Discovering Reliable Dependencies from Data: Hardness and Improved Algorithms

Panagiotis Mandros, Mario Boley, Jilles Vreeken
Max Planck Institute for Informatics and Saarland University
Saarland Informatics Campus, Saarbrücken, Germany
{pmandros,mboley,jilles}@mpi-inf.mpg.de

***Abstract*—The reliable fraction of information is an attractive score for quantifying (functional) dependencies in high-dimensional data. In this paper, we systematically explore the algorithmic implications of using this measure for optimization. We show that the problem is NP-hard, which justifies the usage of worst-case exponential-time as well as heuristic search methods. We then substantially improve the practical performance for both optimization styles by deriving a novel admissible bounding function that has an unbounded potential for additional pruning over the previously proposed one. Finally, we empirically investigate the approximation ratio of the greedy algorithm and show that it produces highly competitive results in a fraction of time needed for complete branch-and-bound style search.**

***Index Terms*—knowledge discovery, approximate functional dependency, information theory, optimization, branch-and-bound**

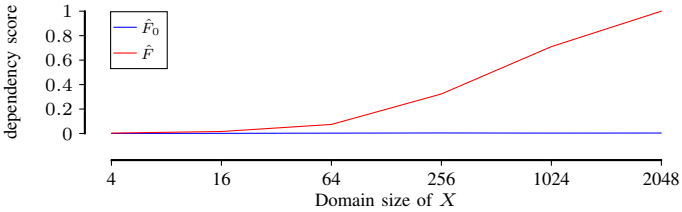


Figure 1: **Dependency-by-chance.** Estimated fraction of information for variables X of increasing domain size (4 to 2048) to independent Y (domain size 4) for fixed sample size (1000). Estimated dependency increases for naive estimator $\hat{F}$, while the corrected-for-chance estimator $\hat{F}_0$ accurately estimates population value $F(X;Y) = 0$.

I. Introduction

Given a data sample $\mathbf{D}_n = \{\mathbf{d}_1, \ldots, \mathbf{d}_n\}$ drawn from the joint distribution p of some input variables $\mathcal{I}$ and an output variable Y, it is a fundamental problem in data analysis to find variable subsets $\mathcal{X} \subseteq \mathcal{I}$ that jointly influence or (approximately) determine Y. This **functional dependency discovery** problem, i.e., to find

$$\arg\max\{Q(\mathcal{X};Y) : \mathcal{X} \subseteq \mathcal{I}\} \tag{1}$$

for some real-valued measure Q that assesses the dependence of Y on $\mathcal{X}$, is a classic topic in the database community [1, Ch. 15], but also has many other applications including feature selection [2] and knowledge discovery [3]. For instance, finding such dependencies can help identify compact sets of descriptors that capture the underlying structure and actuating mechanisms of complex scientific domains (e.g., [4], [5]).

For categoric input and output variables, the measure Q can be chosen to be the **fraction of information** [6], [7], [8] defined as

$$F(\mathcal{X};Y) = (H(Y) - H(Y \mid \mathcal{X}))/H(Y) \ ,$$

where $H(Y) = \sum_{y \in Y} p(y) \log p(y)$ denotes the **Shannon entropy**. This score represents the relative reduction of uncertainty about Y given $\mathcal{X}$. It takes on values between 0 and 1 corresponding to independence and exact functional dependency, respectively.

Estimating the score naively with empirical probabilities $\hat{p}$, however, leads to an overestimation of the actual dependence between $\mathcal{X}$ and Y, a behavior known as *dependency-by-chance* [9]. In particular, since the bias is increasing with the domain size of variables [10], it is unsuitable for dependence discovery where we have to soundly compare different variable sets of varying dimensionality and consequently of widely varying domain sizes (see Fig. 1). In some feature selection approaches (see, e.g., [11]) this problem is mitigated by only considering dependencies of individual variables or pairs. Alternatively, some algorithms from the database literature, e.g., [12], [13], neglect this issue by assuming a closed-world, i.e., the unknown data generation process p is considered equal to the empirical $\hat{p}$ [7].

Both of these approaches are infeasible in the statistical setting with arbitrary sized variable sets that we are interested in. Instead, here, the fraction of information can be corrected by subtracting its estimated expected value under the hypothesis of independence. This gives rise to the **reliable fraction of information** [14], [15] defined as

$$\hat{F}_0(\mathcal{X};Y) = \hat{F}(\mathcal{X};Y) - \hat{E}_0(\hat{F}(\mathcal{X};Y)) \ ,$$

where $\hat{E}_0(\hat{F}(\mathcal{X};Y)) = \sum_{\sigma \in S_n} \hat{F}(X;Y_\sigma)/n!$ is the expected value of $\hat{F}$ under the **permutation model** [16, p. 214], i.e., under the operation of permuting the empirical Y values with a random permutation $\sigma \in S_n$. This estimator can be computed efficiently in time $O(nk)$ for $\mathcal{X}$ with domain size k (see [17] and appendix). Moreover, the maximization problem (Eq. (1)) can be solved effectively by a simple branch-and-bound scheme: the maximally attainable $\hat{F}_0$ for supersets of some partial solution $\mathcal{X}$ can be bounded by the function $\bar{f}_{\text{mon}}(\mathcal{X}) = 1 - \hat{E}_0(\hat{F}(\mathcal{X};Y))$, which follows from

the *monotonicity* of $\hat{E}_0(\hat{F}(\cdot\,;Y))$ [14].

This, however, is a rather simplistic bounding function that leaves room for substantial improvements. Moreover, it is unclear whether one has to rely on exponential-time worst-case branch-and-bound algorithms in the first place. Finally, the option of heuristic optimization has not yet been explored.

To this end, this paper provides the following contributions:

1) We show that the problem of maximizing the reliable fraction of information is NP-hard. This justifies the usage of worst-case exponential-time algorithms as well as heuristic search methods (Sec. III).
2) Motivated by this insight, we then greatly improve the practical performance for both of these optimization styles by deriving a novel admissible bounding function $\bar{f}_{\text{spc}}(\mathcal{X})$. This function is not only tighter than the previously proposed $\bar{f}_{\text{mon}}(\mathcal{X})$ but in particular we have that the supremum of $\bar{f}_{\text{mon}}(\mathcal{X})/\bar{f}_{\text{spc}}(\mathcal{X})$—and thus the potential for additional pruning in search—is unbounded (Sec. IV).
3) Finally, we report extensive empirical results evaluating the proposed bounding function and the various algorithmic strategies. In particular, we consider the approximation ratio of the greedy algorithm and show that in fact, it produces highly competitive results in a fraction of time needed for complete branch-and-bound style search—motivating further investigation of this fact (Sec. V).

We round up with a concluding discussion (Sec. VI). Before presenting the main contributions, we recall reliable functional dependency discovery and prove some basic results (Sec. II).

II. Reliable Dependency Discovery

Let us denote by $[n]$ the set of positive integers up to n. The symbols $\log$ and $\ln$ refer to the logarithms of base 2 and e, respectively. We assume a set of discrete random variables $\mathcal{A} = \mathcal{I} \cup \{Y\}$ is given along with an empirical sample $\mathbf{D}_n = \{\mathbf{d}_1, \ldots, \mathbf{d}_n\}$ of their joint distribution. For a variable X we denote its domain, called **categories** (or distinct values), by $V(X)$ but we also write $x \in X$ instead of $x \in V(X)$ whenever clear from the context. We identify a random variable X with the **labeling** $X\colon [n] \to V(X)$ it induces on the data sample, i.e., $X(i) = \mathbf{d}_i(X)$. Moreover, for a set $\mathcal{S} = \{S_1, \ldots, S_l\}$ of labelings over $[n]$, we define the corresponding vector-valued labeling by $\mathcal{S}(i) = (S_1(i), \ldots, S_l(i))$. With $X_{\mathcal{Q}}$ for a subset $\mathcal{Q} \subseteq [n]$, we denote the map X restricted to domain $\mathcal{Q}$.

We define $c_X\colon V(X) \to \mathbb{Z}_+$ to be the **empirical counts** of X, i.e., $c_X(x) = |\{i \in [n] : X(i) = x\}|$. We further denote with $\hat{p}_X\colon V(X) \to [0,1]$, where $\hat{p}_X(x) = {}^{c_X(x)}/_{n}$, the **empirical distribution** of X. Given another random variable Z, $\hat{p}_{Z\,|\,X=x}\colon V(Z) \to [0,1]$ is the **empirical conditional distribution** of Z given $X = x$, with $\hat{p}_{Z|X=x}(z) = {}^{c_{X\cup Z}(x,z)}/_{c_X(x)}$ for $z \in Z$. However, we use $\hat{p}(x)$ and $\hat{p}(z\,|\,x)$ respectively whenever clear from the context. These empirical probabilities give rise to the **empirical conditional entropy** $\hat{H}(Y\,|\,X) = \sum_{x\in X} \hat{p}(x)\hat{H}(Y\,|\,X = x)$, the **empirical mutual information** $\hat{I}(X;Y) = \hat{H}(Y) - \hat{H}(Y\,|\,X)$, and the **empirical fraction of information** $\hat{F}(X;Y) = \hat{I}(X;Y)/\hat{H}(Y)$.

Recall that the reliable fraction of information is defined as the empirical fraction of information $\hat{F}(X;Y)$ minus its expected value under the permutation model $\hat{E}_0(\hat{F}(X;Y))$ where $\hat{E}_0(\hat{F}(X;Y)) = \sum_{\sigma\in S_n} \hat{F}(X;Y_\sigma)/n!$. Here, S_n denotes the **symmetric group** of $[n]$, i.e., the set of bijections from $[n]$ to $[n]$, and A_σ denotes the composition of a map A with the permutation $\sigma \in S_n$, i.e., $A_\sigma(\cdot) = A(\sigma(\cdot))$. We abbreviate the **correction term** $\hat{E}_0(\hat{F}(X;Y))$ as $\hat{b}_0(X,Y,n)$ and the unnormalized version as $\hat{m}_0(X,Y,n) = \hat{b}_0(X,Y,n)\hat{H}(Y)$.

A. Specializations and Labeling Homomorphisms

Since we identified sets of random variables with their corresponding sample-index-to-value map, they are subject to the following general relations of maps with common domains.

Definition 1. *Let A and B be maps defined on a common domain N. We say that A is **equivalent** to B, denoted as $A \equiv B$, if for all $i, j \in N$ it holds that $A(i) = A(j)$ if and only if $B(i) = B(j)$. We say that B is a **specialization** of A, denoted as $A \preceq B$, if for all $i, j \in N$ with $A(i) \neq A(j)$ it holds that $B(i) \neq B(j)$.*

A special case of specializations is given by the subset relation of variable sets, e.g., if $\mathcal{X} \subseteq \mathcal{X}' \subseteq \mathcal{I}$ then $\mathcal{X} \preceq \mathcal{X}'$. The specialization relation implies some important properties for empirical probabilities and information-theoretic quantities.

Proposition 1. *Given variables X, Z and Y, with $X \preceq Z$, the following statements hold:*

a) there is a projection $\pi : V(Z) \to V(X)$, s.t. for all $x \in V(X)$, it holds that $\hat{p}_X(x) = \sum_{z\in\pi^{-1}(x)} \hat{p}_Z(z)$,
b) $\hat{H}(X) \leq \hat{H}(Z)$
c) $\hat{H}(Y\,|\,Z) \leq \hat{H}(Y\,|\,X)$,
d) $\hat{I}(X;Y) \leq \hat{I}(Z;Y)$,

Proof. Let us denote with p and q the $\hat{p}_{X\cup Y}$ and $\hat{p}_{Z\cup Y}$ distributions respectively. Statement a) follows from the definition. For b), we define $h(x) = -p(x)\log p(x)$ for $x \in X$, and similarly $h(z)$ for $z \in Z$. We show that for all $x \in X$, $h(x) \leq \sum_{z\in\pi^{-1}(x)} h(z)$. The statement then follows from the definition of $\hat{H}$. We have

$$\begin{aligned} h(x) &= -p(x)\log p(x) \\ &= -\left(\sum_{z\in\pi^{-1}(x)} q(z)\right)\log\left(\sum_{z\in\pi^{-1}(x)} q(z)\right) \\ &= -\sum_{z\in\pi^{-1}(x)}\left(q(z)\log\left(\sum_{s\in\pi^{-1}(x)} q(s)\right)\right) \\ &\leq -\sum_{z\in\pi^{-1}(x)} q(z)\log q(z) = \sum_{z\in\pi^{-1}(x)} h(z)\ , \end{aligned}$$

where the inequality follows from the monotonicity of the $\log$ function (and the fact that $q(z)$ is positive for all $z \in Z$).
c) Let us first recall the log-sum inequality [18, p. 31]: for non-negative numbers $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_n$,

$$\sum_{i=1}^{n} a_i \log\frac{a_i}{b_i} \geq \Big(\sum_{i=1}^{n} a_i\Big)\frac{\sum_{i=1}^{n} a_i}{\sum_{i=1}^{n} b_i} \tag{2}$$

with equality if and only if a_i/b_i constant. We have:

$$\begin{aligned}\hat{H}(Y \mid Z) &= -\sum_{z\in Z, y\in Y} q(z,y)\log\frac{q(z,y)}{q(z)} \\ &\overset{(a)}{=} -\sum_{x\in X, y\in Y}\sum_{z\in\pi^{-1}(x)} q(z,y)\log\frac{q(z,y)}{q(z)} \\ &\overset{(2)}{\leq} -\sum_{x\in X, y\in Y}\Big(\sum_{z\in\pi^{-1}(x)} q(z,y)\Big)\frac{\sum_{z\in\pi^{-1}(x)} q(z,y)}{\sum_{z\in\pi^{-1}(x)} q(z)} \\ &= -\sum_{x\in X, y\in Y} p(x,y)\log\frac{p(x,y)}{p(x)} = \hat{H}(Y \mid X)\end{aligned}$$

d) We have $\hat{I}(Z;Y) = \hat{H}(Y) - \hat{H}(Y \mid Z) \leq \hat{H}(Y) - \hat{H}(Y \mid X) = \hat{I}(X;Y)$ following from (c). □

In order to analyze monotonicity properties of the permutation model, the following additional definition is useful.

Definition 2. *We call a labeling X **homomorphic** to a labeling Z (w.r.t. the target variable Y), denoted as $X \precsim Z$, if there exists $\sigma \in S_n$ with $Y \equiv Y_\sigma$ such that $X \preceq Z_\sigma$.*

See Tab. I for examples of both introduced relations. Importantly, the inequality of mutual information for specializations (Prop. 1d) carries over to homomorphic variables and in turn to their correction terms.

Proposition 2. *Given variables X, Z and Y, with $X \precsim Z$, the following statements hold:*

a) $\hat{I}(X;Y) \leq \hat{I}(Z;Y)$
b) $\hat{m}_o(X,Y,n) \leq \hat{m}_o(Z,Y,n)$

Proof. Let $\sigma^* \in S_n$ be a permutation for which $Y \equiv Y_{\sigma^*}$ and $X \preceq Z_{\sigma^*}$. Property a) follows from

$$\hat{I}(Z;Y) = \hat{I}(Z_{\sigma^*};Y_{\sigma^*}) = \hat{I}(Z_{\sigma^*};Y) \geq \hat{I}(X;Y) \ ,$$

where the inequality holds from Prop. 1d). For b), note that for every $\sigma \in S_n$, it holds from Prop. 1d) that $\hat{I}(Z_{\sigma\circ\sigma^*};Y) \geq \hat{I}(X_\sigma;Y)$. Hence

$$\begin{aligned}\hat{m}_o(Z,Y,n) &= \frac{1}{n!}\sum_{\sigma\in S_n}\hat{I}(Z_\sigma;Y) \\ &= \frac{1}{n!}\sum_{\sigma\in S_n}\hat{I}(Z_{\sigma\circ\sigma^*};Y) \\ &\geq \frac{1}{n!}\sum_{\sigma\in S_n}\hat{I}(X_\sigma;Y) = \hat{m}_o(X,Y,n)\end{aligned}$$

□

B. Search Algorithms

Effective algorithms for maximizing the reliable fraction of information over all subsets $\mathcal{X} \subseteq \mathcal{I}$ are enabled by the concept of bounding functions. A function $\bar{f}$ is called an **admissible bounding function** for an optimization function f if for all candidate solutions $\mathcal{X} \subseteq \mathcal{I}$, it holds that $\bar{f}(\mathcal{X}) \geq f(\mathcal{X}')$ for all $\mathcal{X}'$ with $\mathcal{X} \subseteq \mathcal{X}' \subseteq \mathcal{I}$. Such functions allow to prune all supersets $\mathcal{X}'$ of $\mathcal{X}$ whenever $\bar{f}(\mathcal{X}) \leq f(\mathcal{X}^*)$ for the current best solution $\mathcal{X}^*$ found during the optimization process.

X_1	X_2	X_3	X_4	Y
a	a	a	b	a
a	b	b	a	b
b	c	b	b	b
b	c	c	c	b

Table I: **Specialization and homomorphism examples**. We have $X_1 \preceq X_2$, $X_1 \precsim X_2$, $X_1 \precsim X_3$, $X_1 \precsim X_4$, $X_2 \precsim X_3$. Note that $X_3 \not\precsim X_4$ as there is no $\sigma \in S_4$ that satisfies specialization w.r.t. X_4 and $Y \equiv Y_\sigma$

Algorithm 1 OPUS: Given a set of input variables $\mathcal{I}$, function f, bounding function $\bar{f}$, and $\alpha \in (0,1]$, the algorithm returns the $\mathcal{X}^* \subseteq \mathcal{I}$ satisfying $f(\mathcal{X}^*) \geq \alpha \max\{f(\mathcal{X}') \colon \mathcal{X}' \subseteq \mathcal{I}\}$

1: **function** OPUS($\mathbf{Q}, \mathcal{S}$)
2: **if** $\mathbf{Q}$ is empty **then**
3: **return** $\mathcal{S}$
4: **else**
5: $(\mathcal{X}, \mathcal{Z}) = pop(\mathbf{Q})$
6: $\mathbf{R} = \{(\mathcal{X} \cup \{Z\}, Z) \colon Z \in \mathcal{Z}\}$
7: $\mathcal{X}^* = \arg\max\{f(\mathcal{X}') \colon \mathcal{X}' \in \mathbf{R} \cup \{\mathcal{S}\}\}$
8: $\mathbf{R}' = \{(\mathcal{X}', Z) \in \mathbf{R} \colon \alpha\bar{f}(\mathcal{X}') > f(\mathcal{X}^*)\}$
9: $\mathcal{Z}' = \{Z \colon (\mathcal{X}', Z) \in \mathbf{R}'\}$
10: $[(\mathcal{X}_1, Z_1), \ldots, (\mathcal{X}_k, Z_k)] = sort(\mathbf{R}')$
11: $\mathbf{Q}' = \mathbf{Q} \cup \{(\mathcal{X}_i, \mathcal{Z}' \setminus \{Z_1, \ldots, Z_i\}) \colon i \in [k]\}$
12: **return** OPUS($\mathbf{Q}', \mathcal{X}^*$)
13: $\mathcal{X}^* =$ OPUS($\{(\emptyset, \mathcal{I})\}, \emptyset$)

Branch-and-bound, as the name suggests, combines this concept with a branching scheme that completely (and non-redundantly) enumerates the search space $2^{\mathcal{I}}$. Here, we consider **optimized pruning for unordered search** (**OPUS**), an advanced variant of branch-and-bound that effectively propagates pruning information to siblings in the search tree [19]. Algorithm 1 shows the details of this approach.

In addition to keeping track of the best solution $\mathcal{X}^*$ seen so far, the algorithm maintains a priority queue $\mathbf{Q}$ of pairs $(\mathcal{X}, \mathcal{Z})$, where $\mathcal{X} \subseteq \mathcal{I}$ is a candidate solution and $\mathcal{Z} \subseteq \mathcal{I}$ constitutes the variables that can still be used to augment $\mathcal{X}$, e.g., $\mathcal{X}' = \mathcal{X} \cup \{Z\}$ for a $Z \in \mathcal{Z}$. The top element is the one with the smallest cardinality and the highest potential (a combination of breadth-first and best-first order). Starting with $\mathbf{Q} = \{(\emptyset, \mathcal{I})\}$, $\mathcal{X}^* = \emptyset$, and a desired approximation guarantee $\alpha \in (0,1]$, in every iteration OPUS creates all refinements of the top element of $\mathbf{Q}$ and updates $\mathcal{X}^*$ accordingly (lines 5-7). Next the refinements are pruned using $\bar{f}$ and α (line 8). Following, the pruned list is sorted according to decreasing potential (this is a heuristic that propagates the most augmentation elements to the least promising refinements [19]), the possible augmentation elements $\mathcal{Z}'$ are non-redundantly propagated to the refinements of the top element, and finally the priority queue is updated with the new candidates (lines 9-11).

Algorithm 2 GREEDY: Given a set of input variables $\mathcal{I}$, function f, and bounding function $\bar{f}$, the algorithm returns the $\mathcal{X}^* \subseteq \mathcal{I}$ approximating $f(\mathcal{X}^*) = \max\{f(\mathcal{X}') \colon \mathcal{X}' \subseteq \mathcal{I}\}$

1: **function** GREEDY($\mathcal{C}, \mathcal{S}$)
2: **if** $\mathcal{I} \setminus \mathcal{C}$ is empty or $\bar{f}(\mathcal{C}) \leq f(\mathcal{S})$ **then**
3: **return** $\mathcal{S}$
4: **else**
5: $\mathbf{R} = \{\mathcal{C} \cup \{Z\} \colon Z \in \mathcal{I} \setminus \mathcal{C}\}$
6: $\mathcal{C}^* = \arg\max\{f(\mathcal{X}') \colon \mathcal{X}' \in \mathbf{R}\}$
7: $\mathcal{X}^* = \arg\max\{f(\mathcal{X}') \colon \mathcal{X}' \in \{\mathcal{S}, \mathcal{C}^*\}\}$
8: **return** GREEDY($\mathcal{C}^*, \mathcal{X}^*$)
9: $\mathcal{X}^* =$ GREEDY($\emptyset, \emptyset$)

A commonly used alternative to complete branch-and-bound search for the optimization of dependency measures is the standard **greedy algorithm** (see [11], [20]). This algorithm only refines the best candidate in a given iteration. Moreover, bounding functions can be incorporated as an early termination criterion. For the reliable fraction of information in particular, there is potential to prune many of the higher levels of the search space as it favors solutions that are small in cardinality [14]. The algorithm is presented in Algorithm 2.

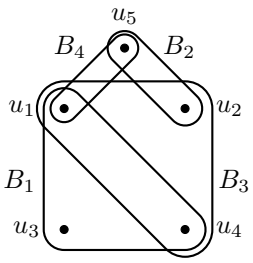
The algorithm keeps track of the best solution $\mathcal{X}^*$ seen, as well as the best candidate for refinement $\mathcal{C}^*$. Starting with $\mathcal{X}^* = \emptyset$ and $\mathcal{C}^* = \emptyset$, the algorithm in each iteration checks whether $\mathcal{C}^*$ can be refined further, i.e., if $\mathcal{I} \setminus \mathcal{C}^*$ is not empty, or if $\mathcal{C}^*$ has potential (the early termination criterion). If not, the algorithm terminates returning $\mathcal{X}^*$ (lines 2-3). Otherwise $\mathcal{C}^*$ is refined to all possible refinements, and the best one is selected as a candidate to update $\mathcal{X}^*$ (lines 5-7).

Concerning the approximation ratio of the greedy algorithm, there exists a large amount of research focused on submodular and/or monotone functions, e.g., [21], [22], [23]. However, we note that $\hat{F}_0$ is neither submodular nor monotone, and hence these results are not directly applicable. To demonstrate empirically the quality of the results, we perform an evaluation in Sec. V-B. We discuss further on this topic in Sec. VI.

III. HARDNESS OF OPTIMIZATION

In this section, we show that the problem of maximizing $\hat{F}_0$ is NP-hard by providing a reduction from the well-known NP-hard **minimum set cover** problem: given a finite universe $U = \{u_1, \ldots, u_n\}$ and collection of subsets $\mathcal{B} = \{B_1, \ldots, B_m\} \subseteq 2^U$, find a set cover, i.e., a sub-collection $\mathcal{C} \subseteq \mathcal{B}$ with $\bigcup \mathcal{C} = U$, that is of minimal cardinality.

The reduction consists of two parts. First, we construct a base transformation $\tau_1(U, \mathcal{B}) = \mathbf{D}_l$ that maps a set cover instance to a dataset $\mathbf{D}_l$ such that set covers correspond to attribute sets with an empirical fraction of information score $\hat{F}$ of 1 and bias correction terms $\hat{b}_0$ that are a monotonically decreasing function of their cardinality. In a second step, we then calibrate the $\hat{b}_0$ terms such that, when considering the corrected score $\hat{F}_0$, they cannot change the order between attribute sets with different $\hat{F}$ values but only act as a tie-breaker between attribute sets of equal $\hat{F}$ value. This is achieved by copying the dataset $\mathbf{D}_l$ a suitable number of times k such that the correction terms are sufficiently small but the overall transformation, denoted $\tau_k(U, \mathcal{B}) = \mathbf{D}_{kl}$, is still of polynomial size.



		$\mathbf{X_1}$	$\mathbf{X_2}$	X_3	X_4	Y
	1	**1**	**a**	1	1	a
	2	**a**	**2**	2	a	a
S_1	3	**3**	**a**	a	a	a
	4	**4**	**a**	4	a	a
	5	**a**	**5**	a	5	a
	6	**a**	**a**	a	a	b
	7	**a**	**a**	a	a	b
S_2	8	**a**	**a**	a	a	b
	9	**a**	**a**	a	a	b
	10	**a**	**a**	a	a	b
	11	**b**	**c**	c	c	c
	12	**c**	**b**	c	c	c
S_3	13	**c**	**c**	b	c	c
	14	**c**	**c**	c	b	c
	15	**c**	**c**	c	c	c

Figure 2: **Base transformation example**. **Left:** a set cover instance $U = \{u_1, \ldots, u_5\}$ and $\mathcal{B} = \{\mathbf{B_1}, \mathbf{B_2}, B_3, B_4\}$. **Right:** the resulting $\mathbf{D}_{15}$ using $\tau_1(U, \mathcal{B})$ (bold indicates the set cover)

The **base transformation** $\tau_1(U, \mathcal{B}) = \mathbf{D}_l$ is defined as follows. The dataset $\mathbf{D}_l$ contains m descriptive attributes $\mathcal{I} = \{X_1, \ldots, X_m\}$ corresponding to the sets of the set cover instance, and a target variable Y. The sample size is $l = 2n + m + 1$ with a logical partition of the sample into the three regions $S_1 = [1, n]$, $S_2 = [n + 1, 2n]$, and $S_3 = [2n + 1, l]$. The target attribute Y assigns to sample points one of three values corresponding to the three regions, i.e., $Y \colon [l] \to \{\mathrm{a}, \mathrm{b}, \mathrm{c}\}$ with

$$Y(j) = \begin{cases} \mathrm{a}, & j \in S_1 \\ \mathrm{b}, & j \in S_2 \\ \mathrm{c}, & j \in S_3 \end{cases}$$

and the descriptive attributes X_i assign up to $n + 3$ distinct values dependending on the set of universe elements covered by set B_i, i.e., $X_i \colon [l] \to \{1, 2, \ldots, n, \mathrm{a}, \mathrm{b}, \mathrm{c}\}$ with

$$X_i(j) = \begin{cases} j, & j \in S_1 \wedge u_j \in B_i \\ \mathrm{a}, & (j \in S_1 \wedge u_j \notin B_i) \vee j \in S_2 \\ \mathrm{b}, & j = 2n + i \\ \mathrm{c}, & j \in S_3 \setminus \{2n + i\} \end{cases}.$$

See Fig. 2 for an illustration. This transformation establishes a one-to-one correspondence of partial set covers $\mathcal{C} \subseteq \mathcal{B}$ and variable sets $\mathcal{X} \subseteq \mathcal{I}$, which we denote as $\mathcal{X}(\mathcal{C})$. Let us denote $(\mathrm{a}, \ldots, \mathrm{a})$ as $\vec{\mathrm{a}}$. The first part of the construction (S_1 and S_2) couples the amount of uncovered elements $U \setminus \bigcup \mathcal{C}$ to the conditional entropy of Y given $\mathcal{X}(\mathcal{C}) = \vec{\mathrm{a}}$ through $\hat{p}(Y = \mathrm{a} \,|\, \mathcal{X}(\mathcal{C}) = \vec{\mathrm{a}}) = |U \setminus \bigcup \mathcal{C}|/(n + |U \setminus \bigcup \mathcal{C}|)$. The second part ($S_3$) links the size of $\mathcal{C}$ to the number of distinct values on S_3.

We can note the following central properties.

Lemma 3. *Let $\tau_1(U,\mathcal{B}) = \mathbf{D}_l$ be the transformation of a set cover instance $(U,\mathcal{B})$ and $\mathcal{C},\mathcal{C}' \subseteq \mathcal{B}$ be two partial set covers. Then the following holds.*

a) If $|\bigcup\mathcal{C}| \geq |\bigcup\mathcal{C}'|$ then $\hat{F}(\mathcal{X}(\mathcal{C});Y) \geq \hat{F}(\mathcal{X}(\mathcal{C}');Y)$; in particular, $\mathcal{C}$ is a set cover, i.e., $\bigcup\mathcal{C} = U$, if and only if $\hat{F}(\mathcal{X}(\mathcal{C});Y) = 1$,

b) If C is a set cover and C' is not *a set cover then $\hat{I}(\mathcal{X}(\mathcal{C});Y) - \hat{I}(\mathcal{X}(\mathcal{C}');Y) \geq 2/l$.*

c) If $\mathcal{C}$ and $\mathcal{C}'$ are both set covers then $\mathcal{X}(\mathcal{C}) \precsim \mathcal{X}(\mathcal{C}')$ if and only if $|\mathcal{C}| \leq |\mathcal{C}'|$.

Proof. Statement a) follows from the definition of τ_1.

To show b), since $\hat{F}(\mathcal{X}(\mathcal{C}');Y)$ and thus $\hat{I}(\mathcal{X}(\mathcal{C}');Y)$ are monotone in $|\bigcup\mathcal{C}'|$, it is sufficient to consider the case where $|U \setminus \bigcup\mathcal{C}'| = 1$. In this case we have

$$\hat{I}(\mathcal{X}(\mathcal{C});Y) - \hat{I}(\mathcal{X}(\mathcal{C}');Y) = \hat{H}(Y \mid \mathcal{X}(\mathcal{C}')) - \underbrace{\hat{H}(Y \mid \mathcal{X}(\mathcal{C}))}_{=0}$$

and, moreover, as required

$$\begin{aligned}\hat{H}(Y \mid \mathcal{X}(\mathcal{C}')) &= -\hat{p}(\vec{\mathrm{a}},\mathrm{a}) \log \hat{p}(\mathrm{a} \mid \vec{\mathrm{a}}) - \hat{p}(\vec{\mathrm{a}},\mathrm{b}) \log \hat{p}(\mathrm{b} \mid \vec{\mathrm{a}}) \\ &= -\frac{1}{l} \log \left(\frac{1}{n+1}\right) - \frac{n}{l} \log \left(\frac{n}{n+1}\right) \geq \frac{2}{l} \ .\end{aligned}$$

For c) observe that for a variable set $\mathcal{X} = \mathcal{X}(\mathcal{C})$ corresponding to a set cover $\mathcal{C}$, we have for all $i,j \in S_1$ that $\mathcal{X}(i) \neq \mathcal{X}(j)$. Thus, $\mathcal{X}_{S_1} \equiv \mathcal{X}'_{S_1}$ for a variable set $\mathcal{X}' = \mathcal{X}(\mathcal{C}')$ corresponding to another set cover $\mathcal{C}'$. Moreover, we trivially have $\mathcal{X}_{S_2} \equiv \mathcal{X}'_{S_2}$. Finally, let $Q, Q' \subseteq S_3$ denote the indices belonging to S_3 where $\mathcal{X}$ and $\mathcal{X}'$ take on values different from $(c,\ldots,c)$. Note that all values in these sets are unique. Furthermore, if $|\mathcal{C}| \leq |\mathcal{C}'|$ then $|Q| \leq |Q'|$ and in turn $|Q \setminus Q'| \leq |Q' \setminus Q|$. This means we can find a permutation $\sigma \in S_n$ such that for all $i \in Q \setminus Q'$ it holds that $\sigma(i) = j$ with $j \in Q' \setminus Q$ and $\sigma(i) = i$ for $i \notin Q \cap Q'$ (that is σ permutes all indices of non-$(c,\ldots,c)$ values of $\mathcal{C}$ in S_3 to indices of non-$(c,\ldots,c)$ values of $\mathcal{C}'$). For such a permutation it holds that $Y_\sigma \equiv Y$ and $\mathcal{X}_{S_3} \preceq \mathcal{X}'_{S_3\sigma}$. Therefore, $\mathcal{X} \precsim \mathcal{X}'$ as required. □

Now, although set covers $\mathcal{C} \subseteq \mathcal{B}$ correspond to variable sets $\mathcal{X}$ with the maximal empirical fraction of information value of 1, due to the correction term, it can happen that $\hat{F}_0(\mathcal{X}';Y) > \hat{F}_0(\mathcal{X};Y)$ for a variable set $\mathcal{X}'$ corresponding to a partial set cover. To prevent this we make use of the following upper bound of the expected mutual information under the permutation model.

Proposition 4 ([15], Thm. 7)**.** *For a sample of size n of the joint distribution of variables A and B having a and b distinct values, respectively, we have*

$$\hat{m}_0(A,B,n) \leq \log \left(\frac{n + ab - a - b}{n-1}\right)$$

This result implies that we can arbitrarily shrink the correction terms if we increase the sample size but leave the number of distinct values constant. Thus, we define the **extended transformation** $\tau_i(U,\mathcal{B}) = \mathbf{D}_{il}$ through simply copying $\mathbf{D}_l$ a number of i times, i.e., by defining $\mathbf{d}_j = \mathbf{d}_{(j \mod l)}$ for $j \in [l+1, il]$. With this definition we show our main result.

Theorem 5. *Given a sample of the joint distribution of variables $\mathcal{I}$ and Y, the problem of maximizing $\hat{F}_0(\cdot\,;Y)$ over all subsets of $\mathcal{I}$ is NP-hard.*

Proof. We show that there is a number $k \in O(l)$ such that w.r.t. transformation τ_k we have that for all set covers $\mathcal{C} \subseteq \mathcal{B}$ and their corresponding variable sets $\mathcal{X} = \mathcal{X}(\mathcal{C})$, $\hat{m}_0(\mathcal{X},Y,n) < 2/l$. Since all properties of Lemma 3 transfer from τ_1 to τ_k, this implies that for all variable sets $\mathcal{X}' = \mathcal{X}(\mathcal{C}')$ corresponding to non-set-covers $\mathcal{C}' \subseteq \mathcal{B}$, it holds that

$$\begin{aligned}\hat{F}_0(\mathcal{X};Y) &= \hat{F}(\mathcal{X};Y) - \hat{m}_0(\mathcal{X},Y,n)/\hat{H}(Y) \\ &> \hat{F}(\mathcal{X};Y) - 2/l\hat{H}(Y) \\ &\geq \hat{F}(\mathcal{X};Y) - (\hat{I}(\mathcal{X};Y) - \hat{I}(\mathcal{X}';Y))/\hat{H}(Y) \\ &= \hat{F}(\mathcal{X}';Y) \geq \hat{F}_0(\mathcal{X}';Y)\end{aligned}$$

where the greater-than follows from Lm. 3a) and 3b). Thus, any $\mathcal{X}$ with maximum $\hat{F}_0$ corresponds to a set cover $\mathcal{C}$.

Moreover, we know that $\mathcal{C}$ must be a minimum set cover as required, because for a smaller set cover $\mathcal{C}'$, $\mathcal{X}(\mathcal{C}') \precsim \mathcal{X}(\mathcal{C})$ by Lemma 3c) and thus $\hat{b}_0(\mathcal{X}(\mathcal{C}'),Y,n) \leq \hat{b}_0(\mathcal{X}(\mathcal{C}),Y,n)$ from Prop. 2b)—therefore, $\mathcal{X}(\mathcal{C})$ would not maximize $\hat{F}_0$.

To find the number k that defines the final transformation τ_k, let $\mathbf{D}_{il} = \tau_i(U,\mathcal{B})$ and $\mathcal{C}$ be a set cover of $(U,\mathcal{B})$. Since $\mathcal{X} = \mathcal{X}(\mathcal{C})$ has at most $3l$ distinct values in $\mathbf{D}_{il}$ and Y exactly 3, from Prop. 4 and the monotonicity of $\ln$ we know that

$$\ln(2)\hat{m}_0(\mathcal{X}(\mathcal{C}),Y,n) \leq \ln\left(\frac{il+3l}{il-1}\right) \leq \ln\left(\frac{i+3}{i-1}\right) \leq \frac{4}{i-1}$$

where the last inequality follows from $\ln(x) \leq x - 1$. Thus, for $k = \lceil 2l/\ln 2 \rceil + 1 \in O(l)$ we have $\hat{m}_0(\mathcal{X},Y,n) < 2/l$ as required. The proof is concluded by noting that the final transformation $\tau_k(U,\mathcal{B})$ is of size $O(l^2 m)$ (where $l = 2n+m+1$), which is polynomial in the size of the set cover instance. □

IV. Refined bounding function

The NP-hardness established in the previous section excludes (unless P=NP) the existence of a polynomial time algorithm for maximizing the reliable fraction of information, leaving therefore exact but exponential search and heuristics as the two options. For both, and particularly the former, reducing the search space can lead to more effective algorithms. For this purpose, we derive in this section a novel bounding function for $\hat{F}_0$ to be used for pruning.

Recall that an admissible bounding function $\bar{f}$ for effective search is an upper bound to the optimization function value f of all supersets of a candidate solution $\mathcal{X} \subseteq \mathcal{I}$. That is, it must hold that $\bar{f}(\mathcal{X}) \geq f(\mathcal{X}')$ for all $\mathcal{X}'$ with $\mathcal{X} \subseteq \mathcal{X}' \subseteq \mathcal{I}$. At the same time, in order to yield optimal pruning, the bound should be as tight as possible. Thus, the ideal function is

$$\bar{f}_{\text{ideal}}(\mathcal{X}) = \max\{\hat{F}_0(\mathcal{X}';Y) \colon \mathcal{X} \subseteq \mathcal{X}' \subseteq \mathcal{I}\} \ .$$

Computing this function is of course equivalent to the original optimization problem and hence NP-hard. We can, however,

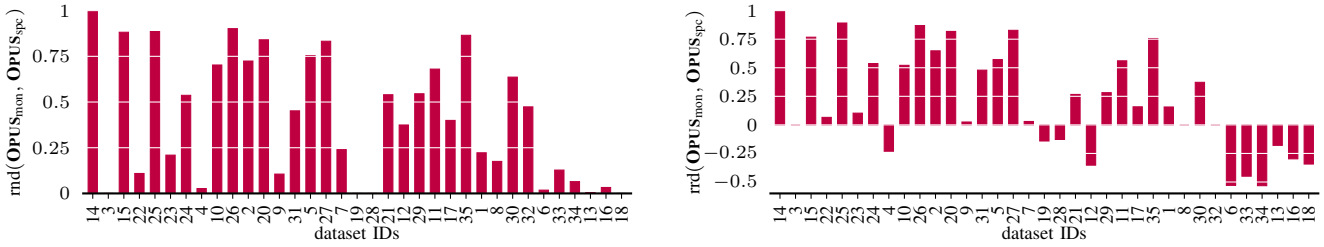


Figure 3: **Evaluating $\bar{f}_{\text{spc}}$ for branch-and-bound optimization.** Relative nodes explored difference (left) and relative runtime difference (right) between methods $\textsc{Opus}_{\text{spc}}$ and $\textsc{Opus}_{\text{mon}}$. Positive (negative) numbers indicate that $\textsc{Opus}_{\text{spc}}$ ($\textsc{Opus}_{\text{mon}}$) is proportionally "better". The datasets are sorted in decreasing number of attributes.

relax the maximum over all supersets to the maximum over all *specializations* of $\mathcal{X}$. That is, we define a bounding function $\bar{f}_{\text{spc}}(\mathcal{X})$ through

$$\begin{aligned}\bar{f}_{\text{spc}}(\mathcal{X}) &= \max\{\hat{F}_0(\mathcal{X}';Y)\colon\ \mathcal{X} \preceq \mathcal{X}'\} \\ &\geq \max\{\hat{F}_0(\mathcal{X}';Y)\colon\ \mathcal{X} \subseteq \mathcal{X}' \subseteq \mathcal{I}\} = \bar{f}_{\text{ideal}}(\mathcal{X})\ .\end{aligned}$$

While this definition obviously constitutes an admissible bounding function, it is unclear how it can be efficiently evaluated. Let us denote by R^+ the operation of joining a labeling R with the target attribute Y, i.e., $R^+ = \{R\} \cup \{Y\}$ (see Tab. II for an example). This definition gives rise to a simple constructive form for computing $\bar{f}_{\text{spc}}$.

Theorem 6. *The function $\bar{f}_{spc}$ can be efficiently computed as $\bar{f}_{spc}(\mathcal{X}) = \hat{F}_0(\mathcal{X}^+;Y)$ in time $O(n|V(\mathcal{X})||V(Y)|)$.*

Proof. We start by showing that the $(\cdot)^+$ operation causes a positive gain in $\hat{F}_0$, i.e., for an arbitrary labeling R it holds that $\hat{F}_0(R^+;Y) \geq \hat{F}_0(R;Y)$.

Let us define $\hat{I}_0(R^+;Y) = \hat{I}(R^+;Y) - \hat{E}_0(\hat{I}(R^+,Y))$. It is then sufficient to show $\hat{I}_0(R^+;Y) \geq \hat{I}_0(R;Y)$. We have

$$\begin{aligned}\hat{I}_0(R^+;Y) =& \left(\hat{H}(Y) + \hat{H}(R^+) - \hat{H}(R^+,Y)\right) \\ &- \frac{1}{n!}\left(\sum_{\sigma\in S_n}(\hat{H}(Y_\sigma) + \hat{H}(R^+) - \hat{H}(R^+,Y_\sigma)\right) \\ =& \frac{1}{n!}\sum_{\sigma\in S_n}\hat{H}(R^+,Y_\sigma) - \hat{H}(R^+,Y) \\ \geq& \frac{1}{n!}\sum_{\sigma\in S_n}\hat{H}(R,Y_\sigma) - \hat{H}(R,Y) = \hat{I}_0(R;Y)\ ,\end{aligned}$$

since $\hat{H}(R^+,Y) = \hat{H}(R\cup Y,Y) = \hat{H}(R,Y)$, and from Prop. 1b), for every $\sigma \in S_n$, $\hat{H}(R^+,Y_\sigma) \geq \hat{H}(R,Y_\sigma)$.

To conclude, let $\mathcal{Z}$ be an arbitrary specialization of $\mathcal{X}$. We have by definition of $\mathcal{Z}$ and $\mathcal{Z}^+$, that $\mathcal{X}^+ \preceq \mathcal{Z}^+$. Moreover, $\hat{F}(\,\cdot\,;Y) = \hat{F}(\{\,\cdot\,\}\cup\{Y\};Y) = 1$. Thus

$$\begin{aligned}\hat{F}_0(\mathcal{X}^+;Y) =& \hat{F}(\mathcal{X}^+;Y) - \hat{b}_0(\mathcal{X}^+,Y,n) \\ =& 1 - \hat{b}_0(\mathcal{X}^+,Y,n) \\ \geq& 1 - \hat{b}_0(\mathcal{Z}^+,Y,n) \\ =& \hat{F}_0(\mathcal{Z}^+;Y) \geq \hat{F}_0(\mathcal{Z};Y)\ ,\end{aligned}$$

as required. Here, the first inequality follows from Prop. 1b), the second from the positive gain of $\mathcal{Z}^+$ over $\mathcal{Z}$.

For the complexity recall that $\hat{b}_0(\mathcal{X},Y,n)$ can be computed in time $O(n\max\{|V(\mathcal{X})|,|V(Y)|\})$ (see appendix). The complexity follows from $|V(\mathcal{X}^+)| \leq |V(\mathcal{X})||V(Y)|$. □

Note that the $\mathcal{X}^+$ operation does not have to be computed explicitly because the non-zero marginal counts for $\mathcal{X}^+$ can simply be obtained as the non-zero counts of the joint contingency table of $\mathcal{X}$ and Y (which has to be computed anyway for $\hat{F}_0$; see appendix).

Intuitively, $\mathcal{X}^+$ constitutes the most efficient specialization of $\mathcal{X}$ in terms of growth in $\hat{F}$ and $\hat{b}_0$ (which is not necessarily attainable by a subset of input variables). In contrast, the bounding function $\bar{f}_{\text{mon}}(\mathcal{X}) = 1 - \hat{b}_0(\mathcal{X},Y,n)$ of [14] assumes that full information about the target can be attained (i.e., $\hat{F} = 1$) without "paying" an increased $\hat{b}_0$ term. The following proposition shows that this idea leads to an inferior bound.

Proposition 7. *Let $\mathcal{X} \subseteq \mathcal{I}$ and $\Delta = \bar{f}_{mon}(\mathcal{X}) - \bar{f}_{spc}(\mathcal{X})$. The following statements hold:*

a) $\Delta \geq 0$ for all datasets, i.e., $\bar{f}_{spc}(\mathcal{X}) \leq \bar{f}_{mon}(\mathcal{X})$

b) there are datasets $\mathbf{D}_{4l}$ for all $l \geq 1$ s.t. $\Delta \in \Omega(1 - \frac{1}{\log 2l})$

Proof. a)

$$\begin{aligned}\bar{f}_{\text{spc}}(\mathcal{X}) =& 1 - \hat{b}_0(\mathcal{X}^+,Y,n) \\ \leq& 1 - \hat{b}_0(\mathcal{X},Y,n) = \bar{f}_{\text{mon}}(\mathcal{X})\ ,\end{aligned}$$

where the inequality holds from Prop. 1b) and $\mathcal{X} \preceq \mathcal{X}^+$.

b) For $l \geq 1$ we construct a dataset $\mathbf{D}_{4l}$ with two variables $X\colon [4l] \to \{\mathrm{a},\mathrm{b}\}$ and $Y\colon [4l] \to [2l]$, with

$$X(i) = \begin{cases}\mathrm{a}, & i \mod 2 = 1 \\ \mathrm{b}, & i \mod 2 = 0\end{cases}$$

and $Y(i) = \lceil i/2 \rceil$ respectively (see Tab. II). We have

$$\begin{aligned}\Delta =& 1 - \hat{b}_0(X,Y,4l) - 1 + \underbrace{\hat{b}_0(X^+,Y,4l)}_{=\hat{H}(Y\,|\,X^+_\sigma)/\hat{H}(Y)=0} \\ =& \frac{1}{n!}\sum_{\sigma\in S_n}\hat{H}(Y\,|\,X_\sigma)/\hat{H}(Y) \\ \geq& \min_{\sigma\in S_n}\hat{H}(Y\,|\,X_\sigma)/\hat{H}(Y)\ .\end{aligned}$$

X	Y	X^+	X_{σ^*}
a	1	(a,1)	a
b	1	(b,1)	a
a	2	(a,2)	a
b	2	(b,2)	a
		⋮	

X	Y	X^+	X_{σ^*}
		⋮	
a	2l-1	(a,2l-1)	b
b	2l-1	(b,2l-1)	b
a	2l	(a,2l)	b
b	2l	(b,2l)	b

Table II: **Construction showing advantage of upper bound** $1-\hat{b}_0(X^+,Y,n)=0$ **vs** $1-\hat{b}_0(X,Y,n)\geq 1-1/\log(n/2)$, i.e., all specializations of X that contain full information about Y are injective (key) maps (see Prop. 7).

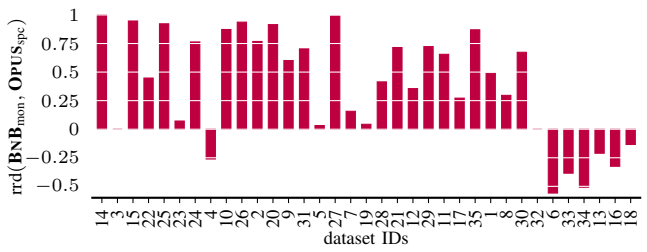


Figure 4: **Evaluating the two branch-and-bound frameworks.** Relative time difference between methods **OPUS**$_{\text{spc}}$ and **BNB**$_{\text{mon}}$. Positive (negative) numbers indicate that **OPUS**$_{\text{spc}}$ (**BNB**$_{\text{mon}}$) is proportionally "better". Datasets are sorted in decreasing number of attributes.

One can show that the minimum of the last step is attained by the permutation $\sigma^* \in S_n$ with

$$\sigma^*(i) = \begin{cases} 2i-1, & i \in [1, 2l] \\ 4l - 2(4l-i), & i \in [2l+1, 4l] \end{cases} ,$$

which corresponds to sorting the a and b values of X (see Tab. II). For this permutation the normalized conditional entropy evaluates to $1-1/\log(2l)$ as required. □

Thus, we have established that $\bar{f}_{\text{spc}}$ is not only tighter than $\bar{f}_{\text{mon}}$, but even that the difference can be arbitrary close to 1 (for an increasing domain size of Y). Put differently, their ratio, and thus the potential for additional pruning, is unbounded.

Computationally, $\bar{f}_{\text{spc}}(\mathcal{X})$ is more expensive than $\bar{f}_{\text{mon}}(\mathcal{X})$ by a factor of $|V(Y)|$. In order to partially alleviate this increase, note that one can first check the pruning condition (line 8 of Alg. 1 or line 2 of Alg. 2) w.r.t. $\bar{f}_{\text{mon}}$ and only compute $\bar{f}_{\text{spc}}$ if that first check fails. That is, whenever $\bar{f}_{\text{mon}}(\mathcal{X})$ is sufficient to prune a candidate $\mathcal{X}$ we can still do so with the same computational complexity. However, the additional evaluation of $\bar{f}_{\text{spc}}(\mathcal{X})$ can be a disadvantage in case it still does not allow to prune. This trade-off is evaluated in the following section.

V. Evaluation

For ease of comparison to [14], we consider datasets from the KEEL data repository [24]. In particular, we use all classification datasets with $d \in [10, 90]$ and no missing values, resulting in 35 datasets with 52000 and 30 rows and columns on average, respectively. All metric attributes are discretized in 5 equal-frequency bins. The datasets are summarized in Table III. The runtimes are averaged over 3 runs. All implementations are available online[1].

We use two metrics for evaluation, the relative runtime difference and the relative difference in number of explored nodes. For methods A and B, the relative runtime difference on a particular dataset is computed as

$$\text{rrd}(A,B) = \frac{(\tau_A - \tau_B)}{\max(\tau_A, \tau_B)} ,$$

where τ_A and τ_B are the run times for A and B respectively. The rrd score lies in $[-1,1]$, where positive (negative) values indicate that B is proportionally faster (slower). For example, a rrd score of 0.5 corresponds to a factor of 2 speed-up, 0.66 to a factor of 3, 0.75 to 4 etc. The relative nodes explored difference rnd is defined similarly. For both scores, we consider $(-0.5, 0.5)$ to be a region of practical equivalence, i.e., a factor of 2 of improvement is required to consider a method "better".

A. Branch-and-bound

We first investigate the effect of the refined bounding function by comparing **OPUS**$_{\text{spc}}$ and **OPUS**$_{\text{mon}}$, i.e., Alg. 1 with $\bar{f}_{\text{spc}}$ and $\bar{f}_{\text{mon}}$ as bounding functions respectively. Last, we compare the proposed branch-and-bound framework **OPUS**$_{\text{spc}}$ to the one of [14], which we call **BNB**$_{\text{mon}}$ (a combination of best-first search and refinement based on lexicographical order). For a fair comparison, we set a common α value for all three methods on each dataset by determining the largest α value in increments of 0.05 such that they terminate in less than 90 minutes (see Tab. III).

In Fig. 3 we present the comparison between **OPUS**$_{\text{spc}}$ and **OPUS**$_{\text{mon}}$. The left plot demonstrates that $\bar{f}_{\text{spc}}$ can lead to a considerable reduction of nodes explored over $\bar{f}_{\text{mon}}$. In particular, 15 cases have at least a factor of 2 reduction, 7 have 4, and there is one 1 with 760. For 20 cases there is no practical difference. The plot validates that the potential for additional pruning is indeed unbounded (Sec. IV). In terms of runtime efficiency (right plot), **OPUS**$_{\text{spc}}$ is "faster" in 70% of the datasets. In more detail, and considering practical improvements, 12 datasets have at least a factor of 2 speedup, 6 have 4, 1 has 266, while only 2 have a factor of 2 slowdown. Moreover, we observe from the plot (where datasets are sorted in decreasing number of attributes) a clear correlation between number of attributes and efficiency: the 6 out of 10 datasets with the slowdown are also the ones with the lowest number of features. Overall, $\bar{f}_{\text{spc}}$ leads to a more effective optimization with branch-and-bound, and particularly for the higher-dimensional cases.

Following, we compare **OPUS**$_{\text{spc}}$ to **BNB**$_{\text{mon}}$, presenting the results in Fig. 4. The plot is quite evident that the new framework outperforms the baseline. There are 16 cases with at least a speedup of 2x, 10 with 4x, and there even exists a case with 2880x. Moreover, since both **OPUS**$_{\text{mon}}$ and **BNB**$_{\text{mon}}$ use the same bounding function $\bar{f}_{\text{mon}}$, the two plots Fig. 4 and Fig. 3

[1] https://github.com/pmandros/fodiscovery

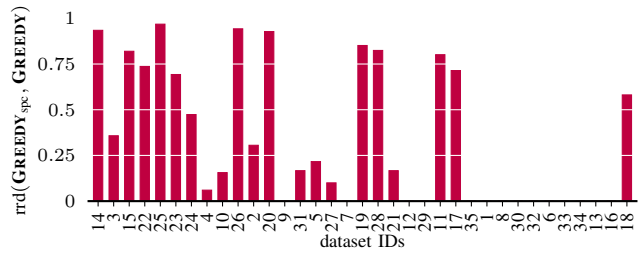


Figure 5: **Evaluating $\bar{f}_{\text{spc}}$ for heuristic optimization.** Relative time difference between methods **GREEDY**$_{\text{spc}}$ and **GREEDY**. Positive (negative) numbers indicate that **GREEDY**$_{\text{spc}}$ (**GREEDY**) is proportionally "better". The datasets are sorted in decreasing number of attributes.

(right) suggest that Alg 1 is a more effective branch-and-bound framework for the reliable fraction of information.

B. Greedy

We begin the evaluation with the performance of $\bar{f}_{\text{spc}}$ for heuristic search. We present the relative runtime differences of **GREEDY** and **GREEDY**$_{\text{spc}}$, i.e., Alg 2 with and without $\bar{f}_{\text{spc}}$, in Fig. 5 (raw results in Table III). The plot shows that $\bar{f}_{\text{spc}}$ indeed improves the efficiency of the heuristic search, as we find that for 12 datasets there is a speedup of at least a factor of 2, and 8 of at least a factor of 4.

Next, we investigate the quality of the greedy results. Note that this is possible as we have access to the branch-and-bound results. In Fig. 6 we plot the differences between the $\hat{F}_0$ score of the results obtained by greedy and branch-and-bound on each dataset (raw results in Table III). Note that branch-and-bound uses the same α values as with the experiments in Sec V-A, and that we only plot the non-zero differences in the two plots, left for $\alpha = 1$, i.e, optimal solutions, and right for $\alpha < 1$, i.e., approximate solutions with guarantees.

At a first glance, we observe that there is no difference in 21 out of 35 cases considered, 7 where greedy is better (this of course on the datasets where $\alpha < 1$), and 7 for branch-and-bound. Out of the 21 cases where the two algorithms have equal $\hat{F}_0$, 16 of them have $\alpha = 1$, i.e., the greedy algorithm is optimal roughly 45% of the time. Moreover, the cases where branch-and-bound is better is only by a small margin, 0.03 on average, while greedy "wins" by 0.1 on average. Another observation from the right plot of Fig. 6 is that the largest differences between the two algorithms is for the 3 datasets where the lowest α values where used, i.e., $0.05, 0.1$, and 0.35.

Lastly in Fig 7, we consider the relative runtime difference between the greedy algorithm and branch-and-bound, i.e., **GREEDY**$_{\text{spc}}$ and **OPUS**$_{\text{spc}}$. As expected, the greedy algorithm is significantly faster in the majority of cases. There are, however, 4 cases where branch-and-bound terminates much faster, which also happen to coincide with more aggressive α values for branch-and-bound.

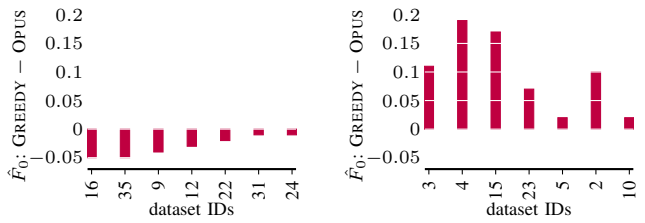


Figure 6: **Evaluating the heuristic algorithm for result quality. Left**: difference in $\hat{F}_0$ between methods **GREEDY**$_{\text{spc}}$ and **OPUS**$_{\text{spc}}$ (i.e., $\hat{F}_0(\mathcal{X}^*_{grd}; Y) - \hat{F}_0(\mathcal{X}^*_{bnb}; Y)$ where $\mathcal{X}^*_{grd}$ and $\mathcal{X}^*_{bnb}$ are the solutions of Alg. 2 and 1 respectively) for $\alpha = 1$. Since $\alpha = 1$, the negative values close to 0 indicate that Alg. 2 retrieves nearly optimal solutions. Data are sorted in increasing quality difference. **Right**: difference for $\alpha < 1$. Positive values indicate that Alg. 2 retrieves better solutions when Alg. 1 uses guarantees $\alpha < 1$. Data are sorted in increasing α values.

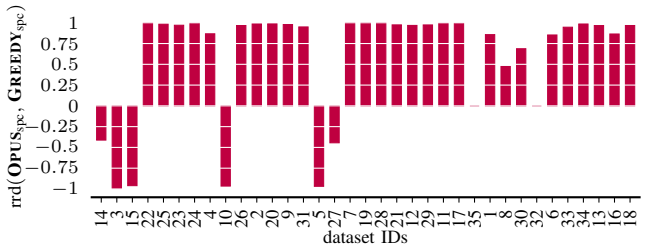


Figure 7: **Evaluating the heuristic algorithm in terms of running time.** Relative time difference between methods **GREEDY**$_{\text{spc}}$ and **OPUS**$_{\text{spc}}$. Positive (negative) numbers indicate that **GREEDY**$_{\text{spc}}$ (**OPUS**$_{\text{spc}}$) is proportionally "better". Datasets are ordered in decreasing number of attributes.

VI. Conclusion

We investigated the algorithmic aspect of discovering dependencies in data using the reliable fraction of information, where we proved the NP-hardness of the problem and derived a refined bounding function for more effective optimization. Moreover, we considered an improved branch-and-bound algorithm and explored the aspects of heuristic optimization. The experimental evaluation showed that the refined bounding function is very effective for both types of optimization, the proposed branch-and-bound framework outperforms the baseline, and that the greedy algorithm provides solutions that are nearly optimal.

While the given reduction from set cover can be extended to show that, unless P=NP, no fully polynomial time approximation scheme exists, the possibility of weaker approximation guarantees remains. In particular, the strong empirical performance of the greedy algorithm hints that $\hat{F}_0$ could have a certain structure favored by the greedy algorithm, e.g., some weaker form of submodularity (we remind that $\hat{F}_0$ is neither submodular nor monotone). For instance, we could explore ideas from Horel and Singer [25] where a monotone function is e-approximately submodular if it can be bounded by a submodular function within $1 \pm e$. Another idea is that of

restricted submodularity for monotone functions [26], where a function is submodular over a subset of the search space. It might be that the greedy algorithm only considers candidates where $\hat{F}_0$ is submodular.

Furthermore, the proposed bounding function is likely to be applicable to a larger selection of corrected-for-chance dependence measures, and a general framework for maximizing reliable measures could be established.

REFERENCES

[1] R. Ramakrishnan and J. Gehrke, *Database Management Systems*, 2nd ed. McGraw-Hill Higher Education, 2000.

[2] L. Song, A. Smola, A. Gretton, J. Bedo, and K. Borgwardt, "Feature selection via dependence maximization," *JMLR*, vol. 13, pp. 1393–1434, 2012.

[3] W. Ziarko, "Rough set approaches for discovery of rules and attribute dependencies," *Handbook of data mining and knowledge discovery*, pp. 328–338, 2002.

[4] L. M. Ghiringhelli, J. Vybiral, S. V. Levchenko, C. Draxl, and M. Scheffler, "Big data of materials science: Critical role of the descriptor," *Physical review letters*, vol. 114, no. 10, p. 105503, 2015.

[5] R. Ouyang, S. Curtarolo, E. Ahmetcik, M. Scheffler, and L. Ghiringhelli, "Sisso: a compressed-sensing method for systematically identifying efficient physical models of materials properties," no. 1710.03319, 2017.

[6] R. Cavallo and M. Pittarelli, "The theory of probabilistic databases," in *Proceedings of the 13th International Conference on Very Large Data Bases (VLDB), Brighton, UK*, 1987, pp. 71–81.

[7] C. Giannella and E. L. Robertson, "On approximation measures for functional dependencies," *Information Systems*, vol. 29, no. 6, pp. 483–507, 2004.

[8] M. Reimherr and D. L. Nicolae, "On quantifying dependence: A framework for developing interpretable measures," *Statistical Science*, vol. 28, no. 1, pp. 116–130, 2013.

[9] S. Romano, N. X. Vinh, J. Bailey, and K. Verspoor, "A framework to adjust dependency measure estimates for chance," in *Proceedings of the SIAM International Conference on Data Mining (SDM), Miami, FL*. SIAM, 2016.

[10] M. S. Roulston, "Estimating the errors on measured entropy and mutual information," *Physica D: Nonlinear Phenomena*, vol. 125, no. 3, pp. 285–294, 1999.

[11] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, Mar. 2003.

[12] Y. Huhtala, J. Kärkkäinen, P. Porkka, and H. Toivonen, "Tane: An efficient algorithm for discovering functional and approximate dependencies," *The Computer Journal*, vol. 42, no. 2, pp. 100–111, 1999.

[13] S. Kruse and F. Naumann, "Efficient discovery of approximate dependencies," *Proc. VLDB Endow.*, vol. 11, no. 7, pp. 759–772, Mar. 2018.

[14] P. Mandros, M. Boley, and J. Vreeken, "Discovering reliable approximate functional dependencies," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '17. ACM, 2017.

[15] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *Journal of Machine Learning Research*, vol. 11, no. Oct, pp. 2837–2854, 2010.

[16] H. Lancaster, *The chi-squared distribution*, ser. Wiley series in probability and mathematical statistics. Probability and mathematical statistics. Wiley, 1969.

[17] S. Romano, J. Bailey, X. V. Nguyen, and K. Verspoor, "Standardized mutual information for clustering comparisons: One step further in adjustment for chance." in *Proceedings of the 31st International Conference on Machine Learning (ICML), Beijing, China*, 2014, pp. 1143–1151.

[18] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York, NY, USA: Wiley-Interscience, 1991.

[19] G. I. Webb, "Opus: An efficient admissible algorithm for unordered search," *Journal of Artificial Intelligence Research*, vol. 3, pp. 431–465, 1995.

[20] G. Brown, A. Pocock, M.-J. Zhao, and M. Luján, "Conditional likelihood maximisation: A unifying framework for information theoretic feature selection," *J. Mach. Learn. Res.*, vol. 13, pp. 27–66, Jan. 2012.

[21] U. Feige, V. S. Mirrokni, and J. Vondrak, "Maximizing non-monotone submodular functions," *SIAM Journal on Computing*, vol. 40, no. 4, pp. 1133–1153, 2011.

[22] A. Das and D. Kempe, "Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection," in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ser. ICML'11, 2011, pp. 1057–1064.

[23] A. A. Bian, J. M. Buhmann, A. Krause, and S. Tschiatschek, "Guarantees for greedy maximization of non-submodular functions with applications," in *International Conference on Machine Learning (ICML)*, 2017.

[24] J. Alcalà-Fdez, A. Fernàndez, J. Luengo, J. Derrac, and S. Garcìa, "Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework." *Multiple-Valued Logic and Soft Computing*, vol. 17, no. 2-3, pp. 255–287, 2011.

[25] T. Horel and Y. Singer, "Maximization of approximately submodular functions," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 3045–3053.

[26] D.-Z. Du, R. L. Graham, P. M. Pardalos, P.-J. Wan, W. Wu, and W. Zhao, "Analysis of greedy approximations with nonsubmodular potential functions," in *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '08, 2008, pp. 167–175.

APPENDIX

Here, we recap how to efficiently compute the correction term $\hat{m}_o(\mathcal{X}, Y, n)$ for an attribute set $\mathcal{X} \subseteq \mathcal{I}$ and target Y in $\mathbf{D}_n$ (cf. [14], [17]). Let the observed domains of $\mathcal{X}$ and Y be $V(\mathcal{X}) = \{\mathbf{x}_1, \ldots, \mathbf{x}_R\}$ and $V(Y) = \{y_1, \ldots, y_C\}$, respectively. We define shortcuts for the observed marginal counts $a_i = c(\mathcal{X} = \mathbf{x}_i)$ and $b_j = c(Y = y_j)$ as well as for the joint counts $c_{i,j} = c(\mathcal{X} = \mathbf{x}_i, Y = y_j)$. The **contingency table** $\mathbf{c}$ for $\mathcal{X}$ and Y is then the complete joint count configuration $\mathbf{c} = \{c_{i,j} : 1 \leq i \leq R, 1 \leq j \leq C\}$. The empirical mutual information for $\mathcal{X}$ and Y can then be computed as:

$$\hat{I}(\mathcal{X}, Y) = \hat{I}(\mathbf{c}) = \sum_{i=1}^{R} \sum_{j=1}^{C} \frac{c_{ij}}{n} \log \frac{c_{ij} n}{a_i b_j}$$

Each $\sigma \in S_n$ results in a contingency table $\mathbf{c}^\sigma$. We denote with $\mathcal{T} = \{\mathbf{c}^\sigma : \sigma \in S_n\}$ the set of all such contingency tables. Crucially, all these tables have the same marginal counts a_i, b_j, $i \in [1, R], j \in [1, C]$. Hence, one can rewrite $\hat{m}_o$ as

$$\hat{m}_o(\mathcal{X}, Y, n) = \sum_{\mathbf{c}^\sigma \in \mathcal{T}} \hat{p}_o(\mathbf{c}^\sigma) \sum_{i=1}^{R} \sum_{j=1}^{C} \frac{c^\sigma_{ij}}{n} \log \frac{c^\sigma_{ij} n}{a_i b_j}$$

were $\hat{p}_o(\mathbf{c})$ is the probability of contingency table $\mathbf{c} \in \mathcal{T}$. This allows one to re-order terms to have a per cell contribution to $\hat{m}_o$, rather than per contingency table $\mathbf{c} \in \mathcal{T}$, i.e.,

$$\hat{m}_o(\mathcal{X}, Y, n) = \sum_{i=1}^{R} \sum_{j=1}^{C} \sum_{k=0}^{n} \hat{p}_o(c^\sigma_{ij} = k) \frac{k}{n} \log \frac{kn}{a_i b_j} \ .$$

The individual empirical counts c^σ_{ij} are then distributed hypergeometrically, i.e.,

$$\hat{p}_o(c^\sigma_{ij} = k) = \binom{b_i}{k} \binom{n - b_i}{a_j - k} \Big/ \binom{n}{a_j} \ .$$

These probabilities can be computed efficiently in an incremental manner by noting that they are non-zero only for k between $\max(0, a_i + b_j - n)$ and $\min(a_i, b_j)$ and by using the hypergeometric recurrence formula.

Table III: Datasets used in Section V along with the raw results produced from the evaluation.

						time(s)					nodes explored		$\hat{F}_0$	
ID	dataset	#rows	#attr	#classes	α	OPUS$_{\text{spc}}$	OPUS$_{\text{mon}}$	BNB$_{\text{mon}}$	GREEDY$_{\text{spc}}$	GREEDY	OPUS$_{\text{spc}}$	OPUS$_{\text{mon}}$	OPUS$_{\text{spc}}$	GREEDY$_{\text{spc}}$
1	*australian*	690	14	2	1.00	7.0	8.3	13.7	1.0	1.0	4190	5388	0.54	0.54
2	*chess*	3196	36	2	0.75	192.1	545.9	828.4	2.5	3.6	69713	252766	0.77	0.87
3	*coil2000*	9822	85	2	0.05	1.0	1.0	1.0	189.1	294.4	86	86	0.06	0.17
4	*connect-4*	67557	42	3	0.10	1236.8	951.5	914.3	164.3	174.8	36183	37176	0.10	0.29
5	*fars*	100968	29	8	0.65	3.0	7.0	3.1	93.9	119.8	45	183	0.66	0.68
6	*flare*	1066	11	6	1.00	6.8	3.2	3.0	1.0	1.0	2011	2048	0.65	0.65
7	*german*	1000	20	2	1.00	931.5	960.1	1104.9	1.0	1.0	216250	284397	0.21	0.21
8	*heart*	270	13	2	1.00	1.9	1.9	2.7	1.0	1.0	2275	2758	0.42	0.42
9	*ionosphere*	351	33	2	1.00	46.4	47.6	116.4	1.0	1.0	48094	53784	0.62	0.58
10	*kddcup*	494020	41	23	0.90	18.1	37.8	142.3	520.2	616.4	69	232	0.97	0.99
11	*letter*	20000	16	26	1.00	659.5	1501.0	1915.5	3.8	19.1	4894	15300	0.60	0.60
12	*lymphography*	148	18	4	1.00	31.2	20.2	48.4	1.0	1.0	23971	38319	0.48	0.45
13	*magic*	19020	10	2	1.00	38.5	31.6	30.4	1.3	1.3	1012	1017	0.43	0.43
14	*move-libras*	360	90	15	0.50	1.0	266.6	2881.7	1.7	25.9	213	163630	0.32	0.32
15	*optdigits*	5620	64	104	0.35	1.0	4.3	18.9	25.1	139.3	105	888	0.36	0.53
16	*pageblocks*	5472	10	5	1.00	7.4	5.2	5.0	1.0	1.0	831	859	0.65	0.60
17	*penbased*	10992	16	10	1.00	233.6	277.5	320.9	1.6	5.6	8099	13486	0.75	0.75
18	*poker*	1025010	10	10	1.00	2594.7	1705.2	2247.4	86.0	205.3	908	908	0.57	0.57
19	*ring*	7400	20	2	1.00	1393.9	1197.3	1456.7	1.1	7.4	60460	60460	0.29	0.29
20	*satimage*	6435	36	7	0.80	173.8	954.4	2054.2	2.0	27.6	24622	154287	0.74	0.74
21	*segment*	2310	19	7	1.00	39.1	53.3	137.2	1.0	1.2	8815	19159	0.84	0.84
22	*sonar*	208	60	2	1.00	403.5	431.9	730.8	1.0	3.8	472554	530478	0.34	0.32
23	*spambase*	4597	57	2	0.55	515.6	574.6	555.4	15.4	50.1	167695	212147	0.54	0.60
24	*spectfheart*	267	44	2	1.00	171.1	369.3	724.7	1.0	1.9	155364	335365	0.23	0.22
25	*splice*	3190	60	3	0.65	92.3	851.0	1193.0	1.5	46.9	36052	315125	0.65	0.65
26	*texture*	5500	40	11	0.80	62.9	480.3	1000.8	2.1	36.6	10835	109878	0.76	0.76
27	*thyroid*	7200	21	3	0.50	1.0	5.8	110.4	1.8	2.0	247	1475	0.50	0.50
28	*twonorm*	7400	20	2	1.00	1332.2	1162.4	2274.4	1.3	7.4	60460	60460	0.42	0.42
29	*vehicle*	846	18	4	1.00	38.2	53.2	137.9	1.0	1.0	10670	23462	0.48	0.48
30	*vowel*	990	13	11	1.00	3.2	5.1	9.8	1.0	1.0	590	1622	0.45	0.45
31	*wdbc*	569	30	2	1.00	19.9	38.2	67.0	1.0	1.2	18564	33862	0.76	0.75
32	*wine*	178	13	3	1.00	1.0	1.0	1.0	1.0	1.0	199	378	0.71	0.71
33	*wine-red*	1599	11	11	1.00	18.7	10.3	11.5	1.0	1.0	1481	1698	0.20	0.20
34	*wine-white*	4898	11	11	1.00	77.4	36.2	38.1	1.0	1.0	1881	2011	0.19	0.19
35	*zoo*	101	15	7	1.00	1.0	4.1	7.7	1.0	1.0	773	5724	0.80	0.75
Avg.						296	360	603	32	51	41434	78309	0.51	0.53