社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2004－AL－97　(3)
2004／10／14

二部グラフの高濃度部分グラフ問題の近似アルゴリズムについて

鈴木晶子　　徳山豪

† 東北大学情報科学研究科
E-mail: †{akiko,tokuyama}@dais.is.tohoku.ac.jp

あらまし　二部グラフの高濃度部分グラフ問題と、そのデータマイニングにおける次元制限クラスタリングへの応用を論じる。過去の研究と異なり、出力される部分グラフが小さい場合にも良い理論的近似アルゴリズムを与える。
キーワード　近似アルゴリズム、クラスタリング、高濃度部分グラフ

Approximation algorithms for the bipartite dense subgraph problem

Akiko SUZUKI and Takeshi TOKUYAMA

† GSIS Tohoku University
E-mail: †{akiko,tokuyama}@dais.is.tohoku.ac.jp

Abstract　We consider the (weighted) bipartite dense subgraph problem that extracts a dense subgraph of a given (weighted) bipartite graph. We give approximation algorithms with theoretical error bounds even if the output subgraph is of small size provided that it is dense: This is significantly different from known results on the non-bipartite problem. We also discuss its application to dimension-reducing clustering in data mining applications.
Key words　Approximation Algorithm, Clustering, Dense subgraph

1. Introduction

Motivated from a clustering application, we consider approximate algorithms for the bipartite dense subgraph problem. The unweighted *bipartite dense subgraph problem* is defined as follows:

> Consider a bipartite graph $G = (U, V, E)$, where $|U| = m$ and $|V| = n$. Given two natural numbers $m' \leqq m$ and $n' \leqq n$, find a subgraph $H = (X, Y, F)$ of G such that $|X| = m'$, $|Y| = n'$ and $|F|$ is maximized.

The problem can be considered as a variant of the *dense subgraph problem*, which finds a subgraph with k vertices of a given (not necessary bipartite) graph G with the maximum number of edges. Its weighted version is as follows:

> Consider a weighted bipartite graph $G = (U, V, E)$, where $|U| = m$, $|V| = n$ and each edge e has a nonnegative weight $0 \leqq w(e) \leqq 1$. Given two natural numbers $m' \leqq m$ and $n' \leqq n$, find a subgraph $H = (X, Y, F)$ of G such that $|X| = m'$, $|Y| = n'$ and $w(F) = \sum_{e \in F} w(e)$ is maximized.

We note the condition $0 \leqq w(e) \leqq 1$ is given since it is convenient for presenting our theoretical results, although we can define the problem without this condition.

For the dense subgraph problem, several algorithms have been proposed [6], [8], [9] in the literature. Especially, PTAS algorithms are known when the subgraph has $\Omega(n^2)$ edges, where n is the number of vertices of the original graph [6], [9]. On the other hand, it is considered to be difficult to design an efficient approximation algorithm with a good approximation ratio in general for the dense subgraph problem. Indeed, the current best approximation ratio is $cn^{-1/3}$ for a constant c, and it is conjectured that there exists some constant ϵ such that $n^{-\epsilon}$ approximation is hard [7], [9]. Here, we use a convention that an algorithm has an approximation ratio $r < 1$ for a maximization problem if its objective value is at least r times the optimal value. We also call it an r-approximation algorithm. (We remark that some papers regard r^{-1} the approximation ratio for a maximization problem.)

Although the bipartite dense subgraph problem looks easier than the general dense subgraph problem, it has not been revealed how easier it is. The problem is $\mathcal{NP}$-hard, since the $\mathcal{NP}$-hard edge-maximizing bipartite clique problem is reduced to this problem. Note that bipartite clique problem

is polynomial-time soluble if the criterion is to maximize the number of vertices. (See [10] for the complexities of bipartite clique problems). The known PTAS algorithms for the dense subgraph problem only work when the input graph is dense and the output subgraph is large and dense. However, in many applications for the bipartite dense subgraph problem, it is desired to solve the problem with only a density assumption on the optimal output graph (e.g. a near-clique condition of the output).

Our results

We consider three parameters: $\Delta = \frac{w(F)}{m'n'}$, $p = m/m'$, and $q = n/n'$, where F is the edge set of the optimal solution. We remark that Δ is the (weighted) density of the optimal output graph. Under the condition that Δ is a constant (i.e. Δ^{-1} is bounded by a given constant), we give a polynomial-time approximation algorithm with an approximation ratio $(\min(p,q))^{-\alpha}$ for any constant $\alpha > 0$.

Moreover, we give an algorithm with a complexity $O(mn2^{O(\Delta^{-1}\epsilon^{-2}\log p \log q)})$ that outputs an $(1-\epsilon)$-approximation solution; that is, it outputs $H_0 = (X_0, Y_0, F_0)$ such that $|X_0| = m'$, $|Y_0| = n'$ with an weighted density $\Delta_0 \geq (1-\epsilon)\Delta$. The time complexity implies that we have PTAS if Δ and $\min(p,q)$ are constants. Moreover, the time complexity implies that it also runs in polynomial time if (1) both p and q are constants and $\Delta = \Omega(\log^{-1} n)$, or (2) Δ is a constant and $\max(p,q) = 2^{O(\sqrt{\log mn})}$.

These results show that the bipartite dense subgraph problem has substantially efficient solutions compared to the dense subgraph problem. In particular, our algorithms give a approximate solutions for the bipartite clique problem (i.e., unweighted and $\Delta = 1$), if the measurement is done by the number of edges in the output graph.

Our algorithms are simple but they are not practically efficient if we implement them as they are. Practical impact of our results should be interpreted that we give theoretical performance gurantee for a very simple multi-start heuristics: We have an $(1-\epsilon)$-approximation solution if we perform the algorithm for $O(2^{O(\Delta^{-1}\epsilon^{-2}\log p \log q)})$ different starting instances, although we may run the same algorithm in practice with a smaller number of starting instances so that the process finishes within practical computation time.

Application to dimension-reducing cluster finding

We consider a data set S with m data and n attributes. Thus, each data is considered as an n-dimensional vector. Each attribute may be binary, categorical, or numeric. Let $Q = \prod_{i=1}^{n} M_i$ be the space of possible data vectors, where M_i is the range of the i-th attribute. If each attribute is numeric, Q is a subset of n-dimensional Euclidean space, while Q is the hypercube if each attribute is binary.

A *cluster* in S is a subset with a strong correlation between data tupples in the set. The quality of a cluster depends on its size and strength of the correlation. *Clustering* usually means partitioning of a data set into clusters. Instead of partitioning data into clusters, we focus on the *cluster finding* problem that extracts one cluster in a data. Indeed, in database applications such as data mining, it is often required to find a set of possibly overlapping clusters covering a part of whole data. Accordingly, it is important to compute each cluster independently.

In a purely geometric setting, a distance measure (e.g. Euclidean distance, Hamming distance, etc.) is given, and two data should be in a same cluster if the distance between the corresponding data vectors is small. More generally, if some metric to give similarity between data is given, there are many efficient clustering methods, for example, CLARA, CLARANCE and BIRCH [16] are such methods.

However, in data mining situation, we often consider a situation where two data are similar to each other if they are near in a (unknown) subspace of Q corresponding to a subset of attributes. In other words, dimension reduction is crucial in clustering, and we want to select a subset of attributes (i.e. dimensions) and a subset of data forming a geometric cluster projected to the subspace. Such dimensions are called *effective dimensions* for the cluster. A cluster should be reasonably large, and a cluster with a larger number of effective dimensions is stronger. Unfortunately, a similarity condition for the dimension-reducing clustering does not define a metric in the original space (i.e. triangle inequality does not hold).

If each attribute is binary or categorical, the dimension-reducing cluster-finding problem is a problem in a bipartite graph between the set U of data tupples and the set V of categories of attributes. Precisely speaking, an element of V is a pair (A, v) of attribute and its categories ($v \in \{0,1\}$ if the attribute is binary). In the graph, $u \in U$ is incident to $(A, v) \in V$ iff the attribute value $A(u)$ is v. The output cluster is represented by a subset $X \in U$ and $Y \in V$, and its quality is measured by $m' = |X|$, $n' = |Y|$, and property (e.g. density) of the induced subgraph.

It is popular to formulate the problem into a bipartite clique problem. In Agrawal-Imielinski-Swami's seminal paper [4], Y is called *item set* and $|X|/|U| = m'/m = p^{-1}$ is called the *support* of the item set, where X is the set of vertices in U that are incident to all vertices in Y. Thus, X and Y induces a clique. An item set whose support is larger than a threshold is called a *large item set*. Agrawal-Srikant [5] presents an algorithm (named Apriori) to enumerate all large item sets with a time complexity linear in the output size as well as a small number of rounds of secondary-data access. For the case of numeric attributes, the CLIQUE algorithm

given by Agrawal *et al.* [3] formulate the problem as an optimization problem by cutting the space into a n-dimensional mesh, and apply a version of Apriori algorithm to find clusters.

Since Apriori is a bottom-up enumeration algorithm, its weak point is that if there exists a high-dimensional cluster (i.e., a large item set with a large $|Y|$), the time complexity may become huge, since the number of outputs is exponential in the largest $|Y|$. Thus, the method efficiently works under a statistical assumption that a bipartite graph corresponding to a database rarely have a high-dimensional large clique (i.e. a clique with large $|X|$ and $|Y|$). In practice, we use the Apriori algorithm to enumerate item sets with small numbers of attributes, and higher dimensional clusters are constructed by using heuristic methods from those item sets. Also, there are several algorithms (e.g. [13], [15]) that enumerate maximal bipartite cliques (corresponding to *closed item sets*) to resolve the above problem moderately, although there are exponential number of maximal cliques in the worst case.

Procopiuc *et al.* [14] gave a breakthrough to design a dimension-reducing clustering (often referred to as *projective clustering*) algorithm with a theoretical complexity analysis. Their DOC (Density-based Optimal projective Clustering) algorithm is a Monte-Carlo algorithm that extracts a cluster in a numeric database in which a cluster is represented by an orthogonal region in $Q = \mathcal{R}^n$. In the heart of their algorithm, they construct a bipartite graph, and reduce the clustering problem to the bipartite clique problem. A function $f(m', n')$ on $m' = |X|$ and $n' = |Y|$ representing the quality of a cluster is considered, and the optimal cluster corresponds to the bipartite clique maximizing $f(m', n')$. DOC computes an optimal cluster for the case $f(m', n') = m'\beta^{-n'}$ for a given real number $\beta < 0.5$ under the *β-balanced condition* that assumes the clique maximizing $f(m', n')$ has more than cm elements for a given constant c (i.e., $p = m/m'$ is a bounded by the constant c^{-1}). After obtaining a cluster, the data tupples in the cluster are removed so that the next cluster can be extracted from the rest of the data set.

The time complexity of DOC is $O(m2^{O(\log p \log n \log^{-1}(1/2\beta))})$. When β is small (say, $\beta = 1/4$), it becomes $O(m2^{O(\log p \log n)})$, which resembles to ours (for the bipartite clique version). However, in our experiemce of implementation, DOC often poorly performs on a practical input: A major weak point of DOC is that it requires the β-balanced condition. Users need to find a suitable β, and it may happen that no $\beta < 0.5$ satisfies the balancing condition.

One of our motivations of this research is to remove the β-balanced condition from DOC by formulating the problem into bipartite dense subgraph problem and consider approximate solutions. For a cathegorical database, we can directly formulate the problem into the unweighted bipartite dense subgraph problem. If we consider numeric database, our cluster is geometrically a union of orthogonal regions, and we need to discretize the input to convert the geometric problem to the bipartite dense subgraph problem. This is similar to CLIQUE algoirthm [3], where an output cluster is a union of orthogonal regions in a discretized space.

We can implement our algorithm as a variation of DOC (Section 2.3) that outputs a dense subgraph (instead of a clique) and does not require β-balanced condition. It often happens that there exists a dense subgraph with large m' and n' even if there is no large bipartite clique, and thus our approach enables to obtain a high-dimensional cluster even if there is no large clique. Moreover, we can reduce effect of undefined entries and input errors that are often observed in an industrial database. Also, our $O((\min(p,q))^{-\alpha}))$- approximation algorithm can detect a small dense cluster to some extent.

2. Quasi-polynomial time approximation scheme with respect to p and q

Our main result is the following: Recall that $p = m/m'$, $q = n/n'$, and $\Delta = \frac{w(F)}{m'n'}$. Since ϵ is the error ratio of the solution, the approximation ratio is $(1-\epsilon)$.

Theorem 2.1. *We can compute an $(1-\epsilon)$-approximation solution for the bipartite dense subgraph problem in $O(nk2^{O(\Delta^{-1}\epsilon^{-2}\log p \log q)})$ randomized expected time.*

2.1 Outline of the algorithm

Technically, the algorithm is based on a framework given by Arora *et al.* [6] for solving the dense subgraph problem: That is, select a small sample set and search in its powerset to find a subset that leads an approximate solution with the desired theoretical quality.

We first formulate the problem into a quadratic integer programming problem. We write $U = \{u_1, u_2, \ldots, u_m\}$, $V = \{v_1, v_2, \ldots, v_n\}$. We introduce a matrix $W = (w_{i,j})_{1 \leq i \leq m, 1 \leq j \leq n}$ indicating the graph structure of G. For the unweighted case, $w_{i,j}$ is binary, and $w_{i,j} = 1$ if and only if $(u_i, v_j) \in E$. For the weighted problem, we regard $w_{i,j}$ as the weight of the edge (u_i, v_j). It is straightforward to see that the bipartite dense subgraph problem is equivalent to the following **QIP**.

QIP: Maximize ${}^t\mathbf{x}W\mathbf{y}$, subject to

$$\sum_{1 \leq i \leq m} x_i = m', \quad \sum_{1 \leq j \leq n} y_j = n', \text{ and } x_i, y_j \in \{0,1\}.$$

Here, ${}^t\mathbf{x}W\mathbf{y} = \sum_{1 \leq i \leq m} \sum_{1 \leq j \leq n} w_{i,j} x_i y_j$ is the matrix

product (we consider $\mathbf{x}$ and $\mathbf{y}$ as column vectors and ${}^t\mathbf{x}$ is the transpose of $\mathbf{x}$). A very useful feature is that the objective function is bilinear in x_i and y_j. We note that the integrality condition can be relaxed to $0 \leqq x_i \leqq 1$ and $0 \leqq y_j \leqq 1$ for this particular problem, although we do not use this property directly.

Let $(\mathbf{x}^{opt}, \mathbf{y}^{opt})$ be an optimal solution of **QIP** and $z^{\rm opt} = {}^t\mathbf{x}^{opt} W \mathbf{y}^{opt}$. For a given $\mathbf{x}$, we define an n-dimentional vector $\mathbf{a}(\mathbf{x}) = {}^t\mathbf{x}W$. Then, $z^{\rm opt} = \mathbf{a}(\mathbf{x}^{opt}) \cdot \mathbf{y}^{opt}$, where $\cdot$ is the inner product operation. Therefore, we can rewrite **QIP** into the following form:

QIP2 : Maximize $\mathbf{a}(\mathbf{x}) \cdot \mathbf{y}$, subject to

$$\sum_{1 \leqq i \leqq m} x_i = m', \quad \sum_{1 \leqq j \leqq n} y_j = n', \text{ and } x_i, y_j \in \{0,1\}.$$

Symmetrically, if we define $b_i(\mathbf{y}) = \sum_{1 \leqq j \leqq n} w_{i,j} y_j$ for $i = 1, 2, \ldots, m$, we have another form of **QIP** as follows:

QIP3: Maximize $\mathbf{b}(\mathbf{y}) \cdot \mathbf{x}$, subject to

$$\sum_{1 \leqq i \leqq m} x_i = m', \quad \sum_{1 \leqq j \leqq n} y_j = n', \text{ and } x_i, y_j \in \{0,1\}.$$

We consider simplified relatives of the above two formulations, where we only concentrate on one of $\mathbf{x}$ and $\mathbf{y}$ assuming $\mathbf{a}$ or $\mathbf{b}$ is given.

QIP2*: Maximize $\mathbf{a} \cdot \mathbf{y}$

Subject to $\sum_{1 \leqq j \leqq n} y_j = n'$ and $y_j \in \{0,1\}$.

QIP3*: Maximize $\mathbf{b} \cdot \mathbf{x}$

Subject to $\sum_{1 \leqq i \leqq m} x_i = m'$ and $x_i \in \{0,1\}$.

Both of **QIP2*** and **QIP3*** can be solved by greedy algorithms. Indeed, given $\mathbf{a} = (a_1, a_2, \ldots, a_n)$ (resp. $\mathbf{b} = (b_1, b_2, \ldots, b_m)$), we consider the largest n' (resp. m') entries (breaking tie arbitrary) of $\mathbf{a}$ (resp. $\mathbf{b}$), and let $J(\mathbf{a}) \subset \{1, 2, \ldots, m\}$ (resp. $J(\mathbf{b}) \subset \{1, 2, \ldots, n\}$) be the set of corresponding indices.

We define the binary vector $\mathbf{y}(\mathbf{a}) = (y_1(\mathbf{a}), y_2(\mathbf{a}), \ldots, y_n(\mathbf{a}))$ such that $y_j(\mathbf{a}) = 1$ if and only if $j \in J(\mathbf{a})$. Similarly, we define a binary vector $\mathbf{x}(\mathbf{b}) = (x_1(\mathbf{b}), x_2(\mathbf{b}), \ldots, x_m(\mathbf{b}))$ such that $x_i(\mathbf{b}) = 1$ if and only if $i \in J(\mathbf{b})$.

Lemma 2.2. *The vectors* $\mathbf{y}(\mathbf{a})$ *and* $\mathbf{x}(\mathbf{b})$ *are optimal solutions for* **QIP2*** *and* **QIP3***, *respectively.*

Now, we can design a very simple algorithm to find a feasible solution for **QIP**. Start with any nonnegative vector $\mathbf{a}^*$ and solve **QIP2*** for $\mathbf{a} = \mathbf{a}^*$. Next, using the output $\mathbf{y}^1 = \mathbf{y}(\mathbf{a}^*)$ of **QIP2***, we compute the vector $\mathbf{b}(\mathbf{y}^1) = (b_1(\mathbf{y}^1), b_2(\mathbf{y}^1), \ldots, b_m(\mathbf{y}^1))$, and solve the follwing **QIP3*** for $\mathbf{b} = \mathbf{b}(\mathbf{y}^1)$ to have an output vector $\mathbf{x}^1 = \mathbf{x}(\mathbf{b})$.

Lemma 2.3. *The pair* $(\mathbf{x}^1, \mathbf{y}^1)$ *is a feasible solution of* **QIP**. *Moreover, let* $(\mathbf{x}^0, \mathbf{y}^0)$ *be any feasible solution of* **QIP**, *and let* $(\mathbf{x}^1, \mathbf{y}^1)$ *be the vector obtained by applying the above procedure to* $\mathbf{a}(\mathbf{x}^0)$. *Then,* ${}^t\mathbf{x}^1 W \mathbf{y}^1 \geqq {}^t\mathbf{x}^0 W \mathbf{y}^0$.

Let $z^1 = {}^t\mathbf{x}^1 W \mathbf{y}^1$ be the objective function value associated with $(\mathbf{x}^1, \mathbf{y}^1)$. We compare z^1 to $z^{\rm opt}$. We first claim that if $\mathbf{a}^*$ is a good approximation of $\mathbf{a}^{opt} = \mathbf{a}(\mathbf{x}^{opt})$, then $z^{\rm opt} - z^1$ is small; in other words, $(\mathbf{x}^1, \mathbf{y}^1)$ gives a good approximation solution for **QIP**. Next, we give a method to obtain an $\mathbf{a}^*$ that approximates $\mathbf{a}^{opt}$.

Remark: Lemma 2.3 implies that we can iterate the procedure by applying it to $\mathbf{a} = \mathbf{a}(\mathbf{x}^1)$ to have a new solution $(\mathbf{x}^2, \mathbf{y}^2)$, and continue until we have no improvement. Indeed, this is a heuristic method (a kind of local search) that system engineers tend to try without considering its performance gurantee. Indeed, the performance depends on the choice of initial $\mathbf{a}$, and our results give a guiding principal to design its multi-start version that has a theoretical gurantee.

Lemma 2.4. *Let* $J = J(\mathbf{a}^{opt})$ *and* $J^* = J(\mathbf{a}^*)$. *Then,* $\sum_{j \in J^*} a_j^* \geqq \sum_{j \in J} a_j^*$ *and* $z^1 \geqq z^* = \sum_{j \in J^*} a_j^{opt}$.

Proof. The first formula is straightforward from the definition of J^*. For the second formula, $z^* = \sum_{j \in J^*} a_j^{opt} = {}^t\mathbf{x}^{opt} W \mathbf{y}^1$ is the objective function value of a feasible solution $(\mathbf{x}^{opt}, \mathbf{y}^1)$ of **QIP**. However, if we fix $\mathbf{y}^1$, $\mathbf{x}^1$ is the best possible assignment of x values. Thus, this solution cannot be better than $(\mathbf{x}^1, \mathbf{y}^1)$. □

We define $F_1(\mathbf{a}^*) = \sum_{j \in J} a_j^{opt} - \sum_{j \in J} a_j^*$ and $F_2(\mathbf{a}^*) = \sum_{j \in J^*} a_j^* - \sum_{j \in J^*} a_j^{opt}$.

Lemma 2.5. $z^{\rm opt} - z^1 \leqq F_1(\mathbf{a}^*) + F_2(\mathbf{a}^*)$.

Proof. From the previous lemma, $z^{\rm opt} - z^1 \leqq \sum_{j \in J} a_j^{opt} - \sum_{j \in j^*} a_j^{opt} \leqq \sum_{j \in J} a_j^{opt} - \sum_{j \in j^*} a_j^{opt} + \sum_{j \in J^*} a_j^* - \sum_{j \in J} a_j^*$ $= F_1(\mathbf{a}^*) + F_2(\mathbf{a}^*)$. □

Thus, in order to obtain an approximate solution whose objective function value is at least $(1 - \epsilon) z^{\rm opt}$, it suffices to find $\mathbf{a}^*$ such that

$$F_1(\mathbf{a}^*) + F_2(\mathbf{a}^*) \leqq \epsilon z^{\rm opt}.$$

Now, we consider a random sample $\Gamma \subset U$ of size $|\Gamma| = \gamma m$. We identify U and the index set $\{1, 2, \ldots, m\}$, thus we regard $\Gamma \subset \{1, 2, \ldots, m\}$. We define $\mathbf{a}(\Gamma) = (a_1(\Gamma), a_2(\Gamma), \ldots, a_n(\Gamma))$ by $a_j(\Gamma) = \gamma^{-1} h_j$. This $\mathbf{a}(\Gamma)$ is

our candidate for $\mathbf{a}^*$, and we estimate $F_1(\mathbf{a}(\Gamma)) + F_2(\mathbf{a}(\Gamma))$. Note that we are not ready to claim that we can compute $\mathbf{a}(\Gamma)$ at this stage, since we do not know $\mathbf{x}^{opt}$.

2.2 Analysis of $F_1(\mathbf{a}(\Gamma)) + F_2(\mathbf{a}(\Gamma))$

Since the corresponding terms in F_1 and F_2 cancells out for indices in $J \cap J^*$, we can remove them from the estimation. Thus, the worst case occurs when $J \cap J^* = \emptyset$, and we assume this situation without loss of generality in our analysis.

Let $h_j = \sum_{i \in \Gamma} w_{i,j} x_i^{opt}$, for each $j = 1, 2, \ldots, n$. Since $\sum_{1 \leq i \leq m} w_{i,j} x_i^{opt} = a_j^{opt}$, the expected value of h_j is

$$E(h_j) := \mu_j = \gamma a_j^{opt}.$$

Lemma 2.6. *$Pr[h_j > \mu_j + \delta] < e^{-\delta^2/(2\mu_j + \delta)}$ and $Pr[h_j < \mu_j - \delta] < e^{-\delta^2/2\mu_j}$ for each $1 \leq j \leq n$. Here, $e = 2.718\ldots$ is the base of natural logarithm.*

Proof. We take each element of U in the sample with probability γ, and the element u_i contributes by $w_{i,j} x_i^{opt}$ to h_j if it is selected. Thus, for each fixed j, we define a random variable $Z_{i,j}$ that takes $w_{i,j} x_i^{opt}$ with probability γ and 0 with probability $1 - \gamma$ for each $i = 1, 2, \ldots, m$. Note that if $w_{i,j} x_i^{opt} = 0$, $Z_{i,j} \equiv 0$. This gives a series of independent variables for each fixed j, and $h_j = \sum_{i=1}^{m} Z_{i,j}$. Thus, we apply Chernoff's bounds (See Appendix, Theorem 5.1) to obtain this lemma. □

Corollary 2.7. *If $\mu_j \leq f$, $Pr[h_j < \mu_j - rf] < e^{-r^2 f/2}$ and $Pr[h_j > \mu_j + rf] < e^{-r^2 f/(2+r)}$ for any $r > 0$.*

We utilize the following combinatorial fact:

Lemma 2.8. *Let $S = \{n_1, n_2, \ldots n_K\}$ be a set of K real numbers, and there exists an index set $I \subseteq \{1, 2, \ldots, K\}$ of size k satisfying $\sum_{i \in I} n_i \geq N$, then there exists a natural number ℓ such that at least $2^{-\ell+1}k$ entries of S exceeds $\ell N/2k$.*

Proof. Assume that on the contrary that there are less than $2^{-\ell+1}k$ entries that exceed $\ell N/2k$ for each ℓ. Then, $\sum_{n_i \in I} n_i \leq \sum_{\ell=1}^{\log k+1} (2^{-\ell+1} - 2^{-\ell}) k(\ell N/2k)$. The right hand summation is $\sum_{\ell=1}^{\log k+1} \frac{N}{2} 2^{-\ell} \ell < N$, and we have contradiction. □

First, we consider $F_2[\mathbf{a}(\Gamma)]$. Without loss of generality, we assume that $q \geq e$ so that $\ln q \geq 1$ (if $q < e$, we replace the $\ln q$ term in the following proposition by 1).

Proposition 2.9. *If $\gamma = \frac{cn' \ln q}{z^{\rm opt} \epsilon^2}$ for a sufficiently large constant c, $F_2(\mathbf{a}(\Gamma)) < \epsilon z^{\rm opt}/2$ with probability at least 0.9.*

Proof. Suppose that $F_2(\mathbf{a}(\Gamma)) \geq \epsilon z^{\rm opt}/2$. Thus, $\gamma F_2(\mathbf{a}(\Gamma)) \geq cn' \epsilon^{-1}/2$.

We apply Corollary 2.7 for $f = \gamma z^{\rm opt}/n' = c\epsilon^{-2} \ln q$ that is the average of γa_j^{opt} over $j \in J$. Recall that we can assume $J^* \cap J = \emptyset$. J is the set of indices for the n' largest elements of a_j^{opt}. Thus, $\mu_j \leq f$ if $j \notin J$, and accordingly $\mu_j \leq f$ for $j \in J^*$. Thus, $Pr[h_j > \mu_j + r\epsilon f] < e^{-cr^2 \ln q/(2+r\epsilon)} < q^{-cr^2/(2+r)}$. We define $P(r) = q^{-cr^2/(2+r)}$. Thus, the expected number of indices $j \in \{1, 2, \ldots, n\}$ such that $h_j - \mu_j > r\epsilon f$ is bounded by $nP(r)$. Hence, from Markov's inequality, the probability that there are L such indices is bounded by $nP(r)/L$.

We use Lemma 2.8 by setting $n_j = h_j - \mu_j$, $K = n$, $I = J^*$, $k = n'$ and $N = \gamma \epsilon z^{opt}/2 = cn' \epsilon^{-1} \ln q/2$.

If $F_2(\mathbf{a}(\Gamma)) < \epsilon z^{\rm opt}/2$, $\sum_{j \in J^*} n_j \geq N$. Thus, there exists an ℓ such that at least $2^{-\ell+1} n'$ entries of S exceeds $\ell N/2n' = \frac{\ell cn' \epsilon^{-1} \ln q}{4n'} \geq \frac{c\ell\epsilon^{-1} \ln q}{4} = \frac{\ell \epsilon f}{4}$. From the argument above, the probability that there are $2^{-\ell+1} n'$ such indices is bounded by $nP(\ell/4)/(2^{-\ell+1} n') = qP(\ell/4) 2^{\ell-1}$.

It is routine to show $qP(\ell/4) 2^{\ell-1} < e^{-3\ell}$ if c is sufficiently large and $q > e$. Thus, the probability that there exists at least one such ℓ is bounded by $\sum_{\ell=1}^{\log n'} e^{-3\ell} < 1/10$. Thus, the probability that $F_2(\mathbf{a}(\Gamma)) < \epsilon z^{\rm opt}/2$ is at most 0.1. □

We next consider $F_1(\mathbf{a}(\Gamma))$. Note that a factor of $\ln q$ is saved in γ in the next proposition from that in Proposition 2.9. This is overkill for our proof of Theorem thm:main, but we will utilized the saving in Section 3..

Proposition 2.10. *If $\gamma = \frac{cn'}{z^{\rm opt} \epsilon^2}$ for a sufficiently large c, $F_1(\mathbf{a}(\Gamma)) < \epsilon z^{\rm opt}/2$ with probability at least 0.9.*

Proof. We estimate $\sum_{j \in J} (\mu_j - h_j)$, which equals $\gamma F_1(\mathbf{a}(\Gamma))$. We would like to apply Corollary 2.7, but some μ_j are larger than the avarage f of μ_j over $j \in J$. However, J is independent of choice of Γ, and for each j such that $(L-1)f < \mu_j \leq Lf$, there exists a decomposition of $\{1, 2, \ldots, m\}$ into subsets $I_1, I_2, \ldots I_L$ such that the expectation of the expression $\sum_{i \in \Gamma \cup I_\ell} w_{i,j} x_i^{opt}$ is at most f for each $1 \leq \ell \leq L$. It is easy to see that we generate $n'' \leq 2n'$ such expressions in total. Thus, for analyzing $F_1(\mathbf{a}(\Gamma))$, it suffices to consider n'' linear expressions instead of original n' expressions. Let J' be the set of indices of these new expressions.

The rest of the proof is analogous to Proposition 2.9. The reason that we can save a $\ln q$ factor in the sample size is that J' is independent of choice of Γ. Thus, we can set $K = n''$ instead of $K = n$, and save $n/n'' = q/2$ factor in the probability. This enables to save a $\ln q$ factor in the sample size. □

From Proposition 2.9 and Proposition 2.10, we obtain the following:

Corollary 2.11. *If $\gamma \geq \frac{cn' \ln q}{z^{\rm opt} \epsilon^2}$ for a sufficently large constant c, $F_1(\mathbf{a}(\Gamma)) + F_2(\mathbf{a}(\Gamma)) < \epsilon z^{\rm opt}$ with probability 0.8.*

Note that $z^{\rm opt} = \Delta n'm'$. We consider a sample Γ of size

$$|\Gamma| = \gamma m = \frac{cn'm\ln q}{z^{\rm opt}\epsilon^2} = \frac{cn'm\ln q}{n'm'\Delta\epsilon^2} = c\epsilon^{-2}\Delta^{-1}p\ln q$$

suggested in the above corollary. Let $Z(\Gamma)$ be the subset of Γ defined by $Z(\Gamma) = \{i \in \Gamma | x_i^{opt} = 1\}$. $h_j = \sum_{i\in Z(\Gamma)} w_{i,j}$ and $a_j(\Gamma) = \gamma^{-1}h_j$ are computed from $Z(\Gamma)$, thus we obtain a $(1-\epsilon)$ approximation solution with probability 0.8 if we can correctly guess $Z(\Gamma)$.

The number of all subsets of Γ is $2^{O(\Delta^{-1}\epsilon^{-2}p\ln q)}$. Thus, if we exhaustively search all subsets of Γ to guess $Z(\Gamma)$, the computation time is exponential in p. Fortunately, we do not need to examine all subsets, since the expected value of the size of $Z(\Gamma)$ is $p^{-1}|\Gamma|$, and the following lemma is obtained from Chernoff's bounds.

Lemma 2.12. *The probability that* $||Z(\Gamma)| - p^{-1}|\Gamma|| > 3\sqrt{p^{-1}|\Gamma|}$ *is at most* $2e^{-2}$.

From the above lemma and Corollary 2.11, we have the following:

Corollary 2.13. *With probability* $0.8 - 2e^{-2}$, *we have a sample* Γ *such that* $||Z(\Gamma)| - p^{-1}|\Gamma|| < 3\sqrt{p^{-1}|\Gamma|}$ *and* $Z(\Gamma)$ *gives an* $(1-\epsilon)$*-approximation solution for* **QIP**.

The number of subsets of Γ whose cardinality is at most $r = p^{-1}|\Gamma| + 3\sqrt{p^{-1}|\Gamma|}$ is bounded by ${}_{|\Gamma|+r}C_r$. From the Stirling's formula, the number is asymptotically bounded by $(2\pi)^{-0.5}(e(p+1))^r$, where e is the natural base of logarithm. Since $r = O(\Delta^{-1}\epsilon^{-2}\ln q)$, we have the following:

Theorem 2.14. *We can compute an* $(1-\epsilon)$*-approximation solution for the bipartite dense subgraph problem in* $O(mn2^{O(\Delta^{-1}\epsilon^{-2}\ln p\ln q)}$ *time with high probability.*

Proof. We can enumerate all subsets of Γ whose cardinality is at most $r = p^{-1}|\Gamma| + 3\sqrt{p^{-1}|\Gamma|}$ in time that is linear in the output size. For each subset, we can compute its associated feasible solution of QIP in $O(mn)$ time. We can do sampling multiple times to increase the success probability. □

2.3 Simplified algorithm

In the above algorithm, we take a sample Γ, and exhaustively search all its subsets of size approximately $p^{-1}|\Gamma|$ to find $Z(\Gamma)$. However, it is easy to see that there exists a subset of size exactly $\lfloor p^{-1}|\Gamma|\rfloor$ instead of $Z(\Gamma)$ to give an approximation algorithm, if we allow to increase the error ratio ϵ slightly to $(1+3p^{-0.5})\epsilon$. Thus, it suffices to find a subset of size exactly $\lfloor p^{-1}|\Gamma|\rfloor$, and we can randomly generate such subsets instead of enumerating all of them. Now, instead of two-step sampling, we can randomly select subsets of size $p^{-1}\gamma m$ directly from U, and apply **QIP2*** and **QIP3***. The time complexity depends on the success probability, and our time complexity analysis given above works.

This is almost same as the strategy employed in the DOC algoirthm of [14], where (instead of our J^*) the maximum subset of V forming a bipartite clique with the sample set is reported. In other words, our algorithm can be considered as a variant of DOC such that it outputs a dense subgraph instead of a clique.

3. Approximation algorithm without size restriction

Now, let us consider the case where both p and q are large, and hence the output graph has considerably smaller number of vertices. By symmetry, we assume $p \geqq q$ without loss of generality. We give a polynomial time algorithm to give a solution with an approximation ratio $q^{-\alpha}/2$ for any $\alpha > 0$ under only the density condition $\Delta = \Omega(1)$. We assume q is sufficiently large such that $q^\alpha > 10$, since the time complexity of the algorithm in the previous section is polynomial if $q < \log_\alpha 10$.

The algorithm itself does not change from the one in the previous section except the sample size. Here, we take a sample Γ of size $\gamma m = \frac{cmn'}{z^{\rm opt}}$, where c is a constant dependent on α. In the analysis given in the previous section, the time complexity becomes $O(mn2^{O(\log p\Delta^{-1})}) = O(mnp^{O(\Delta^{-1})})$, which is polynomial even if neither p nor q is a constant.

As we have already seen in Proposition 2.10 that $F_1(\mathbf{a}(\Gamma))$ is less than $z^{\rm opt}/2$ with probability 0.9 for this sample size if c is sufficiently large. Thus, it suffices to show that $F_2(\mathbf{a}(\Gamma)) \leqq \frac{1-q^{-\alpha}}{2}z^{\rm opt}$ with a probability larger than 0.1.

If $F_2(\mathbf{a}(\Gamma)) \leqq (q^\alpha - 1)z^*$, where $z^* = \sum_{j\in J^*} a_j^{opt}$, we have either $F_2(\mathbf{a}(\Gamma)) \leqq \frac{1-q^{-\alpha}}{2}z^{\rm opt}$ or $z^{\rm opt} \leqq 2q^\alpha z^*$; Hence, we have $q^{-\alpha}/2$-approximation solution in either case, since the objective function value of our solution is at least z^*.

We divide $F_2(\mathbf{a}(\Gamma)) \leqq (q^\alpha - 1)z^*$ by z^* to have $r - 1 \leqq q^\alpha - 1$, where $r = \frac{\sum_{j\in J^*} h_j}{\sum_{j\in J^*}\mu_j}$. Thus, it is sufficient to show that the probability that $Pr[r > q^\alpha]$ is not very high.

The average of μ_j for $j \in J^*$ is $\mu = r^{-1}\gamma z^{\rm opt}/n' = r^{-1}c$. We define $\tau_j = \max\{1, \mu_j/\mu\}$.

Lemma 3.1. *Let* $\theta > e^2 \approx 7.39$. *If* $\mu_j > \mu$, $Pr[h_j > (1+\theta)\mu_j] < (\theta+1)^{-\theta r^{-1}c\tau_j/2}$. *If* $\mu_j < \mu$, $Pr[h_j > (1+\theta)\mu] < (\theta+1)^{-\theta r^{-1}c\tau_j/2}$.

Proof. Chernoff's bound gives $Pr[Z > (1+\theta)\mu_j] < \{\frac{e^\theta}{(1+\theta)^{(1+\theta)}}\}^{\mu_j}$. The right hand is bounded by $\{(\theta+1)e^{-1}\}^{-\theta\mu_j}$, which is less than $(\theta+1)^{-\theta\mu_j/2}$ since $\theta > e^2$. Thus, the first inequality holds. The second inequality also follows easily. □

Lemma 3.2. *Let* $S = \{n_1, n_2, \ldots n_K\}$ *and* $T = \{m_1, m_2, \ldots m_K\}$ *be sets of positive numbers. If there exists a subset* $I \subset \{1, 2, \ldots, K\}$ *satisfying* $\sum_{j\in I} m_j = M$ *and*

$\sum_{j\in I} n_j > tM$, *then there exists a natural number* ℓ *and a set* $I_\ell \subset \{1,2,\ldots,K\}$ *such that* $\sum_{j\in I_\ell} m_j > 2^{-\ell+1}M$ *and* $n_j \geqq \frac{tm_j}{2}$ *for each* $j \in I_\ell$.

Proof. Analogous to the proof of Lemma 2.8. □

We apply Lemma 3.2 for $m_j = \tau_j$, $n_j = (h_j - \mu_j)$, $t = \frac{(r-1)\mu}{2}$, and $I = J^*$. We remark that $n' \leqq M = \sum_{j\in J^*} \tau_j < 2n'$. Because $\sum_{j\in J^*} h_j = r\sum_{j\in J^*} \mu_j = rn'\mu$, we have $\sum_{j\in J^*}(h_j - \mu_j) = (r-1)n'\mu > (r-1)M\mu/2 = tM$.

Thus, there exists a naturl number ℓ and a set I_ℓ such that $\sum_{j\in I_\ell} \tau_j > 2^{-\ell+1}n'$ and $n_j \geqq \frac{t\ell\tau_j}{2}$ for each $j \in I_\ell$. Thus, $h_j - \mu_j \geqq \frac{\ell(r-1)\tau_j\mu}{4}$ for each $j \in I_\ell$. From Lemma 3.1, the probability P that $h_j - \mu_j \geqq \frac{\ell(r-1)\tau_j\mu}{4}$ is bounded by $(\ell(r-1)\tau_j)^{-c\ell\tau_j(r-1)/8r} < (\ell r)^{-c\ell\tau_j/9}$, if r is sufficiently large.

Now, suppose that $r > q^\alpha$, and set the constant $c = 18\alpha^{-1}$. Then, the probability P is $(\ell q)^{-2\ell\tau_j} < q^{-2\ell\tau_j}$. By using Markov's inequality, we can show that the probability of the existence of I_ℓ for a fixed ℓ is at most $nP \leqq \frac{n}{2^{-\ell+1}n'} q^{-2\ell} < q^{-\ell}$. Thus, the probability of existence of such a pair (ℓ, I_ℓ) is bounded by $\sum_{\ell=1}^{\log q} q^{-\ell}$, which is very small if q is large.

Therefore, if we take $c > 18\alpha^{-1}$, we have $r \leqq q^\alpha$ with high probability, and hence the approximation ratio of our algorithm is $q^{-\alpha}/2$.

For the bipartite clique problem, we have the following corollary:

Corollary 3.3. *If a bipartite graph* G *has a biclique* $H = (X, Y, F)$ *with* $|X| = m' = p^{-1}m$ *and* $|Y| = n' = q^{-1}n$, *we can find a dense bipartite subgraph* $H' = (X', Y', F')$ *of* G *with* $|X'| = m'$, $|Y'| = n'$ *and* $|F'| > (\min(p,q))^{-\alpha}m'n'$ *in polynomial expected time for any positive constant* α.

3.1 Application to clustering in data mining

We first consider the case where the database is categorical: That is, we have a set $V = \{u_1, u_2, \ldots u_m\}$ of m data tupples each of that has d categorical attributes $a_1, a_2, \ldots, a_d$, where a_i maps a data to a discrete set S_i. Let $A = \{a_1, a_2, \ldots, a_d\}$ be the set of attributes.

Consider the disjoint union $V = \cup S_i$, and consider a bipartite graph $G = (U, V; E)$ where $e = (u, v) \in E$ when $v \in S_i$ and $a_i(u) = v$ for a suitable $i = 1, 2, \ldots, d$.

A cluster is a subset C of U such that each pair of elements in C are similar to each other. Here, the similarity criterion depends on applications, and quality of the cluster is measured by using a measure of the similarity. We consider that two elements u and u' of U are similar to each other with respect an attribute a_i if $a_i(u) = a_i(u')$.

When we formulate a cluster by using a bipartite clique, we consider a subset D of A such that each attribute in D takes the same value on each pair of elements v and v' in C. In other words, elements of C are identical to each other if they are projected to D. The quality of the cluster is measured by a function $f(|C|, |D|)$. Thus, the problem is reduced to find the bipartite clique of G maximizing a function $f(|C|, |D|)$.

We relax the condition such that the induced subgraph $H = (C, Y, F)$ of G forms a Δ-dense subgraph; that is, $|F| \geqq \Delta|C||Y|$. The quality of the cluster is measured by a function dependent on $|C|$, $|Y|$, and Δ. We remark that it may happen that Y may contain more than one categories corresponding to an attribute, although it unlikely happens for a practical input. We denote A_Y for the set of the corresponding attributes to Y. The density condition implies that in our approximation solution each element u in the cluster satisfies $A(u) \in Y$ for at least $(1-\epsilon)\Delta|A(Y)|$ attributes A in $A(Y)$. Accordingly, each pair of elements in the cluster have the same attribute values for a large portion (at the rate of $2(1-\epsilon)\Delta - 1$) of A_Y if Δ is near to 1 and ϵ is small.

We can assign an weight $w(e)$ for each edge in E to represent strength of an attribute to control the data. Thus, the problem is formulated into the weighted bipartite dense subgraph problem. This is important if we deal with numeric attributes. For handling a numeric attribute A, a typical method is to discretize the attribute as preprocessing and transform it into a cathegorical attribute A' such that $A'(u) = f(A(u))$, where f is the discretization function. That is, we decompose the range of A into a small number of intervals (called buckets), and $A'(u)$ shows the index of the bucket containing $A(u)$. Equal size bucketting [11] and optimal quantization [2] are typical methods for the discretization.

There is information loss due to the discretization, and hence the strength of the fact that $A'(u) = v$ depends on the original value $A(u)$; indeed, the strength should be strong if $A(u)$ is in the middle of the bucket interval. Thus, it is reasonable to formulate the problem into the weighted bipartite dense subgraph problem.

We remark that in the DOC algorithm [14], discretization is done in a different way: A width parameter w is given, and the an interval of width $2w$ is given for each numeric attribute by using a random sample point u^*. Indeed, the interval has the value $A(u^*)$ as its center for an attribute A. The discretization is binary, and $A'(u) = 1$ if and only if $A(u)$ is in the interval. With a constant probability u is in the clique if the size of the clique is $\Omega(m)$, and this fact assures theoretical performance of the DOC algorithm. If we formulate the problem into the bipartite dense subgraph problem where $\Delta = 1$, we can apply this discretization method. Unfortunately, for the case where $\Delta < 1$, this discretization method cannot be adopted as it is. Although we can take multiple sample points to simulate it if $p = m/m'$ is a constant, it seems that the bucketing method is better in practice.