine which is the case.

Let G be this bipartite graph, and assume that it has n vertices on each side. Take this graph, and construct a $(n, \log n, 1/n^3)$ booster for the left side of the graph, and then construct a $(n, \log n, 1/n^3)$ booster for the right side of the graph (see Arora and Lund [32] for a description of the concept of a *booster*). Such a booster can be constructed in polynomial time [32]. Each of the sets in the left booster will form a node in the left side of a new bipartite graph, G' . Each of the sets in the right booster will form a node on the right side of G' . Two nodes in G' are connected by an edge if and only if all of the nodes in the two corresponding sets of nodes in G are connected to one another.

Assume that the biclique in G with the largest number of edges had L nodes on the left side, and R nodes on the right side. Using Theorem 10.4 from Arora and Lund, G' now has a biclique having between $\zeta(L/n - 1/n^3)^{\log n}(R/n - 1/n^3)^{\log n}$ and $\zeta(L/n + 1/n^3)^{\log n}(R/n + 1/n^3)^{\log n}$ edges, for some number ζ . Simplifying these two expressions:

$$\begin{aligned} \Rightarrow \min &= \zeta \left[\left(\frac{L}{n} - \frac{1}{n^3} \right) \left(\frac{R}{n} - \frac{1}{n^3} \right) \right]^{\log n}, \\ \max &= \zeta \left[\left(\frac{L}{n} + \frac{1}{n^3} \right) \left(\frac{R}{n} + \frac{1}{n^3} \right) \right]^{\log n} \\ \Rightarrow \min &= \zeta \left[\frac{LR}{n^2} - \frac{L}{n^4} - \frac{R}{n^4} + \frac{1}{n^6} \right]^{\log n}, \\ \max &= \zeta \left[\frac{LR}{n^2} + \frac{L}{n^4} + \frac{R}{n^4} + \frac{1}{n^6} \right]^{\log n}. \end{aligned}$$

Say the largest biclique in G was of size $k(1 - \varepsilon)$. Then the largest biclique in G' now has at most $\zeta(k(1 - \varepsilon)/n^2 + L/n^4 + R/n^4 + 1/n^6)^{\log n} < \zeta(k(1 - \varepsilon)/n^2 + 3/n^3)^{\log n}$ edges. Say instead the largest biclique in G was size k . Then the smallest biclique in G' now has at least $\zeta(k/n^2 - L/n^4 - R/n^4 + 1/n^6)^{\log n} > \zeta(k/n^2 - 2/n^3)^{\log n}$ edges. Furthermore, it is NP-hard to decide between these two cases (if we could, then we could have decided the satisfiability of our original formula in polynomial time). Since ζ is polynomial in n (see Arora and

Lund [32]), it is NP-hard to decide whether G' has at most $\text{poly}(n)n^{\log(k(1-\varepsilon)/n^2+3/n^3)}$ edges, or at least $\text{poly}(n)n^{\log(k/n^2-2/n^3)}$ edges. Now, loop through all of the vertices in G' , and clone each vertex v in G' , $n^{\log n}$ times, connecting each new clone of v to all of the same vertices that v is connected to. This can be done in quasi-polynomial time. The gap that we cannot decide in G' is now from $\text{poly}(n)n^{\log n^2+\log(k(1-\varepsilon)/n^2+3/n^3)}$ to $\text{poly}(n)n^{\log n^2+\log^2 n \log(k/n^2-2/n^3)}$. Simplifying, it is from $\text{poly}(n)n^{\log(k(1-\varepsilon)+3/n)}$ to $\text{poly}(n)n^{\log(k-2/n)}$. Since, without loss of generality, k can be assumed to be larger than any arbitrary constant, the difference between these two values is at least n^β for some $\beta > 0$, and the theorem is proven.

It is worthwhile to note that the reason that we resorted to a quasipolynomial transformation to prove the hardness of the problem was the roundabout way that we showed that the problem is MAX SNP-hard in Lemma A.4. If we had some sort of proof that it was still MAX SNP-hard to find a biclique of size k when k is larger than εn^2 for some fixed constant ε (probably a reasonable assumption), then we could prove Theorem 6.1 subject only to the requirement that $P \neq NP$. However, it is not obvious how this can be done. $\square$

Proof of Theorem 6.2. This proof stems from the fact that it is possible to find a biclique in a bipartite graph where $|V_1| + |V_2| = k$ in polynomial time, via a number of different algorithms (such as bipartite matching or a max flow/min cut algorithm; see Johnson for more on this [23]). Thus, to determine if a database has a support-biased subcube, we can do the following. First, we determine the slope a/b (support to dimensionality) of the contour line for our desired level of correlation (as in the lines from Fig. 5). We then normalize this slope to 1 to 1 by replicating all of the rows of the database (which decreases the numerator of the slope) and/or replicating all of the columns (which decreases the denominator). Next, we essentially reverse the database construction of the proof of Theorem 5.3 in order to convert the database into a graph. Finally, solving for a biclique of appropriate size will find a support-biased subcube in polynomial time. $\square$

References

- [1] K. Fujisawa, Y. Hamuro, N. Katoh, T. Tokuyama, K. Yada, Approximation of optimal two-dimensional association rules for categorical attributes using semidefinite programming, Discovery Science, Second International Conference (DS '99), Tokyo, Japan, December 1999, pp. 148–159.
- [2] K.S. Beyer, R. Ramakrishnan, Bottom-up computation of sparse and iceberg CUBEs, SIGMOD 1999, Proceedings of the ACM SIGMOD International Conference on Management of Data, Philadelphia, PA, USA, June 1–3, 1999, pp. 359–370.
- [3] J. Gray, A. Bosworth, A. Layman, H. Pirahesh, Data cube: a relational aggregation operator generalizing group-by, cross-tab, and sub-total, Proceedings of the 12th International Conference on Data Engineering, IEEE Computer Society, New Orleans, Louisiana, February 26–March 1, 1996, pp. 152–159.
- [4] R. Agrawal, T. Imielinski, A.N. Swami, Mining association rules between sets of items in large databases, Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, DC, May 26–28, 1993, pp. 207–216.
- [5] R. Agrawal, R. Srikant, Fast algorithms for mining association rules in large databases, Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94), Santiago de Chile, Chile, September 12–15, 1994, pp. 487–499.
- [6] J. Han, J. Pei, G. Dong, K. Wang, Efficient computation of iceberg cubes with complex measures, SIGMOD Conference 2001, Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data, Santa Barbara, CA, USA, May 21–24, 2001.
- [7] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J.D. Ullman, C. Yang, Finding interesting associations without support pruning, Proceedings of the 16th International Conference on Data Engineering, San Diego, CA, USA, 28 February–3 March, 2000, pp. 489–499.
- [8] S. Brin, R. Motwani, C. Silverstein, Beyond market baskets: generalizing association rules to correlations, Proceedings of the ACM SIGMOD International Conference on Management of Data, Tucson, AZ, USA, May 13–15, 1997, pp. 265–276.
- [9] Citeseer: Scientific Literature Digital Library; <http://citeseer.nj.nec.com/cs/>
- [10] DBLP Most Referenced; <http://www.acm.org/sigmod/dblp/db/about/top.html>.
- [11] S. Chakrabarti, S. Sarawagi, B. Dom, Mining surprising patterns using temporal description length, Proceedings of the 24th International Conference on Very Large Data Bases, New York City, NY, USA, August 24–27, 1998, pp. 606–617.
- [12] B. Padmanabhan, A. Tuzhilin, Small is beautiful: discovering the minimal set of unexpected patterns, Proceedings of the Sixth ACM SIGKDD International Conference

- on Knowledge Discovery and Data Mining, Boston, MA, USA, August 20–23, 2000, pp. 54–63.
- [13] B. Padmanabhan, A. Tuzhilin, A belief-driven method for discovering unexpected patterns, *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, New York City, NY, USA, August 27–31, 1998, pp. 94–100.
 - [14] S. Sarawagi, Explaining differences in multidimensional aggregates, *Proceedings of the 25th International Conference on Very Large Data Bases*, Edinburgh, Scotland, UK, September 7–10, 1999, pp. 42–53.
 - [15] R.J. Bayardo Jr., Efficiently mining long patterns from databases, *SIGMOD 1998, Proceedings of the ACM SIGMOD International Conference on Management of Data*, Seattle, Washington, USA, June 2–4 1998, pp. 85–93.
 - [16] J. Han, J. Pei, Y. Yin, Mining frequent patterns without candidate generation, *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, Dallas, TX, USA, May 16–18, 2000, pp. 1–12.
 - [17] R.J. Bayardo Jr., R. Agrawal, Mining the most interesting rules, *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 1999)*, San Diego, CA, USA, August 15–18, 1999, pp. 145–154.
 - [18] G.I. Webb, Efficient search for association rules, *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2000)*, Boston, MA, USA, August 20–23, 2000, pp. 99–107.
 - [19] S. Morishita, J. Sese, Traversing itemset lattice with statistical metric pruning, *Proceedings of the 19th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS 2000)*, Dallas, TX, USA, May 15–17, 2000, pp. 226–236.
 - [20] M. Reisner, Cadillac desert: the American west and its disappearing water, Penguin USA (Paper), Revised edition, June 1993.
 - [21] R. Peters, The maximum edge biclique problem is NP-complete, Tilburg University, Faculty of Economics and Business Administration Research Memorandum: RePEc:dgr:kubrem:2000789.
 - [22] D. Gunopulos, H. Mannila, S. Saluja, Discovering all most specific sentences by randomized algorithms, *Database Theory—ICDT '97, Sixth International Conference*, Delphi, Greece, January 8–10, 1997, pp. 215–229.
 - [23] D.S. Johnson, The NP-completeness column: an ongoing guide: announcements, updates, and greatest hits, *J. Algorithms* 8 (3) (1987) 438–448.
 - [24] M.J. Zaki and M. Ogihara, Theoretical foundations of association rules, *Proceedings of the Third SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, ACM, June 1998, pp. 71–78.
 - [25] S. Morishita, On classification and regression, *Discovery Science, Proceedings of the First International Conference (DS '98)*, Fukuoka, Japan, December 14–16, 1998, pp. 40–57.
 - [26] R. Rastogi, K. Shim, Mining optimized support rules for numeric attributes, *Proceedings of the 15th International Conference on Data Engineering (ICDE 1999)*, Sydney, Australia, 23–26 March 1999, pp. 206–215.
 - [27] J. Wijsen, R. Meersman, On the complexity of mining quantitative association rules, *Data Mining Knowledge Discov.* 2 (3) (1998) 263–281.
 - [28] U. Feige, Relations between average case complexity and approximation complexity, *Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC 2002)*, ACM, Montréal, Qué., Canada, May 19–21, 2002, pp. 534–543.
 - [29] T. Dunning, Accurate methods for the statistics of surprise and coincidence, *Comput. Linguistics* 19 (1) (1993) 61–74.
 - [30] D.S. Hochbaum, Approximating clique and biclique problems, *J. Algorithms* 29 (1) (1998) 174–200.
 - [31] D.S. Hochbaum, in: D.S. Hochbaum (Ed.), *Approximation Algorithms for NP-Hard Problems*, PWS Publishing Company, Boston, MA, 1997.
 - [32] S. Arora, C. Lund, Hardness of approximations, in: D.S. Hochbaum (Ed.), *Approximation Algorithms for NP-Hard Problems*, PWS Publishing Company, Boston, MA, 1997 (Chapter 6).