Theoretical Computer Science 530 (2014) 1–11

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Solving the maximum duo-preservation string mapping problem with linear programming

Wenbin Chen [a,b,c,1,*], Zhengzhang Chen [d,e], Nagiza F. Samatova [d,e], Lingxi Peng [a], Jianxiong Wang [a], Maobin Tang [a]

[a] Department of Computer Science, Guangzhou University, PR China
[b] Shanghai Key Laboratory of Intelligent Information Processing, Fudan University, PR China
[c] State Key Laboratory for Novel Software Technology, Nanjing University, PR China
[d] Computer Science Department, North Carolina State University, Raleigh, NC 27695, United States
[e] Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge, TN 37831, United States

ARTICLE INFO

Article history:
Received 27 November 2011
Received in revised form 12 February 2014
Accepted 16 February 2014
Communicated by V.Th. Paschos

Keywords:
Approximation algorithm
Constrained maximum induced subgraph problem
Duo-preservation string mapping
Linear programming
Integer programming
Randomized rounding

ABSTRACT

In this paper, we introduce the maximum duo-preservation string mapping problem (MPSM), which is complementary to the minimum common string partition problem (MCSP). When each letter occurs at most k times in any input string, the version of MPSM is called k-MPSM. In order to design approximation algorithms for MPSM, we also introduce the constrained maximum induced subgraph problem (CMIS) and the constrained minimum induced subgraph (CNIS) problem.

We show that both CMIS and CNIS are NP-complete. We also study the approximation algorithms for the restricted version of CMIS, which is called k-CMIS ($k \geqslant 2$). Using Linear Programming method, we propose an approximation algorithm for 2-CMIS with approximation ratio 2 and an approximation algorithm for k-CMIS ($k \geqslant 3$) with approximation ratio k^2. Based on approximation algorithms for k-CMIS, we get approximation algorithms for k-MPSM with the same approximation ratio.

1. Introduction

The minimum common string partition problem (MCSP) has been well-investigated as a fundamental problem in computer science [8,12]. Given two finite length strings over the finite letter alphabet, MCSP is to partition strings into identical substrings with the minimum number of partitions. MCSP is also viewed as the problem of finding a letter-preserving bijective mapping π from letters in one string A to letters in the other string B with the minimum number of breaks, where a *letter-preserving bijective mapping* π means that each letter in A is mapped into the same letter in B and the mapping is bijective, and a break is a pair of consecutive letters in A that are mapped by π to non-consecutive letters in B [12]. In a string, a pair of consecutive letters is called a *duo* [12].

As an example, let us assume that there is a letter-preserving bijective mapping π (see Fig. 1.1) between two strings $A = abcab$ and $B = ababc$. From Fig. 1.1, we can see that π has only one break: ca is a duo of A, but $\pi(c)\pi(a)$ is not a duo

* Corresponding author at: Department of Computer Science, Guangzhou University, PR China.
E-mail address: cwb802@aliyun.com (W. Chen).

[1] The material in this paper was presented in part at the 2010 International Conference on Future Information Technology [9], Changsha, PR China, December 2010.

http://dx.doi.org/10.1016/j.tcs.2014.02.017
0304-3975/

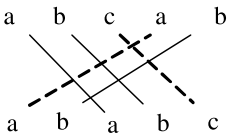

Fig. 1.1. A letter-preserving bijective mapping π for two strings: $A = abcab$, $B = ababc$.

Table 1
The approximation ratio summary for k-MCSP.

Paper	2-MCSP	3-MCSP	4-MCSP	k-MCSP
[8]	1.5			
[12]	1.1037	4		
[6]	3		$\Omega(\log n)$[a]	$O(n^{0.69})$
[18]				$O(k^2)$
[19]				$4k$

[a] It is a lower bound.

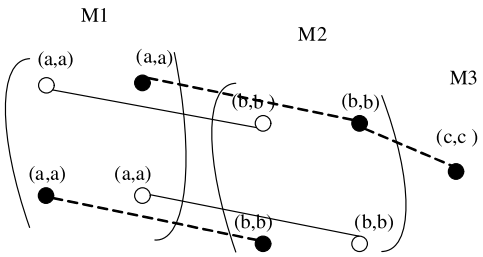

Fig. 1.2. Two strings $A = abcab$ and $B = ababc$ are transformed into a graph G_{AB}.

of B. However, the other three duos in A (ab, bc, ab) are kept by π, each of which is called duo-preservation. So, the sum of the number of breaks and the number of duo-preservations is four, which is the length of any input string minus 1.

For a letter-preserving bijective mapping between two strings, on the one hand, the optimization goal can be to minimize the number of breaks that is known as the MCSP problem. On the other hand, the optimization goal can be to maximize the number of duo-preservations. We define the maximization version of the problem as the maximum duo-preservation string mapping problem (MPSM), i.e. the problem of finding a letter-preservation bijective mapping π from one string to the other string with the maximum number of duo-preservations. When each letter occurs at most k times in any input string, the version of MPSM is called k-MPSM. The MPSM problem is complementary to the MCSP problem as shown in Section 2. From this complementary relationship, it follows that MPSM is also NP-hard since the MCSP problem is NP-hard [12].

While the MCSP problem has been widely studied, to the best of our knowledge, the MPSM problem has not been addressed before. Specifically, various approximation algorithms have been proposed to solve the k-MCSP problem, a version of MCSP, where each letter appears at most k times in any input string. These results are surveyed in Table 1.

Although there are approximation algorithms for MCSP, it is still required to design approximation algorithms for MPSM, because a pair of complementary NP-hard problems may have different approximation cases, i.e. an approximation algorithm for one problem sometimes cannot be used to approximate its complementary problem. For example, the minimum vertex cover problem and the maximum independent set problem are two well-known complementary problems in computer science [11]. For a given graph with n vertices, the minimum vertex cover problem can be approximated within a ratio of 2, but the maximum independent set is NP-hard to approximate within a factor n^{δ}, for some $\delta > 0$ [10,4,3]. Another pair of complementary problems is the Max-Satisfy problem and the Min-Unsatisfy problem [1,2]. Both the Max-Satisfy problem and the Min-Unsatisfy problem are NP-hard, but their approximation cases are also different. For a system of m linear equations with n variables over fractional numbers $\mathbf{Q}$, the Min-Unsatisfy problem can be approximated within a factor of $m+1$, but the Max-Satisfy problem is NP-hard to approximate within a factor of n^{δ}, for some $\delta > 0$ [1,2].

We notice that the MPSM problem can be transformed to a graph optimization problem. We use the example in Fig. 1.1 to explain it. For two strings $A = abcab$ and $B = ababc$ in Fig. 1.1, we can construct a graph G_{AB} as follows (see Fig. 1.2). G_{AB} has three parts M_1, M_2, and M_3. Part M_1 contains four (a, a) nodes. Part M_2 contains four (b, b) nodes, and part M_3 contains one (c, c) node. In G_{AB}, there is an edge between (a, a) node at the position $(1, 1)$ of M_1 and (b, b) node at the position $(1, 1)$ of M_2, because the first a and the first b in A form a duo ab and the first a and the first b in B also form a duo ab. Other edges are similarly constructed.

In the graph G_{AB}, the five black nodes are chosen from different rows and different columns in each M_i part, respectively, because the subgraph induced by these five nodes has the maximum edge number of 3 (dashed lines in Fig. 1.2).

From the chosen five black nodes, we can get the letter-preserving bijective mapping π with the maximum number of duo-preservations in Fig. 1.1.

In order to study approximation algorithms for the MPSM problem, we introduce the following graph optimization problem.

Problem 1.1 *(CMIS and CNIS).* Given an m-partite graph G with m parts: $M_1, \ldots, M_m$, where each M_i has $n_i \times n_i$ vertices and all the vertices are put in an $n_i \times n_i$ matrix, the goal of the constrained maximum induced subgraph problem (CMIS) is to find n_i vertices from each part M_i, where n_i vertices are from different rows and different columns, such that the induced subgraph has the maximum number of edges. If all $n_i \leqslant k$, the restricted version is called k-CMIS. On the other hand, we define the minimization version as the constrained minimum induced subgraph (CNIS) problem in an m-partite graph.

The *contributions* of the paper are as follows:

1. We prove that CMIS and CNIS are NP-complete;
2. Based on the randomized rounding technology, we propose an approximation algorithm for 2-CMIS with approximation ratio 2;
3. We propose an approximation algorithm for k-CMIS ($k \geqslant 3$) with approximation ratio k^2; and
4. We give a polynomial time reduction from the MPSM problem to the CMIS problem. Based on approximation algorithms for k-CMIS, we get approximation algorithms for k-MPSM with the same approximation ratio;
5. We give a polynomial time reduction from the minimum common string partition problem (MCSP) to the CNIS problem.

Note. Without loss of generality, in this paper we assume that the strings do not contain two consecutive occurrences of the same letter. In the case where two consecutive occurrences of the same letter, the reduced graph optimization problems are CMIS and CNIS problems with edges exist inside the same parts. For CMIS and CNIS problems with edges exist inside the same parts, the approximations results in Section 4 remain correct since in the proofs Section 4 don't use the condition $r \neq s$ for $(v_r^{ip}, v_s^{jq}) \in E$ which implies that edges can exist inside the same parts.

2. Preliminaries

In this section, we first reproduce some formal notations and definitions (Definitions 2.1–2.6) about the MCSP problem from [12], then describe the formal problem statement of the MPSM problem. Since our goal is to design a randomized approximation algorithm based on the randomized rounding technology, the definition of approximation ratio of a randomized approximation algorithm is introduced.

Definition 2.1 *(Duo).* A duo is an ordered pair of letters that occur consecutively in a string [12].

Definition 2.2 *(Partition).* A *partition* of a string A is a sequence of strings $\mathcal{P} = (P_1, P_2, \ldots, P_m)$ whose concatenation is equal to A, that is $P_1 \cdots P_m = A$, where the strings P_i $(1 \leqslant i \leqslant m)$ are called the *blocks* of $\mathcal{P}$ and m is called the number of blocks [12].

Definition 2.3 *(Common partition).* Given a partition $\mathcal{P} = (P_1, P_2, \ldots, P_m)$ of a string $A = a_1, \ldots, a_n$ and a partition $\mathcal{Q} = (Q_1, \ldots, Q_m)$ of a string $B = b_1, \ldots, b_n$, we say that the pair $\langle \mathcal{P}, \mathcal{Q} \rangle$ is a common partition π of A and B if $\mathcal{Q}$ is a permutation of $\mathcal{P}$. The common partition π can be naturally interpreted as a bijective mapping from A to B, such that, for each j $(1 \leqslant j \leqslant m)$, the letters from P_j are mapped from left to right to the corresponding letters from $Q_{j'}$ $(1 \leqslant j' \leqslant m)$ [12].

Definition 2.4 *(Break).* A break is a pair of letters that are consecutive in string A but are mapped by π to letters that are not consecutive in string B. Obviously, the block number of a partition is equal to its break number plus 1 [12].

Definition 2.5 *(Letter-preserving bijective mapping).* A *letter-preserving bijective mapping* is a bijective mapping π from letters of string A to letters of the other string B such that any letter in A is mapped into the same letter in B [12]. Duo-preservation means that a duo of A is kept by π in B.

Problem 2.1 *(Minimum common string partition problem).* The minimum common string partition problem (MCSP) is to find a common partition of two strings A and B with the minimum number of blocks. MCSP is also viewed as the problem of finding a letter-preserving bijective mapping from letters in one string to letters in the other string with the minimum number of breaks. The restricted version of MCSP, where each letter occurs at most k times in each input string, is denoted by k-MCSP [12].

Definition 2.6 *(Related strings).* Two strings A and B are related if every letter appears the same number of times in A and B [12].

Obviously, two strings have a common partition iff they are related [12].

Lemma 2.7. *Two strings A and B have a letter-preserving bijective mapping iff they are related.*

Proof. From the definition of the letter-preserving bijective mapping, there is a one-to-one correspondence between letters in A and B. Thus, every letter appears the same number of times in A and B. Hence, A and B are related. □

Problem 2.2 *(MPSM).* The maximum duo-preservation string mapping problem (MPSM) is the problem of finding a letter-preserving bijective mapping π from string A to string B with the maximum number of duo-preservations, where A and B have the same length. The restricted version of *MPSM*, where each letter occurs at most k times in each input string, is denoted by k-MPSM.

Note that two input strings are related in the MPSM problem.

Theorem 2.8. *MPSM and MCSP are complementary.*

Proof. Suppose π is the letter-preserving bijective mapping between two strings A and B in MPSM and MCSP. Then, the number of duos of A is $n-1$. Let n_b be the number of breaks and n_d be the number of duo-preservations. Thus, $n_b + n_d = n-1$. So, the MPSM problem is complementary to the MCSP problem. □

In the following, we give an example.

In Fig. 1.1, the letter-preserving bijective mapping π' has a break: ca is a duo, but $\pi'(c)\pi'(a)$ is not a duo. Thus, the number of blocks is two. However, for other three duos: ab, bc, ab, they are kept by π'. So, the sum of the number of blocks and the number of pairs of consecutive letters that are kept by π' is five, which is the length of one of input strings.

Definition 2.9 *(Approximation ratio).* Let $OPT(I)$ denote the optimum solution for an instance I of the maximization problem. Let $\tilde{E}(R(I))$ denote the expected value of the output solution $R(I)$ of a randomized approximation algorithm R. For some $r \geqslant 1$, if $\frac{OPT(I)}{\tilde{E}(R(I))} \leqslant r$, for any instance I, then the randomized approximation algorithm R is called the algorithm of approximation ratio of r [16].

3. The CMIS and CNIS problem are NP-complete

In this section, we prove that the CMIS and CNIS problems are NP-complete. We give a polynomial time reduction from the MPSM problem to the CMIS problem.

Lemma 3.1. *There exist a polynomial time reduction from the MPSM problem to the CMIS problem*

Proof. Given two related strings $X = x_1x_2\ldots x_n$ and $Y = y_1y_2\ldots y_n$, let m be the number of different letters in X and $S = \{a_1, \ldots, a_m\}$ be the unduplicated letter set. Let n_i be the number of appearances of letter a_i in X. Thus, $n_1 + \cdots + n_m = n$. In the following, we construct an instance G_{XY} of CMIS, which has m parts. For each a_i, we construct one part M_i, which has $n_i \times n_i$ nodes. Let $\langle a_i^{11}, a_i^{12}, \ldots, a_i^{1n_i}\rangle$ be all a_is in X by their appearance order. Let $\langle a_i^{21}, a_i^{22}, \ldots, a_i^{2n_i}\rangle$ be all a_is in Y by their appearance order. For each a_i^{1h} and $a_i^{2\ell}$ $(1 \leqslant h, \ell \leqslant n_i)$, we construct one node $(a_i^{1h}, a_i^{2\ell})$ in the h-th row and ℓ-th column of M_i. Edges only exist between nodes from different parts. There is an edge between $(a_i^{1h}, a_i^{2\ell})$ and (a_j^{1r}, a_j^{2s}) iff $a_i^{1h}a_j^{1r}$ is a duo in X and $a_i^{2\ell}a_j^{2s}$ is a duo in Y. For the graph G_{XY}, the goal of the CMIS problem is to find n_i vertices at different rows and different columns from each part M_i such that the subgraph induced by the chosen n nodes $(n = n_1 + \cdots + n_m)$ has the maximum number of edges.

The number of vertices in G_{XY} is $n_1^2 + n_2^2 + \cdots + n_m^2$ which is $O(n^2)$. Thus, the number of edges in G_{XY} is at most $O(n^4)$. So, the reduction is of polynomial time complexity. □

Fig. 1.2 is a reduction example. The following Theorem 3.2 shows that the MPSM problem for strings X and Y is related to the CMIS problem in the graph G_{XY}.

Theorem 3.2. *For the graph G_{XY}, an induced subgraph by n nodes, of which n_i nodes are chosen from different rows and different columns in each M_i part respectively, has the maximum number of edges iff X and Y have a bijective mapping with the maximum number of duo-preservations.*

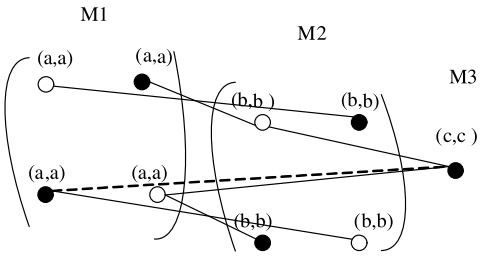

Fig. 3.3. Two strings $A = abcab$ and $B = ababc$ are transformed into a graph $\bar{G}_{AB}$.

Proof. Since each node in the graph G_{XY} denotes that a letter in X is mapped to a letter in Y, these n nodes denote a bijective mapping π from X to Y. Since each edge in G_{XY} denotes a duo-preservation, the number of edges in the induced subgraph by these n nodes is the number of duo-preservations in the bijective mapping π. Thus, the maximum number of edges in the induced subgraph by these n nodes is the maximum number of duo-preservations in a bijective mapping. The other direction of the theorem is trivial. □

Thus, by Theorem 3.2 an approximation algorithm for the MPSM problem can be achieved by designing an approximation algorithm for the CMIS problem with the same approximation ratio. If each letter in X and Y appears at most k times, then each part in G_{XY} has at most $k \cdot k$ nodes by the above reduction process. Thus, the k-MPSM problem can be reduced to the k-CMIS problem with the same approximation ratio.

On the other hand, because the MPSM is NP-hard and it is obvious that CMIS is in NP, we get the following conclusion.

Theorem 3.3. *The CMIS problem is NP-complete.*

The CMIS problem and the CNIS problem are complementary, we get the following conclusion.

Theorem 3.4. *The CNIS problem is NP-complete.*

It is easy to know that if we modify the reduction process of Lemma 3.1, we can get a reduction from the MCSP problem to the constrained minimum induced subgraph (CNIS) problem. In the reduction process of Lemma 3.1, we construct the graph $\bar{G}_{XY}$, which has the same vertices of G_{XY}, but the construction of edges is modified as follows: there is an edge between $(a_i^{1h}, a_i^{2\ell})$ and (a_j^{1r}, a_j^{2s}) iff $a_i^{1h}a_j^{1r}$ is a *duo* in X and $a_i^{2\ell}a_j^{2s}$ is *not a duo* in Y. Then, the modified reduction is a reduction from the MCSP problem to the constrained minimum induced subgraph (CNIS) problem. Thus, we get the following conclusion.

Lemma 3.5. *There exists a polynomial time reduction from the MCSP problem to the constrained minimum induced subgraph (CNIS) problem.*

For example, for two strings $A = abcab$ and $B = ababc$, we can construct a graph $\bar{G}_{AB}$ as follows (see Fig. 3.3).

In the graph $\bar{G}_{AB}$, the subgraph induced by the five black nodes has the minimum edge number of 1 (dashed line in Fig. 3.3).

The following conclusion shows that the MCSP problem for strings X and Y is related to the CNIS problem in the graph $\bar{G}_{XY}$.

Theorem 3.6. *For the graph $\bar{G}_{XY}$, a subgraph induced by n nodes, of which n_i nodes are chosen from different rows and different columns in each M_i part respectively, has the minimum number of edges iff X and Y have a bijective mapping with the minimum number of breaks.*

Proof. Since each node in the graph $\bar{G}_{XY}$ denotes that a letter in X is mapped to a letter in Y, these n nodes denote a bijective mapping π from X to Y. Since each edge in $\bar{G}_{XY}$ must produce a break, the number of edges in the induced subgraph by n nodes is the number of breaks in the bijective mapping π. Thus, the minimum number of edges in the subgraph induced by n nodes is the minimum number of breaks in a bijective mapping. The other direction of the theorem is trivial. □

4. Approximation algorithms for k-CMIS ($k \geqslant 2$)

In order to study approximation algorithms, Raghavan and Thompson introduced a randomized rounding method in [21]. Since then, the randomized rounding method has been widely used to design approximation algorithms for many NP-hard problems ([5,15,22,20,13,14,16,17], etc.). The general idea behind the randomized rounding method is: (1) An NP-hard problem is first transformed into a 0–1 Integer Programming (IP) problem. (2) Then, it is relaxed to a Linear Programming (LP) problem. (3) For the optimal solution to LP, the value of each variable is rounded to 0 or 1 by some specific method. Thus, one approximation solution to some specific NP-hard problem can be achieved if the challenges of steps (1) and (3) can be overcome.

In this section, we will design approximation algorithms for the CMIS problem based on the Linear Programming (LP) technology.

First, we give the 0–1 Integer Programming (IP) formulation for the CMIS problem. Suppose G is an m-partite graph with m parts: $M_1, \ldots, M_m$, where the vertices in each M_i are put in $n_i \times n_i$ matrix. For each node v_r^{ip} $(1 \leqslant i \leqslant n_r,\ 1 \leqslant p \leqslant n_r)$ in M_r $(1 \leqslant r \leqslant m)$, where v_r^{ip} is at the i-th row and p-th column, let x_r^{ip} be a 0–1 decision variable, that $x_r^{ip} = 1$ means that v_r^{ip} is chosen, otherwise v_r^{ip} is not chosen. Thus, the 0–1 IP formulation for the CMIS problem is as follows:

$$
\begin{array}{lll}
\text{Maximize} & \displaystyle\sum_{(v_r^{ip}, v_s^{jq}) \in E} x_r^{ip} x_s^{jq} & (IP_1) \\
\text{subject to} & \displaystyle\sum_{i=1}^{n_r} x_r^{ip} = 1, \quad \text{for } r = 1, \ldots, m & (1) \\
& \displaystyle\sum_{p=1}^{n_r} x_r^{ip} = 1, \quad \text{for } r = 1, \ldots, m & (2) \\
& x_r^{ip} \in \{0, 1\}, \quad \text{for } r = 1, \ldots, m &
\end{array}
$$

Constraints (1) and (2) guarantee that only the nodes at different rows and different columns from each part are chosen. For example, for the graph G_{AB} in Fig. 1.2, we have the following 0–1 IP formulation:

$$
\begin{array}{ll}
\text{Maximize} & A_{11}B_{11} + A_{12}B_{12} + B_{12}C + A_{21}B_{21} + A_{22}B_{22} \\
\text{subject to} & A_{11} + A_{12} = 1; \quad A_{21} + A_{22} = 1; \\
& A_{11} + A_{21} = 1; \quad A_{12} + A_{22} = 1; \\
& B_{11} + B_{12} = 1; \quad B_{21} + B_{22} = 1; \\
& B_{11} + B_{21} = 1; \quad B_{12} + B_{22} = 1; \quad C = 1; \\
& A_{ij}, B_{ij} \in \{0, 1\}, \quad \text{where } i = 1 \text{ or } 2,\ j = 1 \text{ or } 2
\end{array}
$$

Where A_{ij} $(1 \leqslant i, j \leqslant 2)$ is the decision variable corresponding to the node at the i-th row and j-th column in M_1, B_{ij} $(1 \leqslant i, j \leqslant 2)$ is the decision variable corresponding to the node at the i-th row and j-th column in M_2, and C is the decision variable corresponding to the node in M_3.

Using the common relaxation method for IP formulation [16], we build the following LP formulation for the CMIS problem:

$$
\begin{array}{lll}
\text{Maximize} & \displaystyle\sum_{(v_r^{ip}, v_s^{jq}) \in E} z_{r_{ip}s_{jq}} & (LP_1) \\
\text{subject to} & z_{r_{ip}s_{jq}} \leqslant x_r^{ip}, \quad \text{for all } r & (3) \\
& z_{r_{ip}s_{jq}} \leqslant x_s^{jq}, \quad \text{for all } r & (4) \\
& \displaystyle\sum_{i=1}^{n_r} x_r^{ip} = 1, \quad \text{for } r = 1, \ldots, m & (3') \\
& \displaystyle\sum_{p=1}^{n_r} x_r^{ip} = 1, \quad \text{for } r = 1, \ldots, m & (4') \\
& 0 \leqslant z_{r_{ip}s_{jq}} \leqslant 1, \quad \text{for all } r & \\
& 0 \leqslant x_r^{ip} \leqslant 1, \quad \text{for all } r & \\
& 0 \leqslant x_s^{jq} \leqslant 1, \quad \text{for all } r &
\end{array}
$$

For example, the LP_1 formulation for the graph G_{AB} in Fig. 1.2 is as follows:

$$
\begin{array}{lll}
\text{Maximize} & z_1+z_2+z_3+z_4+z_5 & \\
\text{subject to} & z_1 \leqslant A_{11}, \quad z_1 \leqslant B_{11}; & \\
& z_2 \leqslant A_{12}, \quad z_2 \leqslant B_{12}; & \\
& z_3 \leqslant B_{12}, \quad z_3 \leqslant C; & \\
& z_4 \leqslant A_{21}, \quad z_4 \leqslant B_{21}; & \\
& z_5 \leqslant A_{22}, \quad z_5 \leqslant B_{22}; & \\
& A_{11}+A_{12}=1; \quad A_{21}+A_{22}=1; & \\
& A_{11}+A_{21}=1; \quad A_{12}+A_{22}=1; & \\
& B_{11}+B_{12}=1; \quad B_{21}+B_{22}=1; & \\
& B_{11}+B_{21}=1; \quad B_{12}+B_{22}=1; & C=1; \\
& A_{ij}, B_{ij}, z_r \in [0,1] \quad \text{for all } i,\ j,\ r &
\end{array}
$$

First, we design the approximation algorithm for the 2-CMIS problem based on its LP_1 formulation (see Algorithm 4.1):

1: By solving the Linear Programming formulation LP_1 for the 2-CMIS problem, we get an optimum solution x_r^{ip*}, $z^*_{r_{ip}s_{jq}}$ for all r.
2: Randomized rounding:
3: **for** $r=1$ to m **do**
4: **if** $n_r=2$ **then**
5: Let $X_r=\begin{pmatrix} x_r^{11*} & x_r^{12*} \\ x_r^{21*} & x_r^{22*} \end{pmatrix}$. Let $\overrightarrow{Y_1}=(x_r^{11*},x_r^{22*})$ and $\overrightarrow{Y_2}=(x_r^{12*},x_r^{21*})$
6: $\overrightarrow{Y_i}$ is chosen with probability $\frac{\sqrt{x_r^{1i*}}}{\sqrt{x_r^{11*}}+\sqrt{x_r^{12*}}}(i=1,2)$.
7: When $\overrightarrow{Y_1}$ is chosen, we set $x_r^{11}=x_r^{22}=1$ and $x_r^{12}=x_r^{21}=0$; When $\overrightarrow{Y_2}$ is chosen, we set $x_r^{12}=x_r^{21}=1$ and $x_r^{11}=x_r^{22}=0$.
8: **end if**
9: **if** $n_r=1$ **then**
10: We set $x_r^{11}=1$.
11: **end if**
12: **end for**
13: Output those nodes whose variables are set to 1. Let $z'_{r_{ip}s_{jq}}=x_r^{ip}x_s^{jq}$. The edge number of induced subgraph by these output nodes is $\sum_{(v_r^{ip},v_s^{jq})\in E} z'_{r_{ip}s_{jq}}$.

Algorithm 4.1: The approximation algorithm for the 2-CMIS problem.

It is known that Linear Programming can be solved in a polynomial time (see [7]). Thus, Algorithm 4.1 is of polynomial time complexity.

Theorem 4.1. *Algorithm 4.1 is of expected approximation ratio 2 for 2-CMIS.*

Proof. Let $Pr(X)$ denote the probability of the event X. Let $\tilde{E}(X)$ denote the expected value of the event X. For each r, let A_j denote the event that $\overrightarrow{Y_j}$ is chosen $(j=1,2)$.

By the constraints (3) and (4), we get $z^*_{r_{ip}s_{jq}}=\min\{x_r^{ip*},x_s^{jq*}\}$. By the constraints (3′) and (4′), we get $x_r^{11*}+x_r^{12*}=1$, $x_r^{11*}+x_r^{21*}=1$ and $x_r^{12*}+x_r^{22*}=1$. So, we get $x_r^{11*}=x_r^{22*}$ and $x_r^{12*}=x_r^{21*}$.

Thus, for any i,p $(i,p=1 \text{ or } 2)$, when $i=p$, $Pr(x_r^{ip}=1)=Pr(A_1)=\frac{\sqrt{x_r^{11*}}}{\sqrt{x_r^{11*}}+\sqrt{x_r^{12*}}}=\frac{\sqrt{x_r^{ip*}}}{\sqrt{x_r^{11*}}+\sqrt{x_r^{12*}}}$; when $i\neq p$, $Pr(x_r^{ip}=1)=Pr(A_2)=\frac{\sqrt{x_r^{12*}}}{\sqrt{x_r^{11*}}+\sqrt{x_r^{12*}}}=\frac{\sqrt{x_r^{ip*}}}{\sqrt{x_r^{11*}}+\sqrt{x_r^{12*}}}$. So, in any case, $Pr(x_r^{ip}=1)=\frac{\sqrt{x_r^{ip*}}}{\sqrt{x_r^{11*}}+\sqrt{x_r^{12*}}}$.

Since $\frac{\sqrt{x_r^{11*}}+\sqrt{x_r^{12*}}}{2}\leqslant\sqrt{\frac{x_r^{11*}+x_r^{12*}}{2}}\leqslant\sqrt{\frac{1}{2}}$, we get $\sqrt{x_r^{11*}}+\sqrt{x_r^{12*}}\leqslant\sqrt{2}$. Thus $Pr(x_r^{ip}=1)\geqslant\frac{1}{\sqrt{2}}\sqrt{x_r^{ip*}}$. Similarly, we can get $Pr(x_s^{jq}=1)\geqslant\frac{1}{\sqrt{2}}\sqrt{x_s^{jq*}}$.

$$
\begin{aligned}
\text{So} \quad Pr\big(z'_{r_{ip}s_{jq}}=1\big) &= Pr\big(x_r^{ip}x_s^{jq}=1\big) \\
&= Pr\big(x_r^{ip}=1\big)Pr\big(x_s^{jq}=1\big) \\
&\geqslant \frac{1}{2}\sqrt{x_r^{ip*}}\sqrt{x_s^{jq*}}
\end{aligned}
$$

$$\geqslant \frac{1}{2}\min\{x_r^{ip*}, x_s^{jq*}\}$$
$$= \frac{1}{2}z^*_{r_{ip}s_{jq}}.$$

For any instance I, let $A(I)$ denote the output solution of the approximation algorithm. Let $OPT(I)$ denote the optimum solution. Let $OPT(IP_1)$ denote the optimum solution of IP_1 formulation for I. Let $OPT(LP_1)$ denote the optimum solution of LP_1 formulation for I.

$$\begin{aligned}
\text{So}\quad \tilde{E}\big(A(I)\big) &= \tilde{E}\Bigg(\sum_{(v_r^{ip}, v_s^{jq})\in E} z'_{r_{ip}s_{jq}}\Bigg)\\
&= \sum_{(v_r^{ip}, v_s^{jq})\in E} \tilde{E}\big(z'_{r_{ip}s_{jq}}\big)\\
&= \sum_{(v_r^{ip}, v_s^{jq})\in E} Pr\big(z'_{r_{ip}s_{jq}} = 1\big)\\
&\geqslant \frac{1}{2}\sum_{(v_r^{ip}, v_s^{jq})\in E} z^*_{r_{ip}s_{jq}}\\
&= \frac{1}{2}\cdot OPT(LP_1).
\end{aligned}$$

Thus $\frac{OPT(I)}{\tilde{E}(A(I))} = \frac{OPT(IP_1)}{\tilde{E}(A(I))} \leqslant \frac{OPT(LP_1)}{\tilde{E}(A(I))} \leqslant 2$.
Hence, Algorithm 4.1 is of approximation ratio 2 for 2-CMIS. □

Since the 2-MPSM problem can be reduced to 2-CMIS problem, an approximation algorithm for the 2-MPSM problem can be achieved with the same approximation ratio as Algorithm 4.1. Thus, we can get the following conclusion.

Corollary 4.2. *There is an approximation algorithm with expected approximation ratio* 2 *for* 2*-MPSM.*

Second, we design the approximation algorithm for the 3-CMIS problem based on its LP_1 formulation (see Algorithm 4.2):

1: By solving the Linear Programming formulation LP_1 for the 3-CMIS problem, we get an optimum solution x_r^{ip*}, $z^*_{r_{ip}s_{jq}}$ for all r.
2: Randomized rounding:
3: **for** $r = 1$ to m **do**
4: **if** $n_r = 3$ **then**
5: Let $X_r = \begin{pmatrix} x_r^{11*} & x_r^{12*} & x_r^{13*} \\ x_r^{21*} & x_r^{22*} & x_r^{23*} \\ x_r^{31*} & x_r^{32*} & x_r^{33*} \end{pmatrix}$.
6: Let $\overrightarrow{Y_1} = (x_r^{11*}, x_r^{22*}, x_r^{33*})$, $\overrightarrow{Y_2} = (x_r^{12*}, x_r^{23*}, x_r^{31*})$ and $\overrightarrow{Y_3} = (x_r^{13*}, x_r^{21*}, x_r^{32*})$.
7: Let $S_1 = x_r^{11*} + x_r^{22*} + x_r^{33*}$, $S_2 = x_r^{12*} + x_r^{23*} + x_r^{31*}$ and $S_3 = x_r^{13*} + x_r^{21*} + x_r^{32*}$.
8: $\overrightarrow{Y_i}$ is chosen with probability $\frac{\sqrt{S_i}}{\sqrt{S_1}+\sqrt{S_2}+\sqrt{S_3}}(i = 1, 2, 3)$.
9: When $\overrightarrow{Y_1}$ is chosen, we set $x_r^{11} = x_r^{22} = x_r^{33} = 1$; When $\overrightarrow{Y_2}$ is chosen, we set $x_r^{12} = x_r^{23} = x_r^{31} = 1$; When $\overrightarrow{Y_3}$ is chosen, we set $x_r^{13} = x_r^{21} = x_r^{32} = 1$;
10: **end if**
11: **if** $n_r = 2$ **then**
12: We set $x_r^{ip} = 1$ by the method in Algorithm 4.1.
13: **end if**
14: **if** $n_r = 1$ **then**
15: We set $x_r^{11} = 1$.
16: **end if**
17: **end for**
18: Output those nodes whose variables are set to 1. Let $z'_{r_{ip}s_{jq}} = x_r^{ip}x_s^{jq}$. The edge number of induced subgraph by these output nodes is $\sum_{(v_r^{ip}, v_s^{jq})\in E} z'_{r_{ip}s_{jq}}$.

Algorithm 4.2: The approximation algorithm for the 3-CMIS problem.

It is known that Linear Programming can be solved in a polynomial time (see [7]). Thus, Algorithm 4.2 is of polynomial time complexity.

Theorem 4.3. *Algorithm 4.2 is of expected approximation ratio 9 for 3-CMIS.*

Proof. Let $Pr(X)$ denote the probability of the event X. Let $\tilde{E}(X)$ denote the expected value of the event X. For each r, let A_j denote the event that $\overrightarrow{Y_j}$ is chosen ($j = 1, 2, 3$).

By the constraints (3) and (4), we get $z^*_{r_{ip}s_{jq}} = \min\{x_r^{ip*}, x_s^{jq*}\}$. By the constraints (3′) and (4′), we get $S_1 + S_2 + S_3 = x_r^{11*} + x_r^{12*} + x_r^{13*} + x_r^{21*} + x_r^{22*} + x_r^{23*} + x_r^{31*} + x_r^{32*} + x_r^{33*} = 3$.

When $n_r = 3$, for any i, p, if $x_r^{ip*} \in \overrightarrow{Y_j}$, then $Pr(x_r^{ip} = 1) = Pr(A_j) = \frac{\sqrt{S_j}}{\sqrt{S_1}+\sqrt{S_2}+\sqrt{S_3}} \geqslant \frac{\sqrt{x_r^{ip*}}}{\sqrt{S_1}+\sqrt{S_2}+\sqrt{S_3}}$. Since $\frac{\sqrt{S_1}+\sqrt{S_2}+\sqrt{S_3}}{3} \leqslant \sqrt{\frac{S_1+S_2+S_3}{3}} = \sqrt{\frac{3}{3}} = 1$, we get $\sqrt{S_1} + \sqrt{S_2} + \sqrt{S_3} \leqslant 3$. Thus $Pr(x_r^{ip} = 1) \geqslant \frac{1}{3}\sqrt{x_r^{ip*}}$. When $n_r = 2$, by the proof of Theorem 4.1, for any i, p, we have $Pr(x_r^{ip} = 1) \geqslant \frac{1}{\sqrt{2}}\sqrt{x_r^{ip*}} \geqslant \frac{1}{3}\sqrt{x_r^{ip*}}$. Thus, in any case, we get $Pr(x_r^{ip} = 1) \geqslant \frac{1}{3}\sqrt{x_r^{ip*}}$.

Similarly, we can get $Pr(x_s^{jq} = 1) \geqslant \frac{1}{3}\sqrt{x_s^{jq*}}$.

$$\begin{aligned}
\text{So}\quad Pr\big(z'_{r_{ip}s_{jq}} = 1\big) &= Pr\big(x_r^{ip} x_s^{jq} = 1\big) \\
&= Pr\big(x_r^{ip} = 1\big) Pr\big(x_s^{jq} = 1\big) \\
&\geqslant \frac{1}{9}\sqrt{x_r^{ip*}}\sqrt{x_s^{jq*}} \\
&\geqslant \frac{1}{9}\min\big\{x_r^{ip*}, x_s^{jq*}\big\} \\
&= \frac{1}{9} z^*_{r_{ip}s_{jq}}.
\end{aligned}$$

For any instance I, let $A(I)$ denote the output solution of the approximation algorithm. Let $OPT(I)$ denote the optimum solution. Let $OPT(IP_1)$ denote the optimum solution of IP_1 formulation for I. Let $OPT(LP_1)$ denote the optimum solution of LP_1 formulation for I.

$$\begin{aligned}
\text{So}\quad \tilde{E}\big(A(I)\big) &= \tilde{E}\Big(\sum_{(v_r^{ip}, v_s^{jq}) \in E} z'_{r_{ip}s_{jq}}\Big) \\
&= \sum_{(v_r^{ip}, v_s^{jq}) \in E} \tilde{E}\big(z'_{r_{ip}s_{jq}}\big) \\
&= \sum_{(v_r^{ip}, v_s^{jq}) \in E} Pr\big(z'_{r_{ip}s_{jq}} = 1\big) \\
&\geqslant \frac{1}{9} \sum_{(v_r^{ip}, v_s^{jq}) \in E} z^*_{r_{ip}s_{jq}} \\
&= \frac{1}{9} \cdot OPT(LP_1).
\end{aligned}$$

Thus $\frac{OPT(I)}{\tilde{E}(A(I))} = \frac{OPT(IP_1)}{\tilde{E}(A(I))} \leqslant \frac{OPT(LP_1)}{\tilde{E}(A(I))} \leqslant 9$.
Hence, Algorithm 4.2 is of approximation ratio 9 for 3-CMIS. □

Since the 3-MPSM problem can be reduced to 3-CMIS problem, an approximation algorithm for the 3-MPSM problem can be achieved with the same approximation ratio as Algorithm 4.2. Thus, we can get the following conclusion.

Corollary 4.4. *There is an approximation algorithm with expected approximation ratio 9 for 3-MPSM.*

Similar to Algorithm 4.2, we can design an approximation algorithm for k-CMIS with approximation ratio k^2. The algorithm details can be found in Algorithm 4.3.

Thus, using the similar proof method of Theorem 4.3, it is easy to show the following conclusion (in the proof of Theorem 4.3, 3 and 9 are replaced with k and k^2 respectively).

Theorem 4.5. *Algorithm 4.3 is of expected approximation ratio k^2 for k-CMIS ($k \geqslant 4$).*

1: By solving the Linear Programming formulation for the k-CMIS problem, we get an optimum solution x_r^{ip*}, $z^*_{r_{ip}s_{jq}}$ for all r.
2: Randomized rounding:
3: **for** $r = 1$ to m **do**
4: **if** $n_r = k$ **then**
5: Let $X_r = \begin{pmatrix} x_r^{11*} & x_r^{12*} & \cdots & x_r^{1k*} \\ x_r^{21*} & x_r^{22*} & \cdots & x_r^{2k*} \\ \vdots & \vdots & \vdots & \vdots \\ x_r^{k1*} & x_r^{k2*} & \cdots & x_r^{kk*} \end{pmatrix}$.
6: Let $\overrightarrow{Y_1} = (x_r^{11*}, x_r^{22*}, \ldots, x_r^{kk*})$, $\overrightarrow{Y_2} = (x_r^{12*}, x_r^{23*}, \ldots, x_r^{(k-1)k*}, x_r^{k1*}), \ldots, \overrightarrow{Y_k} = (x_r^{1k*}, x_r^{21*}, x_r^{32*}, \ldots, x_r^{k(k-1)*})$.
7: Let $S_1 = x_r^{11*} + x_r^{22*} + \cdots + x_r^{kk*}$, $S_2 = x_r^{12*} + x_r^{23*} + \cdots + x_r^{(k-1)k*} + x_r^{k1*}), \ldots, S_k = x_r^{1k*} + x_r^{21*} + x_r^{32*} + \cdots + x_r^{k(k-1)*}$.
8: $\overrightarrow{Y_i}$ is chosen with probability $\frac{\sqrt{S_i}}{\sqrt{S_1}+\sqrt{S_2}+\cdots+\sqrt{S_k}}$ $(i = 1, 2, \ldots, k)$.
9: When $\overrightarrow{Y_1}$ is chosen, we set $x_r^{11} = x_r^{22} = \cdots = x_r^{kk} = 1$; When $\overrightarrow{Y_2}$ is chosen, we set $x_r^{12} = x_r^{23} = \cdots = x_r^{(k-1)k} = x_r^{k1} = 1; \ldots$; When $\overrightarrow{Y_k}$ is chosen, we set $x_r^{1k} = x_r^{21} = x_r^{32} = x_r^{k(k-1)} = 1$;
10: **end if**
11: **if** $n_r = k'$ and $2 < k' < k$ **then**
12: We set $x_r^{ip} = 1$ by the similar method above.
13: **end if**
14: **if** $n_r = 2$ **then**
15: We set $x_r^{ip} = 1$ by the method in Algorithm 4.1.
16: **end if**
17: **if** $n_r = 1$ **then**
18: We set $x_r^{11} = 1$.
19: **end if**
20: **end for**
21: Output those nodes whose variables are set to 1. Let $z'_{r_{ip}s_{jq}} = x_r^{ip} x_s^{jq}$. The edge number of induced subgraph by these output nodes is $\sum_{(v_r^{ip}, v_s^{jq}) \in E} z'_{r_{ip}s_{jq}}$.

Algorithm 4.3: The approximation algorithm for the k-CMIS problem.

Note. In Algorithm 4.3, the elements of $\overrightarrow{Y_i}(1 \leqslant i \leqslant k)$ are at different rows and different columns. There are $k!$ possible these vectors. We choose k diagonal elements as these $\overrightarrow{Y_i}$. In order to improved the approximation ratio k^2, it is required to use better methods for choosing these $\overrightarrow{Y_i}$. We leave it as an open problem.

Since the k-MPSM problem can be reduced to k-CMIS problem, an approximation algorithm for the k-MPSM problem can be achieved with the same approximation ratio as Algorithm 4.3. Thus, we can get the following conclusion.

Corollary 4.6. *There is an approximation algorithm with expected approximation ratio k^2 for k-MPSM ($k \geqslant 4$).*

5. Conclusion

In this paper, we have proved that CMIS and CNIS are NP-complete. We have also proposed a 2-approximation algorithm for 2-CMIS and a k^2-approximation algorithm for k-CMIS ($k \geqslant 3$), which are based on the randomized rounding technology. Based on approximation algorithms for k-CMIS, we get approximation algorithms for k-MPSM with the same approximation ratio.

Acknowledgements

This research has been supported by the "Exploratory Data Intensive Computing for Complex Biological Systems" project from U.S. Department of Energy (Office of Advanced Scientific Computing Research, Office of Science). The work of N.F.S. was also sponsored by the Laboratory Directed Research and Development Program of Oak Ridge National Laboratory. Oak Ridge National Laboratory is managed by UT-Battelle for the LLC U.S. D.O.E. under contract No. DEAC05-00OR22725.

Wenbin Chen's research has been still supported by the National Natural Science Foundation of China (NSFC) under Grant No. 11271097, the research project of Guangzhou Education Bureau under Grant No. 2012A074, the project IIPL-2011-001 from Shanghai Key Laboratory of Intelligent Information Processing, and the project KFKT2012B01 from State Key Laboratory for Novel Software Technology, Nanjing University. Lingxi Peng's research has been partly supported by the National Natural Science Foundation of China (NSFC) under Grant No. 61100150 and the research project of Guangzhou Education Bureau under Grant No. 2012A077. Jianxiong Wang's research was partially supported under Guangzhou City Council's Science and Technology Projects funding scheme (project number 12C42011622), under Guangdong Provincial Education Department's Yumiao early career researchers development funding scheme (2012WYM0105 and 2012LYM0105) and the research project of Guangzhou Education Bureau under Grant No. 2012A143. Maobin Tang's research has been supported under Guangdong Province's Science and Technology Projects under Grant Nos. 2011B020313023 and 2012A020602065 and the research project of Guangzhou Education Bureau under Grant No. 2012A075.

References

[1] S. Arora, Probabilistic checking of proofs and hardness of approximation problems, PhD thesis, UC Berkeley, 1994.
[2] S. Arora, L. Babai, J. Stern, Z. Sweedyk, The hardness of approximate optima in lattices, codes, and systems of linear equations, in: FOCS, 1993, pp. 724–733.
[3] S. Arora, C. Lund, R. Motwani, M. Sudan, M. Szegedy, Proof verification and intractability of approximation problems, in: Proc. 33rd IEEE Symp. on Foundations of Computer Science, 1992, pp. 13–22.
[4] S. Arora, S. Safra, Probabilistic checking of proofs: A new characterization of NP, in: Proc. 33rd IEEE Symp. on Foundations of Computer Science, 1992, pp. 2–13.
[5] D. Bertsimas, C. Teob, R. Vohrac, On dependent randomized rounding algorithms, Oper. Res. Lett. 24 (1999) 105–114.
[6] M. Chrobak, P. Kolman, J. Sgall, The greedy algorithm for the minimum common string partition problem, in: 7th Int. Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2004, in: Lect. Notes Comput. Sci., vol. 3122, 2004, pp. 84–95.
[7] T. Cormen, C. Leiserson, R. Rivest, C. Stein, Introduction to Algorithms, The MIT Press, 2001.
[8] X. Chen, J. Zheng, Z. Fu, P. Nan, Y. Zhong, S. Lonardi, T. Jiang, Assignment of orthologous genes via genome rearrangement, IEEE/ACM Trans. Comput. Biol. Bioinform. 2 (4) (2005) 302–315.
[9] W. Chen, Z. Zheng, N.F. Samatova, Approximation algorithms for the maximum duo-preservation string mapping problem, in: ICFIT, 2010, pp. 190–198.
[10] U. Feige, S. Goldwasser, L. Lovàsz, S. Safra, M. Szegedy, Approximating clique is almost NP-complete, in: Proc. 32nd IEEE Symp. on Foundations of Computer Science, 1991, pp. 2–12.
[11] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, 1979.
[12] A. Goldstein, P. Kolman, J. Zheng, Minimum common string partition problem: hardness and approximation, in: Proc. of International Symposium on Algorithms and Computation, Hong Kong, China, in: Lect. Notes Comput. Sci., vol. 3341, 2004, pp. 484–495.
[13] M.X. Goemans, D. Williamson, A new 3/4 approximation algorithm for MAX SAT, in: Proceedings of the Third IPCO Conference, 1993, pp. 313–321.
[14] M.X. Goemans, D. Williamson, 0.878 approximation algorithms for MAX-CUT and MAX 2SAT, in: Proceedings of the 26th Annual ACM STOC, 1994, pp. 422–431.
[15] D.S. Hochbaum (Ed.), Approximating covering and packing problem: set cover, vertex cover, independent set, and related problem, Approximation Algorithms for NP-Hard Problems, PWS Publishing Company, 1996, pp. 94–143.
[16] Qiaoming Han, D. Yinyu Ye, Jiawei Zhang, Approximation of Dense-k subgraph, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.41.1899.
[17] Gerold Jäger, Anand Srivastav, Katja Wolf, Solving generalized maximum dispersion with linear programming, in: AAIM 2007, in: Lect. Notes Comput. Sci., vol. 4508, 2007, pp. 1–10.
[18] P. Kolman, T. Waleń, Approximating reversal distance for strings with bounded number of duplicates, Discrete Appl. Math. 155 (3) (2007) 327–336.
[19] P. Kolman, T. Waleń, Reversal distance for strings with duplicates: linear time approximation using hitting set, in: WAOA, 2006, pp. 279–289.
[20] F.T. Leighton, S. Rao, Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms, J. ACM 46 (6) (1999) 787–832.
[21] P. Raghavan, C. Thompson, Randomized rounding: a technique for provably good algorithms and algorithmic proofs, Combinatorica 7 (1987) 365–374.
[22] D.B. Shmoys, Using linear programming in the design and analysis of approximation algorithms: two illustrative problems, APPROX (1998) 15–32.