Department of Social Systems and Management

Discussion Paper Series

No.1291

Lagrangian Relaxation and Pegging Test for the Clique Partitioning Problem

by

Noriyoshi SUKEGAWA, Yoshitsugu YAMAMOTO, and Liyuan ZHANG

May 2012

UNIVERSITY OF TSUKUBA
Tsukuba, Ibaraki 305-8573
JAPAN

LAGRANGIAN RELAXATION AND PEGGING TEST FOR THE CLIQUE PARTITIONING PROBLEM

NORIYOSHI SUKEGAWA, YOSHITSUGU YAMAMOTO, AND LIYUAN ZHANG

Abstract. The clique partitioning problem is an *NP*-hard combinatorial optimization problem with applications to data analysis such as clustering. Though a binary integer linear programming formulation has been known for years, one needs to deal with a huge number of variables and constraints when solving a large instance. In this paper, we propose a size reduction algorithm which is based on the Lagrangian relaxation and the pegging test, and verify its validity through numerical experiments. We modify the conventional subgradient method in order to manage the high dimensionality of the Lagrangian multipliers, and also make an improvement on the ordinary pegging test by taking advantage of the structural property of the clique partitioning problem.

1. Introduction

Given a set of objects with a number of attributes, cluster analysis aims to group them into clusters according to similarity or dissimilarity of the attributes. When the attributes are quantitative, plenty of choices of dissimilarity measurement are available such as Euclidean, Minkowsky and Mahalanobis distances. On the other hand, we have few choices for the qualitative attributes as mentioned by Brusco and Köhn [3]. An approach would be the qualitative-quantitative data conversion, i.e., assigning numerical numbers to the qualitative attributes. This is simple hence popular among practitioners, but the clustering can be greatly affected by the conversion scheme.

Suppose that a qualitative attribute consists of three distinct symbols. Then it defines an equivalence relation on the set of objects and forms three equivalence classes, i.e., two objects having the same symbol are equivalent. Thus if there are q attributes, we have q different equivalence relations. Grötschel and Wakabayashi [8] proposed to aggregate those equivalence relations for a clustering of the objects, and formulated it to a combinatorial optimization problem called *Clique Partitioning Problem*, *CPP* for short. They proposed a cutting plane algorithm based on the simplex method and solved real-world instances up to 158 objects to optimality. For subsequent researches including heuristic methods, see [1, 3, 11, 13]. In recent years, CPP has broadened its application range to other real-world problems such as the flight-to-gate assignment problem [5] and microarray data analysis [9].

The CPP for n objects, when formulated as a binary integer linear programming, has $O(n^2)$ binary variables, $O(n^2)$ equality constraints and $O(n^3)$ inequality constraints, all of which grow very rapidly as n grows and will soon outstrip the computing capability of currently available optimizers. Therefore we need to devise means of reducing the problem size. If we have the information that a binary variable is zero at any optimal solution, we can fix it to zero, delete it, and reduce the number of variables. Such information is obtained by comparing the lower bounds of the optimal value of the problem and the

Date: May 6, 2012.

Key words and phrases. Clique partitioning, Clustering, *NP*-hard, Binary integer programming, Lagrangian dual problem, Subgradient method.

This work is partly supported by Grant-in-Aid for Scientific Research (C) No.22510136.

upper bound of the optimal value of the problem with a variable temporarily fixed to one. If the latter is less than the former, we can conclude that the variable should be zero at any optimal solution. This idea, called the *pegging test*, was proposed years ago, and favorable results have been reported on the knapsack problem [12, 14] and the set covering problem [2]. One of the aims of this paper is to propose the pegging test which is tailored to exploit the problem structure of the CPP. To obtain the upper bound of the problem, we use the Lagrangian relaxation problem instead of the conventional linear programming relaxation since the problem size remains intractable even when the binary constraints are relaxed. We combine the subgradient method for solving the Lagrangian dual problem with some heuristics and local search.

This paper is organized as follows. In Section 2, we give the definition of CPP and its binary integer linear programming formulation. In Section 3, we first summarize the conventional Lagrangian relaxation problem and then develop the subgradient method for the Lagrangian dual problem so that it can handle a huge number of Lagrangian multipliers. In Section 4, we develop a heuristic method and introduce an efficient and numerically stable method. In Section 5, we improve the ordinary pegging test to utilize the structural property of CPP. Describing our algorithm in Section 6, we report the numerical experiments of our method in Section 7. Some comments are made in Section 8.

2. Clique partitioning problem

We first introduce the clique partitioning problem and its formulation as a binary integer linear programming problem according to Grötschel and Wakabayashi [8].

2.1. **Definitions and notations.**

We denote an undirected graph G with vertex set N and edge set E by $G = (N, E)$. An edge e with endnodes u and v is denoted by $\{u, v\}$. For a set S of vertices, we denote the set of edges in G with both endnodes in S by $E(S)$, that is, $E(S) = \{\{u, v\} \in E \mid u, v \in S\}$. For a set of subsets $S_1, S_2, \dots, S_k$ of N, let

$$E(S_1, S_2, \dots, S_k) := \bigcup_{i=1}^{k} E(S_i).$$

A graph is called complete if every pair of its nodes is linked by an edge. A clique is a complete subgraph.

We say that $\mathit{\Gamma} = \{W_1, W_2, \dots, W_k\}$ is a partition of N if $W_i \cap W_j = \emptyset$ for $1 \le i < j \le k$, $V = W_1 \cup W_2 \cup \cdots \cup W_k$, and $W_i \neq \emptyset$ for $1 \le i \le k$. A set A of edges in a graph $G = (N, E)$ is called a clique partitioning of G if there is a partition $\mathit{\Gamma} = \{W_1, W_2, \dots, W_k\}$ of N such that $A = E(W_1, W_2, \dots, W_k)$ and the subgraph $(W_i, E(W_i) \cap A)$ induced by W_i is a clique for $1 \le i \le k$. In case G is complete, every partition of N induces a clique partitioning.

The clique partitioning problem (CPP) is formally defined as follows [8]. Given a complete graph $K = (N, E)$ with weights $c_e \in \mathbb{R}$ for all $e \in E$, find a clique partitioning $A \subseteq E$ such that

$$c(A) := \sum_{e \in A} c_e$$

is as large as possible.

2.2. **Binary integer linear programming formulation.**

In this section, we show a standard formulation of CPP as a binary integer linear programming problem.

For the sake of simplicity we assume that $N = \{1, 2, \dots, n\}$ in what follows. For a clique partitioning A let binary variables x_{ij} for $\{i, j\} \in E$ be defined as

$$x_{ij} = \begin{cases} 1 & \{i, j\} \in A, \\ 0 & \text{otherwise}, \end{cases}$$

then CPP is formulated as

$$\left|\begin{array}{llll}\text{maximize} & \displaystyle\sum_{(i,j)\in N^2_{\neq}} \bar{c}_{ij}x_{ij} & & \\ \text{subject to} & x_{ij}\in\{0,1\} & \text{for all } \{i,j\}\in E & \text{(binary)} \\ & x_{ij}-x_{ji}=0 & \text{for all } \{i,j\}\in E & \text{(symmetry)} \\ & x_{ij}+x_{jk}-x_{ik}\le 1 & \text{for all } i,j,k\in N, i\neq j, j\neq k, k\neq i, & \text{(transitivity)}\end{array}\right.$$

where

$$\bar{c}_{ij}=\bar{c}_{ji}:=\tfrac{c_{\{i,j\}}}{2}.$$

The transitivity constraints guarantee that the relation defined by those variables x_{ij}'s is transitive (i.e., if i and j are in the same clique and j and k are in the same clique, then i and k must be in the same clique), and this assures that the feasible solutions correspond to the clique partitioning. The point is that this formulation has $n(n-1)$ binary variables, $n(n-1)/2$ equality constraints and $n(n-1)(n-2)/2$ inequality constraints, all of which grow very rapidly as n grows. Substituting x_{ij} for x_{ji} for all $i,j\in N$ with $i<j$ halves the decision variables and yields the following equivalent problem

(P) :

$$\left|\begin{array}{llll}\text{maximize} & \displaystyle\sum_{(i,j)\in N^2_{<}} c_{ij}x_{ij} & & \\ \text{subject to} & x_{ij}\in\{0,1\} & \text{for all } (i,j)\in N^2_{<}, & \\ & x_{ij}+x_{jk}-x_{ik}\le 1 & \text{for all } (i,j,k)\in N^3_{<} & \text{(type U)} \\ & x_{ij}-x_{jk}+x_{ik}\le 1 & \text{for all } (i,j,k)\in N^3_{<} & \text{(type V)} \\ & -x_{ij}+x_{jk}+x_{ik}\le 1 & \text{for all } (i,j,k)\in N^3_{<}, & \text{(type W)}\end{array}\right.$$

where

$$\begin{aligned}&c_{ij}:=c_{\{i,j\}},\\ &N^2_{<}:=\{\,(i,j)\mid i,j\in N, i<j\,\},\\ &N^3_{<}:=\{\,(i,j,k)\mid i,j,k\in N, i<j<k\,\}.\end{aligned}$$

The inequality constraints separated into three types, which we call type U, V and W, respectively. We will denote the optimal objective function value of the problem (P) by $\omega(P)$.

3. Lagrangian Relaxation and Subgradient Method

We propose to solve the Lagrangian relaxation problem of (P) instead of the linear programming relaxation in order to obtain an upper bound of $\omega(P)$. The conventional Lagrangian relaxation problem needs to handle Lagrangian multipliers as many as the constraints of (P), which can easily exceed the manipulable number. Then we propose a modified Lagrangian relaxation problem and modified subgradient method.

3.1. **Lagrangian relaxation problem.** One of the common tricks to deal with the problem (P) would be the *Lagrangian relaxation problem.* Namely, introducing a nonnegative multiplier u_{ijk} for each constraint of type U, v_{ijk} for type V and w_{ijk} for type W, we consider the following integer linear programming problem

$(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$:

$$\left|\begin{array}{lll} \text{maximize} & \displaystyle\sum_{(i,j)\in N^2_<} c_{ij}x_{ij} & + \displaystyle\sum_{(i,j,k)\in N^3_<} u_{ijk}(1-x_{ij}-x_{jk}+x_{ik}) \\ & & + \displaystyle\sum_{(i,j,k)\in N^3_<} v_{ijk}(1-x_{ij}+x_{jk}-x_{ik}) \\ & & + \displaystyle\sum_{(i,j,k)\in N^3_<} w_{ijk}(1+x_{ij}-x_{jk}-x_{ik}) \\ \text{subject to} & x_{ij}\in\{0,1\} & \text{for all } (i,j)\in N^2_< \end{array}\right.$$

where $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ denote multiplier vectors $(u_{ijk})_{(i,j,k)\in N^3_<}$, $(v_{ijk})_{(i,j,k)\in N^3_<}$ and $(w_{ijk})_{(i,j,k)\in N^3_<}$, respectively. The inequality constraints are relaxed and plugged into the objective function as a penalty. Denoting the coefficient of variable x_{ij} in the objective function by $r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij}$, the objective function of $(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$ is rewritten as

$$\sum_{(i,j,k)\in N^3_<} (u_{ijk}+v_{ijk}+w_{ijk}) + \sum_{(i,j)\in N^2_<} r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij}x_{ij},$$

and the coefficient $r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij}$ is given as

$$\begin{aligned} r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij} = c_{ij} &- \sum_{k:(i,j,k)\in N^3_<} u_{ijk} - \sum_{k:(k,i,j)\in N^3_<} u_{kij} + \sum_{k:(i,k,j)\in N^3_<} u_{ikj} \\ &- \sum_{k:(i,j,k)\in N^3_<} v_{ijk} + \sum_{k:(k,i,j)\in N^3_<} v_{kij} - \sum_{k:(i,k,j)\in N^3_<} v_{ikj} \\ &+ \sum_{k:(i,j,k)\in N^3_<} w_{ijk} - \sum_{k:(k,i,j)\in N^3_<} w_{kij} - \sum_{k:(i,k,j)\in N^3_<} w_{ikj}. \end{aligned} \tag{3.1}$$

Due to the simple constraint of $(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$, its optimal solution can be obtained simply by

$$x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij} = \begin{cases} 1 & \text{if } r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij} > 0, \\ 0 & \text{if } r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij} \le 0. \end{cases} \tag{3.2}$$

The optimal objective function value $\omega(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$ of the problem $(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$ provides an upper bound of $\omega(P)$, and it clearly depends on the choice of the Lagrangian multiplier vector $(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$. We propose in Section 3.2 to apply the subgradient method to find a multiplier vector that provides the smallest value of $\omega(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$.

Even if the optimal solution $\boldsymbol{x}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ of the problem $(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$ is feasible to the problem (P), it is not necessarily optimal for the problem (P). The following theorem is well known. See e.g., Geoffrion [7].

Theorem 3.1. *Let* $(\bar{\boldsymbol{u}},\bar{\boldsymbol{v}},\bar{\boldsymbol{w}}) := ((\bar{u}_{ijk})_{(i,j,k)\in N^3_<}, (\bar{v}_{ijk})_{(i,j,k)\in N^3_<}, (\bar{w}_{ijk})_{(i,j,k)\in N^3_<})$ *be a Lagrangian multiplier vector, and let* $\boldsymbol{x}$ *be an optimal solution of the Lagrangian relaxation problem of the problem* (P) *with* $(\bar{\boldsymbol{u}},\bar{\boldsymbol{v}},\bar{\boldsymbol{w}})$. *If* $\boldsymbol{x}$ *is feasible to the problem* (P) *and satisfies the complementarity condition :*

$$\begin{aligned} \bar{u}_{ijk}(1-x_{ij}-x_{jk}+x_{ik}) = 0 \quad & \text{for all } (i,j,k)\in N^3_<, \\ \bar{v}_{ijk}(1-x_{ij}+x_{jk}-x_{ik}) = 0 \quad & \text{for all } (i,j,k)\in N^3_<, \\ \bar{w}_{ijk}(1+x_{ij}-x_{jk}-x_{ik}) = 0 \quad & \text{for all } (i,j,k)\in N^3_<, \end{aligned}$$

then it is an optimal solution of the problem (P).

3.2. **Subgradient method.** For the sake of notational simplicity we abbreviate $\omega(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$ to $\omega(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ within this section. The *Lagrangian dual problem*, denoted by (LD), is a problem for finding the smallest value of $\omega(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$, i.e., the best upper bound of $\omega(P)$. Namely,

(LD) :

$$\left|\begin{array}{ll} \text{minimize} & \omega(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}) \\ \text{subject to} & \boldsymbol{u},\boldsymbol{v},\boldsymbol{w} \geq \boldsymbol{0}. \end{array}\right.$$

The objective function $\omega(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ is convex, piecewise linear and not differentiable on the intersection of pieces, and one of the most widely used methods for this problem is the *subgradient method.* See for example Fisher [6].

Definition 3.2. $(\boldsymbol{g}^u,\boldsymbol{g}^v,\boldsymbol{g}^w)$ is said to be a *subgradient* of ω at $(\bar{\boldsymbol{u}},\bar{\boldsymbol{v}},\bar{\boldsymbol{w}}) \geq \boldsymbol{0}$ when

$$\omega(\bar{\boldsymbol{u}},\bar{\boldsymbol{v}},\bar{\boldsymbol{w}}) + \langle \boldsymbol{g}^u, \boldsymbol{u}-\bar{\boldsymbol{u}}\rangle + \langle \boldsymbol{g}^v, \boldsymbol{v}-\bar{\boldsymbol{v}}\rangle + \langle \boldsymbol{g}^w, \boldsymbol{w}-\bar{\boldsymbol{w}}\rangle \leq \omega(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$$

holds for any $(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}) \geq \boldsymbol{0}$, where $\langle\cdot,\cdot\rangle$ means the inner product.

The following lemma is well known.

Lemma 3.3. *Let $\boldsymbol{x}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ denote an optimal solution of the Lagrangian relaxation problem $(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$. Then $(\boldsymbol{g}^u,\boldsymbol{g}^v,\boldsymbol{g}^w)$ defined as*

$$\begin{aligned} g^u_{ijk} &:= 1 - x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij} - x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{jk} + x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ik} \quad \text{for } (i,j,k) \in N^3_<, \\ g^v_{ijk} &:= 1 - x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij} + x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{jk} - x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ik} \quad \text{for } (i,j,k) \in N^3_<, \\ g^w_{ijk} &:= 1 + x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij} - x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{jk} - x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ik} \quad \text{for } (i,j,k) \in N^3_< \end{aligned}$$

is a subgradient of ω at $(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$.

The subgradient method uses the following rule to update the iterate of multiplier vector $(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ to the next iterate $(\boldsymbol{u}^+,\boldsymbol{v}^+,\boldsymbol{w}^+)$.

$$u^+_{ijk} := \max\left\{0,\ u_{ijk} - \mu\frac{\omega(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})-\omega_{\text{low}}}{\|(\boldsymbol{g}^u,\boldsymbol{g}^v,\boldsymbol{g}^w)\|^2}g^u_{ijk}\right\} \quad \text{for } (i,j,k) \in N^3_<, \tag{3.3}$$

$$v^+_{ijk} := \max\left\{0,\ v_{ijk} - \mu\frac{\omega(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})-\omega_{\text{low}}}{\|(\boldsymbol{g}^u,\boldsymbol{g}^v,\boldsymbol{g}^w)\|^2}g^v_{ijk}\right\} \quad \text{for } (i,j,k) \in N^3_<, \tag{3.4}$$

$$w^+_{ijk} := \max\left\{0,\ w_{ijk} - \mu\frac{\omega(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})-\omega_{\text{low}}}{\|(\boldsymbol{g}^u,\boldsymbol{g}^v,\boldsymbol{g}^w)\|^2}g^w_{ijk}\right\} \quad \text{for } (i,j,k) \in N^3_<, \tag{3.5}$$

where μ is a step size control parameter and ω_{low} is a lower bound of $\omega(P)$. It is known that if ω_{low} is replaced by the optimal value $\omega(P)$, the sequence generated will converge to an optimal solution of the Lagrangian dual problem (LD), see e.g., Geoffrion [7] and Larsson et al. [10]. Therefore the quality of lower bound ω_{low} has a decisive influence on the convergence of the method. The value $\omega(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ does not necessarily decrease when the multiplier vector is updated. Often the parameter μ is initially set to 2.0 and is halved whenever $\omega(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ fails to decrease in some fixed number of iterations. This rule performs well empirically. The method we used is as follows.

Basic Subgradient Method (BSM)

Step 1 (Initialization)
- (a) $l \leftarrow 0$, $\mu \leftarrow 2.0$, $(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}) \leftarrow (\boldsymbol{0},\boldsymbol{0},\boldsymbol{0})$.
- (b) $\omega_{\text{low}} \leftarrow$ the objective value of a feasible solution obtained by some heuristic algorithm (see Section 4). $\omega_{\text{up}} \leftarrow \omega(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$.

Step 2 (Solving $(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$

(a) Compute $r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij}$ by (3.1) and set $x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij}$ according to (3.2).

(b) $\omega_{\text{up}} \leftarrow \min\{\omega_{\text{up}}, \omega(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})\}$. If ω_{up} is not improved, $l \leftarrow l+1$. Otherwise, $l \leftarrow 0$.

Step 3 (Optimality check)

If $\boldsymbol{x}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ satisfies the optimality condition in Corollary 3.4 with $(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ (in this case, $\boldsymbol{x}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ is optimal for the problem (P)), then terminate. If $\omega_{\text{up}} - \omega_{\text{low}} < \varepsilon$, then terminate where ε is a predetermined tolerance for the duality gap.

Step 4 (Update of μ)

If $\mu \leq 0.005$, then terminate. If l reaches 30, $\mu \leftarrow \mu/2$.

Step 5 (Update of $(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$)

Update $(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ according to (3.3), (3.4) and (3.5) and go to Step 2.

3.3. **Modified Lagrangian relaxation.**

Since the Lagrangian dual problem (LD) has to handle the multipliers as many as the constraints of the problem (P), which amount to $n(n-1)(n-2)/2$, we need to reserve a huge capacity of memory to solve (LD) for a large CPP instance. To overcome this, we present a modified Lagrangian relaxation problem, which temporarily discards some of the transitivity constraints and retrieves when needed. Let U, V and W be subsets of $N^3_<$, and $(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ be a nonnegative multiplier vector respectively. Then the problem we consider is

$(LR(U,V,W,\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$:

$$\left|\begin{array}{lll} \text{maximize} & \displaystyle\sum_{(i,j)\in N^2_<} c_{ij}x_{ij} & + \displaystyle\sum_{(i,j,k)\in U} u_{ijk}(1 - x_{ij} - x_{jk} + x_{ik}) \\ & & + \displaystyle\sum_{(i,j,k)\in V} v_{ijk}(1 - x_{ij} + x_{jk} - x_{ik}) \\ & & + \displaystyle\sum_{(i,j,k)\in W} w_{ijk}(1 + x_{ij} - x_{jk} - x_{ik}) \\ \text{subject to} & x_{ij} \in \{0,1\} & \text{for all } (i,j) \in N^2_<. \end{array}\right.$$

This modified Lagrangian relaxation problem differs from the ordinary one in that the multipliers with indices not in (U,V,W) are fixed to zero. This reduces the number of multipliers that we have to handle to $|U|+|V|+|W|$. The modified Lagrangian relaxation problem could be viewed as an ordinary Lagrangian relaxation problem of the problem (P) with some transitivity constraints being relaxed:

$(P(U,V,W))$:

$$\left|\begin{array}{lll} \text{maximize} & \displaystyle\sum_{(i,j)\in N^2_<} c_{ij}x_{ij} & \\ \text{subject to} & x_{ij} \in \{0,1\} & \text{for all } (i,j) \in N^2_<, \\ & x_{ij} + x_{jk} - x_{ik} \leq 1 & \text{for all } (i,j,k) \in U, \\ & x_{ij} - x_{jk} + x_{ik} \leq 1 & \text{for all } (i,j,k) \in V, \\ & -x_{ij} + x_{jk} + x_{ik} \leq 1 & \text{for all } (i,j,k) \in W. \end{array}\right.$$

It is likely that some of the discarded inequality constraints would be satisfied. Thus, if we can find a "good and small" triplet of (U,V,W), the modified Lagrangian relaxation problem would be in performance comparable to and in computation better than the ordinary one.

For the problem $(LR(U,V,W,\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$ the coefficient $r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij}$ of variable x_{ij} in the objective function is written as

$$\begin{aligned}
r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij} = c_{ij} &- \sum_{k:(i,j,k)\in U} u_{ijk} - \sum_{k:(k,i,j)\in U} u_{kij} + \sum_{k:(i,k,j)\in U} u_{ikj} \\
&- \sum_{k:(i,j,k)\in V} v_{ijk} + \sum_{k:(k,i,j)\in V} v_{kij} - \sum_{k:(i,k,j)\in V} v_{ikj} \\
&+ \sum_{k:(i,j,k)\in W} w_{ijk} - \sum_{k:(k,i,j)\in W} w_{kij} - \sum_{k:(i,k,j)\in W} w_{ikj}.
\end{aligned} \tag{3.6}$$

and an optimal solution $\boldsymbol{x}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}) = (x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij})_{(i,j)\in N^2_<}$ can be obtained by

$$x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij} = \begin{cases} 1 & \text{if } r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij} > 0, \\ 0 & \text{if } r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij} \le 0. \end{cases} \tag{3.7}$$

Similar to Theorem 3.1, the optimality condition is given as follows.

Corollary 3.4. *If an optimal solution $\boldsymbol{x}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ of the problem $(LR(U,V,W,\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$ is feasible to the problem (P) and satisfies the restricted complementarity condition with $(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$:*

$$\begin{aligned}
u_{ijk}(1 - x_{ij} - x_{jk} + x_{ik}) &= 0 \quad \text{for all } (i,j,k)\in U, \\
v_{ijk}(1 - x_{ij} + x_{jk} - x_{ik}) &= 0 \quad \text{for all } (i,j,k)\in V, \\
w_{ijk}(1 + x_{ij} - x_{jk} - x_{ik}) &= 0 \quad \text{for all } (i,j,k)\in W,
\end{aligned}$$

then it is an optimal solution of the problem (P).

Proof. We readily see that the Lagrangian relaxation problem $(LR(U,V,W,\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$ is an ordinary Lagrangian relaxation problem for $(\bar{\boldsymbol{u}},\bar{\boldsymbol{v}},\bar{\boldsymbol{w}})$ such that

$$\bar{u}_{ijk} = \begin{cases} u_{ijk} & \text{for } (i,j,k)\in U, \\ 0 & \text{for } (i,j,k)\in N^3_< \setminus U, \end{cases}$$

$$\bar{v}_{ijk} = \begin{cases} v_{ijk} & \text{for } (i,j,k)\in V, \\ 0 & \text{for } (i,j,k)\in N^3_< \setminus V, \end{cases}$$

$$\bar{w}_{ijk} = \begin{cases} w_{ijk} & \text{for } (i,j,k)\in W, \\ 0 & \text{for } (i,j,k)\in N^3_< \setminus W. \end{cases}$$

When $\boldsymbol{x}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ meets the restricted complementarity condition with $(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$, it also satisfies the complementarity condition for all constraints with $(\bar{\boldsymbol{u}},\bar{\boldsymbol{v}},\bar{\boldsymbol{w}})$. This together with the feasibility of $\boldsymbol{x}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ and Theorem 3.1 yields the desired result. □

Let us abbreviate $\omega(LR(U,V,W,\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$ to $\omega(U,V,W,\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$. Accordingly, we modify the Lagrangian dual problem as follows.

$(LD(U,V,W))$:

$$\left|\begin{array}{ll}
\text{minimize} & \omega(U,V,W,\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}) \\
\text{subject to} & \boldsymbol{u},\boldsymbol{v},\boldsymbol{w} \ge \boldsymbol{0} \\
& u_{ijk} = 0 \text{ for all } (i,j,k)\in N^3_<\setminus U \\
& v_{ijk} = 0 \text{ for all } (i,j,k)\in N^3_<\setminus V \\
& w_{ijk} = 0 \text{ for all } (i,j,k)\in N^3_<\setminus W.
\end{array}\right.$$

In spite of the reduction in the number of multipliers, the problem of finding a "good and small" triplet of (U, V, W) remains unsolved. To solve the modified Lagrangian dual problem, we propose the following modified subgradient method (*MSM* for short) including the subgradient method and the updating scheme of the (U, V, W). The MSM is composed of the inner and outer cycles. The inner cycle consisting of Step 2 to 6 generates a sequence of Lagrangian multiplier vectors, and the outer cycle expands the constraint index set (U, V, W).

Modified Subgradient Method (MSM)

Step 1 (Initialization)
- (a) $k, l, q \leftarrow 0$, $\mu \leftarrow 2.0$, $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}) \leftarrow (\boldsymbol{0}, \boldsymbol{0}, \boldsymbol{0})$.
- (b) $\omega_{\text{low}} \leftarrow$ the objective value of a feasible solution obtained by some heuristic algorithm (see Section 4). $\omega_{\text{up}}, \bar{\omega}_{\text{up}} \leftarrow \omega((LR(\emptyset, \emptyset, \emptyset, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}))$ and $(U, V, W) \leftarrow$ the transitivity constraints violated by $\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$.

Step 2 (Solving $(LR(U, V, W, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}))$
- (a) Compute $r(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})_{ij}$ by (3.6) and set $x(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})_{ij}$ according to (3.7).
- (b) $\omega_{\text{up}} \leftarrow \min\{\omega_{\text{up}}, \omega(LR(U, V, W, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}))\}$. If ω_{up} is not improved, $l \leftarrow l + 1$. Otherwise, $l \leftarrow 0$.

Step 3 (Optimality check)

If $\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ satisfies the optimality condition in Theorem 3.4 with $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ (in this case, $\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ is optimal for the problem (P)), then terminate.

Step 4 (Update of μ)
- (a) If $\mu \leq 0.005$, then $\mu \leftarrow 2.0$ and go to Step 6 (we decide that there is no chance of improving the upper bound unless we expand (U, V, W)).
- (b) If l reaches 5, $\mu \leftarrow \mu/2$.

Step 5 (Update of $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$)

Update $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ according to (3.3), (3.4) and (3.5) and go to Step 2.

Step 6 (Lagrangian heuristics and local search)

Apply the Lagrangian heuristics and local search (see Section 4) to $\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ for a better solution $\tilde{\boldsymbol{x}}$. $\omega_{\text{low}} \leftarrow \max\{\omega_{\text{low}}, \text{objective function value of } \tilde{\boldsymbol{x}}\}$.

Step 7 (Termination)
- (a) If the improvement of ω_{up} compared to $\bar{\omega}_{\text{up}}$ is less than 0.1% or 1 in value then $q \leftarrow q + 1$. $\bar{\omega}_{\text{up}} \leftarrow \omega_{\text{up}}$
- (b) If $q > 10$ then terminate.

Step 8 (Update of (U, V, W))
- (a) Find the transitivity constraints violated by $\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ and add them to U, V and W.
- (b) $u_{ijk}, v_{ijk}, w_{ijk} \leftarrow 0$ for newly added indices (i, j, k) and go to Step 2.

In Step 6, we apply the Lagrangian heuristics and the local search to make a good lower bound from the optimal solution of the problem $(LR(U, V, W, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}))$. The details will be given in Section 4.

3.4. **Theoretical aspect.**

We will show that the problem (LD) always has an optimal solution having at most $n(n-1)/2$ positive elements. This supports the adequacy of the MSM. We first focus on the optimal solution of the linear programming relaxation problem of (P) given as follows.

$(\bar{P})$:

$$\left|\begin{array}{lll} \text{maximize} & \displaystyle\sum_{(i,j)\in N^2_<} c_{ij}x_{ij} & \\ \text{subject to} & 0 \le x_{ij} \le 1 & \text{for all } (i,j)\in N^2_<, \\ & x_{ij}+x_{jk}-x_{ik} \le 1 & \text{for all } (i,j,k)\in N^3_<, \\ & x_{ij}-x_{jk}+x_{ik} \le 1 & \text{for all } (i,j,k)\in N^3_<, \\ & -x_{ij}+x_{jk}+x_{ik} \le 1 & \text{for all } (i,j,k)\in N^3_<. \end{array}\right.$$

Its dual problem is

$(\bar{D})$:

$$\left|\begin{array}{lll} \text{maximize} & \displaystyle\sum_{(i,j,k)\in N^3_<} (u_{ijk}+v_{ijk}+w_{ijk}) + \sum_{(i,j)\in N^2_<} z_{ij} & \\ \text{subject to} & u_{ijk},\ v_{ijk},\ w_{ijk} \ge 0 & \text{for all } (i,j,k)\in N^3_<, \\ & z_{ij} \ge 0 & \text{for all } (i,j)\in N^2_<, \\ & D^u_{ij}+D^v_{ij}+D^w_{ij}+z_{ij} \ge c_{ij} & \text{for all } (i,j)\in N^2_<, \end{array}\right.$$

where

$$\begin{array}{rcl} D^u_{ij} &:=& \displaystyle\sum_{(i,j,k)\in N^3_<} u_{ijk} + \sum_{(k,i,j)\in N^3_<} u_{kij} - \sum_{(i,k,j)\in N^3_<} u_{ikj}, \\ D^v_{ij} &:=& \displaystyle\sum_{(i,j,k)\in N^3_<} v_{ijk} - \sum_{(k,i,j)\in N^3_<} v_{kij} + \sum_{(i,k,j)\in N^3_<} v_{ikj}, \\ D^w_{ij} &:=& \displaystyle -\sum_{(i,j,k)\in N^3_<} w_{ijk} + \sum_{(k,i,j)\in N^3_<} w_{kij} + \sum_{(i,k,j)\in N^3_<} w_{ikj}. \end{array}$$

Since the Lagrangian relaxation problem $(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$ has the *integrality property* (we say that a problem has the integrality property if the optimal value is unchanged when integrality restriction is removed.), we readily see

$$\begin{array}{rll} \omega(\bar{P}) = \omega(\bar{D}) & & \text{(by the duality theorem)} \\ = \omega(LD) & & \text{(by the integrality property).} \end{array}$$

Further, if a multiplier vector $(\boldsymbol{u}^*,\boldsymbol{v}^*,\boldsymbol{w}^*,\boldsymbol{z}^*)$ is optimal for $(\bar{D})$, then the multiplier vector $(\boldsymbol{u}^*,\boldsymbol{v}^*,\boldsymbol{w}^*)$ is optimal for (LD) (see for instance Geoffrion [7]). Since the problem $(\bar{D})$ is a linear programming problem, there exists a basic optimal solution $(\boldsymbol{u}^*,\boldsymbol{v}^*,\boldsymbol{w}^*,\boldsymbol{z}^*)$, and the number of its positive elements is at most the number of inequality constraints, that is $|N^2_<| = n(n-1)/2$. Therefore the Lagrangian dual problem (LD) has an optimal solution whose positive elements are at most $n(n-1)/2$.

4. Heuristics

Finding a good lower bound ω_{low} is essential for the convergence of the subgradient method, and as we will see in Section 5, it also enhances the capabilities of the pegging test. For this purpose we use the *Forgotten Vertices Method with Variant D*, *FVD* for short, proposed by Charon and Hudry [4]. We repeatedly solve the Lagrangian relaxation problem for different Lagrangian multiplier vectors in the MSM. The optimal solution thus obtained may not be feasible to the problem (P), however it provides us with useful information to make a good feasible solution to (P), hence a good lower bound. We first describe the Lagrangian heuristics, then introduce the local search and FVD.

4.1. **Lagrangian heuristics.** Let $K = (N, E)$ denote the complete undirected graph with n vertices. For a given $S \subseteq N$ of k vertices and $A \subseteq E$ let

$$v_A(S) := \begin{cases} v_A^I(S) + v_A^O(S) & (k \geq 2), \\ v_A^O(S) & (k = 1), \\ +\infty & (k = 0), \end{cases}$$

where

$$v_A^I(S) := \frac{|\{\, \{i,j\} \in E \mid \{i,j\} \notin A,\ i,j \in S \,\}|}{k(k-1)},$$
$$v_A^O(S) := \frac{|\{\, \{i,j\} \in E \mid \{i,j\} \in A,\ i \in S,\ j \notin S \,\}|}{k(n-k)}.$$

We readily see that if A is a clique partitioning of $K = (N, E)$ induced by a partition $\Gamma = \{W_1, W_2, \ldots, W_k\}$ of N, then $v_A(W_i) = 0$ for all $i = 1, 2, \ldots, k$. For an optimal solution $\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ of the problem $(LR(U, V, W, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}))$, let $\bar{A}$ be a set of edges such that

$$\bar{A} := \{\, \{i,j\} \in E_n \mid x(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})_{ij} = 1 \,\}.$$

It would be reasonable to think that a partition $\Gamma = \{W_1, W_2, \ldots, W_k\}$ of N minimizing

$$\sum_{i=1}^{k} v_{\bar{A}}(W_i) \tag{4.1}$$

is a feasible solution "nearest" to $\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$. Here we use the following simple greedy algorithm to minimize (4.1).

Basic Lagrangian Heuristics (BLH)

```
Step 1 (Initialization)
    (a) $k := 1$, $W_k := \{\}$, $\Gamma := \{W_k\}$
    (b) $A := \bar{A}$, $V := N$, $E := E_n$ and $F := N$.
Step 2 (Local Search)
    (a) Compute $v_A(W_k \cup \{i\})$ for all $i \in F$.
    (b) $i^* := \operatorname{argmin}\{\, v_A(W_k \cup \{i\}) \mid i \in F \,\}$.
Step 3 (Update of $\Gamma$)
    (a) If $v_A(W_k \cup \{i^*\}) \leq v_A(W_k)$ then $W_k := W_k \cup \{i^*\}$, $F := F \setminus \{i^*\}$ and go to
        Step 4 if $F = \emptyset$ and go to Step 2 otherwise.
    (b) Otherwise, $V := V \setminus \{W_k\}$, $E := E(V)$, $A := A \cap E$, $k := k + 1$, $W_k := \{\}$,
        $\Gamma := \Gamma \cup \{W_k\}$ and go to Step 4.
Step 4 (Termination) Output $\Gamma$ and terminates.
```

The feasible solution obtained by BLH will be used as an initial feasible solution for the local search to start with.

4.2. **Local search.** Given a partition $\Gamma = \{W_1, W_2, \ldots, W_k\}$, we define the neighborhood $N(\Gamma)$ of Γ as follows

$$N(\Gamma) := \bigcup_{i=1}^{n} N_i(\Gamma)$$

where

$$N_i(\Gamma) := \left\{ \Gamma' \;\middle|\; \begin{array}{l} \Gamma' \text{ is obtained from } \Gamma \text{ by moving } i \text{ from its current set } W_s \\ \text{to another one } W_t \in (\Gamma \cup \{\emptyset\}) \setminus \{W_s\} \end{array} \right\}.$$

This neighborhood is equivalent to that suggested by Charon and Hudry [4]. Since the incidence vector of a partition Γ is feasible for the problem (P), in what follows, let $f(\Gamma)$

denote the objective function value of Γ.

Definition 4.1. Given a partition $\Gamma = \{W_1, W_2, \ldots, W_k\}$ and a vertex $i \in V_n$, let $\Gamma_i^* \in N_i(\Gamma)$ denote the *best partition obtained by moving i, i.e.,*

$$\Gamma_i^* = \mathit{argmax}\left\{f(\Gamma') \mid \Gamma' \in N_i(\Gamma)\right\}.$$

We say that i is in the best position if $f(\Gamma_i^*) \le f(\Gamma)$ *holds.*

To find a better feasible solution, we employ the following local search method based on the neighborhood defined above.

Basic Local Search (BLS)

```
Step 1 (Initialization) Set $\Gamma \leftarrow$ an initial feasible solution for the problem $(P)$.
Step 2 (Decide exploration order) Rank the vertices in a random order.
Step 3 (Exploration)
    (a) opt:= true.
    (b) According to the order defined in Step 2, for all vertex $i$, if $i$ is not in the
        best position, opt:= false and $\Gamma \leftarrow \Gamma_i^*$.
Step 4 (Termination)
    (a) If opt is false, then go to Step 2.
    (b) Otherwise output $\Gamma$ (in this case, $\Gamma$ is a local optimal solution) and terminate.
```

4.3. **Forgotten vertices method with variant D.** The Lagrangian heuristics described in the previous section seems to be successful when the corresponding multiplier vector is near optimal. To obtain a good lower bound in an early stage of the computation, we introduce a heuristic method called *Forgotten vertices method with variant D* (*FVD* for short) by Charon and Hudry [4]. This method is mainly based on the above local search but sometimes forgets some vertices when calculating the objective function value. More specifically, when some vertices are forgotten, the *perturbed* objective function value f_p of a partition $\Gamma = \{W_1, W_2, \ldots, W_k\}$ is given by

$$f_p(\Gamma) = \sum_{\substack{e=\{u,v\}\in A \\ u,v \text{ not forgotten}}} c_e$$

where A is a clique partitioning of V_n induced by Γ.

Definition 4.2. Given a partition $\Gamma = \{W_1, W_2, \ldots, W_k\}$ and a vertex $i \in V_n$, let $\tilde{\Gamma}_i^* \in N_i(\Gamma)$ denote the *perturbed best partition obtained by moving i, i.e.,*

$$\tilde{\Gamma}_i^* = \mathit{argmax}\left\{f_p(\Gamma') \mid \Gamma' \in N_i(\Gamma)\right\}.$$

We say that i is in the perturbed best position if $f_p(\Gamma_i^*) \le f_p(\Gamma)$ *holds.*

Forgetting some vertices can be considered as a "noise." Charon and Hudry [4] proposed 18 variants of noise and reported that the following variant D was always the best and numerically stable.

Forgotten vertices method with variant D (FVD)

```
Step 1 (Initialization)
    Set $\Gamma, \Gamma^* \leftarrow$ some partitioning which is randomly computed and $r \leftarrow r_{\max}$.
Step 2 (Initialization for perturbed exploration) $l \leftarrow 1$.
Step 3 (Decide forgotten vertices and exploration order for perturbed exploration)
```

```
        (a) $k \leftarrow$ the product of $r$ by the number of vertices $n$. Select $k$ vertices which
            will be forgotten in Step 3.
        (b) Rank the vertices in a random order.
Step 4  (Perturbed exploration)
        (a) According to the above order, for all vertex $i$, if $i$ is not in the perturbed best
            position, $\Gamma \leftarrow \tilde{\Gamma}_i^*$.
        (b) If $l \geq 4$ then go to Step 5. Otherwise $l \leftarrow l + 1$ and go to Step 3.
Step 5  (Decide exploration order for unperturbed exploration)
        Rank the vertices in a random order.
Step 6  (Unperturbed exploration)
        (a) opt:= true.
        (b) According to the above order, for all vertex $i$, if $i$ is not in the best position,
            opt:= false and $\Gamma \leftarrow \Gamma_i^*$.
Step 7  (Termination for unperturbed exploration)
        (a) If opt is false, then go to Step 6.
        (b) Otherwise, if the objective function value of $\Gamma$ is larger than that of $\Gamma^*$, then
            $\Gamma^* \leftarrow \Gamma$ (Update the best solution).
Step 8  (Termination and update of $r$)
        (a) $r \leftarrow r - d$.
        (b) If $r > 0$, then go to Step 2. Otherwise, output $\Gamma^*$ and terminates.
```

In our algorithm, we set $r_{max} \leftarrow 0.5$ and $d \leftarrow 0.05$. See Charon and Hudry [4] for the details.

5. Pegging test

We start this section with the explanation of an ordinary pegging test, and then present an idea for an improvement. For the sake of simplicity we abbreviate $(LR(U, V, W, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}))$ to $(LR(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}))$ within this section.

5.1. Ordinary pegging test.

By the information obtained from the optimal solution of the Lagrangian relaxation problem $(LR(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}))$ we can see which variable takes one and which takes zero at the optimal solution of the problem (P).

Let us choose $(s, t) \in N^2_<$ and suppose that the problem (P) has an optimal solution with $x_{st} = \xi$, where ξ is either zero or one. Then the problem (P) with an additional constraint $x_{st} = \xi$ is equivalent to the problem (P) in the sense that optimal values of the two problems coincide. Suppose further we have an incumbent value ω_{low}. Then clearly

$$\omega(P|x_{st} = \xi) = \omega(P) \geq \omega_{\text{low}}.$$

Since the problem $(LR(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})|x_{st} = \xi)$ is a relaxation problem of the problem $(P|x_{st} = \xi)$ we obtain

$$\omega(LR(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})|x_{st} = \xi) \geq \omega(P|x_{st} = \xi),$$

hence

$$\omega(LR(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})|x_{st} = \xi) \geq \omega_{\text{low}}.$$

Lemma 5.1. *Let ξ be either zero or one. If $\omega(LR(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})|x_{st} = \xi) < \omega_{\text{low}}$, then $x_{st} = 1 - \xi$ for any optimal solution of the problem (P).*

Proof. The proof is clear from the above discussion. □

Suppose that we have an optimal solution $\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ of the problem $(LR(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}))$ and that $x(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})_{st} = 0$. By a simple calculation we see that

$$\omega(LR(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})|x_{st} = 1) = \omega(LR(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})) + r(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})_{st}. \tag{5.1}$$

Note that $x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st}=0$ implies $r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st}\le 0$. In the same way we see that

$$\omega(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})|x_{st}=0)=\omega(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))-r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st} \tag{5.2}$$

when $x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st}=1$. Note also that $r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st}>0$ in this case.

Theorem 5.2 (*ordinary pegging test*). *Let $\boldsymbol{x}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$ be an optimal solution of the Lagrangian relaxation problem $LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$. If*

$$\omega(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))-\omega_{\text{low}}<|r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st}|$$

holds, then $x^*_{st}=x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st}$ for any optimal solution $\boldsymbol{x}^*$ of the problem (P).

Proof. Substituting equation (5.1) or (5.2) for the condition in Lemma 5.1 will yield the assertion. □

We say that the variable x_{st} is *pegged* at $x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st}$ when the case holds in the theorem.

5.2. **Improved pegging test.** As the computation goes, we will have several variables pegged. Let P_0 and P_1 denote the index sets of the variables that have been pegged at zero and one, respectively. Given Lagrangian multiplier vector $(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$, the problem

$(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w},P_0,P_1))$:

$$\left|\begin{array}{lll}
\text{maximize} & \displaystyle\sum_{(i,j)\in N^2_<} c_{ij}x_{ij} & +\displaystyle\sum_{(i,j,k)\in U} u_{ijk}(1-x_{ij}-x_{jk}+x_{ik}) \\
 & & +\displaystyle\sum_{(i,j,k)\in V} v_{ijk}(1-x_{ij}+x_{jk}-x_{ik}) \\
 & & +\displaystyle\sum_{(i,j,k)\in W} w_{ijk}(1+x_{ij}-x_{jk}-x_{ik}) \\
\text{subject to} & x_{ij}\in\{0,1\} & \text{for all } (i,j)\in N^2_<, \\
 & x_{ij}=\begin{cases}0\\1\end{cases} & \begin{array}{l}\text{for all } (i,j)\in P_0,\\ \text{for all } (i,j)\in P_1.\end{array}
\end{array}\right.$$

is a relaxation problem of the problem (P).

Let $E(P_1)$ be the set of edges $\{i,j\}$ such that x_{ij} or x_{ji} has been pegged at one, i.e.,

$$E(P_1):=\{\,\{i,j\}\in E_n\mid (i,j)\in P_1 \text{ or } (j,i)\in P_1\,\}. \tag{5.3}$$

Definition 5.3. Given P_1 and $i,j\in N$ with $i\ne j$, we say that i is *connected* to j when there is a path from i to j on the edge set $E(P_1)$.

Definition 5.4. Given $(s,t)\in N^2_<\setminus(P_0\cup P_1)$ let

$$\begin{aligned}
S_= &:= \{s\}\cup\{\,i\in N\mid i \text{ is connected to } s\,\},\\
T_= &:= \{t\}\cup\{\,j\in N\mid j \text{ is connected to } t\,\},\\
ST_= &:= (S_=\times T_=)\cup(T_=\times S_=).
\end{aligned}$$

Take a variable x_{st} that has not yet been pegged, i.e., $(s,t)\in N^2_<\setminus(P_0\cup P_1)$, and fix x_{st} temporarily to one. Then every element connected to s should be connected to every element connected to t by the transitivity. Namely, the variables must satisfy

$$x_{ij}=1 \quad \text{for all } (i,j)\in ST_=\cap N^2_<. \tag{5.4}$$

When x_{st} is fixed temporarily to zero, we have similarly

$$x_{ij}=0 \quad \text{for all } (i,j)\in ST_=\cap N^2_<. \tag{5.5}$$

Now given nonnegative multiplier vectors $\boldsymbol{u}$, $\boldsymbol{v}$ and $\boldsymbol{w}$ we define

$(LR(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}, P_0, P_1)|x_{st} = 1)$:

$$\left|\begin{array}{lll} \text{maximize} & \displaystyle\sum_{(i,j)\in N^2_<} c_{ij}x_{ij} & + \displaystyle\sum_{(i,j,k)\in U} u_{ijk}(1 - x_{ij} - x_{jk} + x_{ik}) \\ & & + \displaystyle\sum_{(i,j,k)\in V} v_{ijk}(1 - x_{ij} + x_{jk} - x_{ik}) \\ & & + \displaystyle\sum_{(i,j,k)\in W} w_{ijk}(1 + x_{ij} - x_{jk} - x_{ik}) \\ \text{subject to} & x_{ij} \in \{0,1\} & \text{for all } (i,j) \in N^2_<, \\ & x_{ij} = \begin{cases} 0 \\ 1 \end{cases} & \begin{array}{l} \text{for all } (i,j) \in P_0, \\ \text{for all } (i,j) \in (ST_= \cap N^2_<) \cup P_1. \end{array} \end{array}\right.$$

This is a relaxation problem of the problem (P) with a temporary constraint $x_{st} = 1$ added.

Lemma 5.5. *If $(ST_= \cap N^2_<) \cap P_0 \neq \emptyset$, then $x^*_{st} = 0$ for any optimal solution $\boldsymbol{x}^*$ of the problem (P).*

Proof. Suppose there is an element (i,j) in the set $(ST_= \cap N^2_<) \cap P_0$. If $(i,j) = (s,t)$, since $(s,t) \in P_0$, the assertion holds. If $i \neq s$, by the transitivity constraint, we see that the variable corresponding to $\{i,s\}$ must be one for any optimal solution. In a similar way, if $j \neq t$, we see that the variable corresponding to $\{j,t\}$ must be one for any optimal solution. Thus, since $x_{ij} \in P_0$, to meet the transitivity constraint, x_{st} must be zero for any optimal solution. $\square$

When x_{st} is temporarily fixed to zero, we have the following problem and lemma.

$(LR(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}, P_0, P_1)|x_{st} = 0)$:

$$\left|\begin{array}{lll} \text{maximize} & \displaystyle\sum_{(i,j)\in N^2_<} c_{ij}x_{ij} & + \displaystyle\sum_{(i,j,k)\in U} u_{ijk}(1 - x_{ij} - x_{jk} + x_{ik}) \\ & & + \displaystyle\sum_{(i,j,k)\in V} v_{ijk}(1 - x_{ij} + x_{jk} - x_{ik}) \\ & & + \displaystyle\sum_{(i,j,k)\in W} w_{ijk}(1 + x_{ij} - x_{jk} - x_{ik}) \\ \text{subject to} & x_{ij} \in \{0,1\} & \text{for all } (i,j) \in N^2_<, \\ & x_{ij} = \begin{cases} 0 \\ 1 \end{cases} & \begin{array}{l} \text{for all } (i,j) \in (ST_= \cap N^2_<) \cup P_0, \\ \text{for all } (i,j) \in P_1. \end{array} \end{array}\right.$$

Lemma 5.6. *If $(ST_= \cap N^2_<) \cap P_1 \neq \emptyset$, then $x^*_{st} = 1$ for any optimal solution $\boldsymbol{x}^*$ of the problem (P).*

Since the problem $(LR(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}P_0, P_1)|x_{st} = \xi)$ is a relaxation problem of the problem (P) with a constraint $x_{st} = \xi$ added, we readily see the following lemma.

Lemma 5.7. *Let ξ be either zero or one, and let x_{st} be a variable that has not been pegged, i.e., $(s,t) \in N^2_< \setminus (P_0 \cup P_1)$. If $\omega(LR(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}, P_0, P_1)|x_{st} = \xi) < \omega_{\text{low}}$, then $x_{st} = 1 - \xi$ for any optimal solution of the problem (P).*

We have seen in (5.1) and (5.2) that

$$\omega(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})) - \omega(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})|x_{st} = 1 - x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st}) = |r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st}|$$

holds. Namely, the objective function value deteriorates by $|r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st}|$ when the constraint $x_{st} = 1 - x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st}$ is added to the problem $(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w}))$. In the similar manner we see that

$$\omega(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})) - \omega(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w},P_0,P_1)|x_{st} = 1 - x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st}) = \sum_{(i,j)\in R} |r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij}|,$$

where

$$\begin{aligned} &R = ((ST_{=} \cap N^2_{<}) \cup P_1) \cap \{(i,j) \mid x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij} = 0\} \\ &\quad \text{when } x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st} = 0, \text{and} \\ &R = ((ST_{=} \cap N^2_{<}) \cup P_0) \cap \{(i,j) \mid x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij} = 1\} \\ &\quad \text{when } x(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st} = 1. \end{aligned} \tag{5.6}$$

The first subset of R of (5.6) corresponds to the variables that should be one but takes zero at $\boldsymbol{x}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$, and the second subset to those that should be zero but takes one at $\boldsymbol{x}(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$.

Theorem 5.8 (*improved pegging test*). *Let ξ be either zero or one, and let x_{st} be a variable that has not been pegged, i.e., $(s,t) \in N^2_{<} \setminus (P_0 \cup P_1)$. If*

$$\omega(LR(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})) - \omega_{\text{low}} < \sum_{(i,j)\in R} |r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij}|$$

holds, then $x_{st} = 1 - \xi$ for any optimal solution of the problem (P).

Proof. The proof is clear from the above discussion. □

Compared to the ordinary pegging test in Theorem 5.2, since the set R always contains (s,t), we see that

$$|r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st}| \le \sum_{(i,j)\in R} |r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij}|.$$

Thus, given a multiplier vector $(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})$, if a variable can be pegged by the ordinary pegging test then the variable can also be pegged by the improved pegging test. In addition, the difference between $|r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{st}|$ and $\sum_{(i,j)\in R} |r(\boldsymbol{u},\boldsymbol{v},\boldsymbol{w})_{ij}|$ would be significant when we have many variables pegged.

5.3. **Post-processing of the pegging test.**

When x_{ij} and x_{jk} have been pegged at one by the pegging test, x_{ik} should be pegged at one to meet the transitivity constraint. We do this post-processing by finding the connected components of the undirected graph $(N, E(P_1))$ through the depth-first search.

Step 1 Find the connected components $\{C_1, C_2, \dots, C_k\}$ of the graph $(N, E(P_1))$ by the depth-first search.

Step 2 For all pairs i and j with $1 \le i < j \le n$, if they are in the same component in $\mathcal{C}$ then we peg x_{ij} at one.

Further, suppose x_{ij} is pegged at zero and i and j are in different components C_p and C_q respectively. From the transitivity constraints, we can peg $x_{i'j'}$ at zero for all pairs (i',j') in either $C_p \times C_q$ or $C_q \times C_p$.

Step 3 For all pairs p and q with $1 \le p < q \le k$, if there exists a variable x_{ij} which is pegged at zero with $(i,j) \in C_p \times C_q$ or $(i,j) \in C_q \times C_p$ then we peg $x_{i'j'}$ at zero for all pairs (i',j') in either $C_p \times C_q$ or $C_q \times C_p$.

Once we execute the above three steps, the two sets in Definition 5.4 are readily obtained by

$$S_= := \{s\} \cup \{\, i \in N \mid \{s,i\} \in \bar{E}\,\},$$
$$T_= := \{t\} \cup \{\, j \in N \mid \{t,j\} \in \bar{E}\,\}.$$

When some variables are newly pegged, we might peg some other variables by applying the above three steps. Thus, we apply the three steps after every pegging test in our algorithm. See the next section for the details.

6. Our algorithm

The algorithm we propose in this paper is a combination of the MSM (described in Section 3) and the pegging test (described in Section 5). We abbreviate it to *MSMPT*.

Modified Subgradient Method with Pegging Test (MSMPT)

Step 1 (Initialization)

(a) $k, l, q \leftarrow 0$, $\mu \leftarrow 2.0$, $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}) \leftarrow (\mathbf{0}, \mathbf{0}, \mathbf{0})$.

(b) $\omega_{\text{low}} \leftarrow$ the objective value of a feasible solution obtained by FVD. $\omega_{\text{up}}, \bar{\omega}_{\text{up}} \leftarrow \omega((LR(\emptyset, \emptyset, \emptyset, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}))$ and $(U, V, W) \leftarrow$ the transitivity constraints violated by $\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$.

(c) $P_0, P_1 \leftarrow \emptyset$.

Step 2 (Solving $(LR(U, V, W, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}, P_0, P_1))$

(a) Compute $r(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})_{ij}$ by (3.6) and set $x(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})_{ij}$ according to (3.7).

(b) $\omega_{\text{up}} \leftarrow \min\{\omega_{\text{up}}, \omega(LR(U, V, W, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}, P_0, P_1))\}$. If ω_{up} is not improved, $l \leftarrow l + 1$. Otherwise, $l \leftarrow 0$.

Step 3 (Optimality check)

If $\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ satisfies the optimality condition in Corollary 3.4 with $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ (in this case, $\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ is optimal for the problem (P)), then terminate.

Step 4 (Update of μ)

(a) If $\mu \leq 0.005$, then $\mu \leftarrow 2.0$ and go to Step 6 (we decide that there is no chance of improving the upper bound unless we expand (U, V, W)).

(b) If l reaches 5, $\mu \leftarrow \mu/2$.

Step 5 (Update of $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$)

Update $(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ according to (3.3), (3.4) and (3.5) and go to Step 2.

Step 6 (Lagrangian heuristics and local search)

Apply the Lagrangian heuristics and local search to $\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ for a better solution $\tilde{\boldsymbol{x}}$. $\omega_{\text{low}} \leftarrow \max\{\omega_{\text{low}}, \text{objective function value of } \tilde{\boldsymbol{x}}\}$.

Step 7 (Pegging test)

(a) If $|P_0| = |P_1| = 0$, then apply the ordinary pegging test (Theorem 5.2). Otherwise, apply the improved pegging test (Theorem 5.8).

(b) Append the indices corresponding variables are newly pegged to P_0 and P_1 respectively.

(c) Apply the post-processing described in Section 5.

Step 8 (Termination)

(a) If the improvement of ω_{up} compared to $\bar{\omega}_{\text{up}}$ is less than 0.1% or 1 in value then $q \leftarrow q + 1$. $\bar{\omega}_{\text{up}} \leftarrow \omega_{\text{up}}$

(b) If $q > 10$ then go to Step 10.

Step 9 (Update of (U, V, W))

(a) Find the transitivity constraints violated by $\boldsymbol{x}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w})$ and add them to U, V and W.

(b) $u_{ijk}, v_{ijk}, w_{ijk} \leftarrow 0$ for newly added indices (i,j,k) and go to Step 2.

Since the pegged variables contribute to downscaling the feasible region of the Lagrangian relaxation problem, possibly the more we peg the variables, the tighter the upper bound ω_{up} would be. In the following section, we will report numerical experiments of the MSMPT.

7. Numerical experiments

In this section, we report numerical experiments with our algorithm described in Section 6. We coded the algorithm in Java, and run it on a PC with an Intel i3, 3.33 GHz processor and 2 GB of memory.

Since all of the instances we solved have integral objective function, we stop the MSMPT if the duality gap fell to below 1 in value (in this case an optimal solution is found).

7.1. **Real-world instances.** The real-world instances we solved are problems of *aggregation of equivalence relations*, which are available from Grötschel and Wakabayashi [8].

Suppose we have q different equivalence relations $\sim^1, \sim^2, \dots, \sim^q$ on N, and want to aggregate them into a single equivalence relation $\sim^X$ on N. Here, we define the "distance" $d(\sim^k, \sim^X)$ of two equivalence relations $\sim^k$ and $\sim^X$ as the total number of disagreement of $\sim^k$ with $\sim^X$ i.e.,

$$d(\sim^k, \sim^X) := |\{\{i,j\} \mid (i \sim^k j \wedge i \not\sim^X j) \vee (i \not\sim^k j \wedge i \sim^X j)\ \}|.$$

By representing an equivalence relation $\sim^1, \sim^2, \dots, \sim^q$ and $\sim^X$ as a binary vector i.e.,

$$r_{ij}^k = \begin{cases} 1 & i \sim^k j, \\ 0 & \text{otherwise,} \end{cases} \quad \text{for all } k \in \{1,2,\dots,q\} \text{ and } x_{ij} = \begin{cases} 1 & i \sim^X j, \\ 0 & \text{otherwise,} \end{cases}$$

we see that the quantity $d(\sim^k, \sim^X)$ can be given by

$$\begin{aligned} d(\sim^k, \sim^X) &= \sum_{(i,j) \in N_<^2} \left(r_{ij}^k - x_{ij}\right)^2 \\ &= \sum_{(i,j) \in N_<^2} r_{ij}^k - 2r_{ij}^k x_{ij} + x_{ij} \ (\because r_{ij}^k, x_{ij} \in \{0,1\}) \\ &= \sum_{(i,j) \in N_<^2} r_{ij}^k + \left(1 - 2r_{ij}^k\right) x_{ij}. \end{aligned}$$

The problem of aggregation of equivalence relations is to find an equivalence relation $\sim^X$ minimizing the total dissimilarity

$$\begin{aligned} \sum_{k=1}^q d(\sim^k, \sim^X) &= \sum_{k=1}^p \left(\sum_{(i,j) \in N_<^2} r_{ij}^k + (1 - 2r_{ij}^k) x_{ij} \right) \\ &= \sum_{k=1}^p \sum_{(i,j) \in N_<^2} r_{ij}^k + \sum_{k=1}^p \sum_{(i,j) \in N_<^2} (1 - 2r_{ij}^k) x_{ij} \\ &= c + \sum_{(i,j) \in N_<^2} \left(\sum_{k=1}^p (1 - 2r_{ij}^k) \right) x_{ij} \\ &= c + \sum_{(i,j) \in N_<^2} c_{ij} x_{ij}. \end{aligned}$$

TABLE 1. Result for the instances described in Grötschel and Wakabayashi [8]

instance	n	$\lvert N^2_<\rvert$	$\lvert N^3_<\rvert$	relative gap	$\%^\blacktriangle$	$\%^\bigstar$	time(s)
A1	36	630	21,420	0	1.18	0	0.01
A2	30	435	12,180	0	6.08	91.49	0.04
A3	34	561	17,952	1.E-03	13.27	91.44	1.96
A4	33	528	16,368	0	14.40	0	0.16
A6	54	1,431	74,412	0	3.14	0	0.03
A7.1	158	12,403	1,934,868	0	0.50	0	0.14
A7.2	158	12,403	1,934,868	0	0.92	0	0.43

To be an equivalence relation, $(x_{ij})_{(i,j)\in N^2_<}$ must satisfy all the inequality constraints of the problem (P). Since minimizing $\sum_{k=1}^{p} d(\sim^k, \sim^X)$ is equivalent to maximizing $-\sum_{k=1}^{q} d(\sim^k, \sim^X)$, we see that the problem of aggregation of equivalence relations reduces to the problem (P).

Table 1 shows the result of the MSMPT for the real-world instances in Grötschel and Wakabayashi [8] where

- the relative gap is the ratio of the duality gap to the best lower bound (i.e., $(\omega_{\text{up}} - \omega_{\text{low}})/\omega_{\text{low}}$),
- $\%^\blacktriangle$ shows the percentage of the transitivity constraints considered in the MSMPT (i.e., $100(|U| + |V| + |W|)/|N^3_<|$),
- $\%^\bigstar$ shows the percentage of the pegged variables in the MSMPT (i.e., $100(|P_0| + |P_1|)/|N^2_<|$).

Optimal solution was obtained within 0.5 seconds for all the instances except instance A3. From the column $\%^\blacktriangle$, we observe that the MSM works well. In addition, we observed that FVD finds an optimal solution in Step 1 for all the instances except instance A1. Also we see that only for instance A1, our Lagrangian heuristics (BLH) contributes to finding an optimal solution. Note that the instance A6 and A7 have too huge number of inequality constraints to apply the BSM on an ordinary workstation.

Figure 1 shows the comparison of our algorithm MSMPT and the cutting plane algorithm described in Grötschel and Wakabayashi [8](GW for short). We cannot say about which algorithm is more efficient only from the computational times since the computational environment has changed dramatically. However, at least we can judge from the figure that the computation time of the MSMPT is less sensitive to the increase of the problem size n than that of the cutting plane algorithm GW. It should be pointed out that for instance A3, the cutting plane algorithm GW finds an optimal solution while the MSMPT terminates with a narrow duality gap remained.

7.2. **Randomly generated instances.** In this section, we report numerical experiments for randomly generated instances. The instances are generated by the following method.

Step 1 Decide the problem size n and the number of equivalence relations q.

Step 2 Make a table $D = (d_{ik})$ of dimension $n \times p$ by choosing d_{ik} randomly from the uniform distribution on $\{0, 1, 2\}$.

Step 3 Make equivalence relations $\sim^1, \sim^2, \ldots, \sim^q$ by setting

$$r^k_{ij} := \begin{cases} 1 & d_{ik} = d_{jk}, \\ 0 & \text{otherwise}, \end{cases} \quad \text{for all } k \in \{1, 2, \ldots, q\}.$$

Step 4 Set $c_{ij} \leftarrow -\sum_{k=1}^{q}(1 - 2r^k_{ij})$ for all $(i, j) \in N^2_<$.

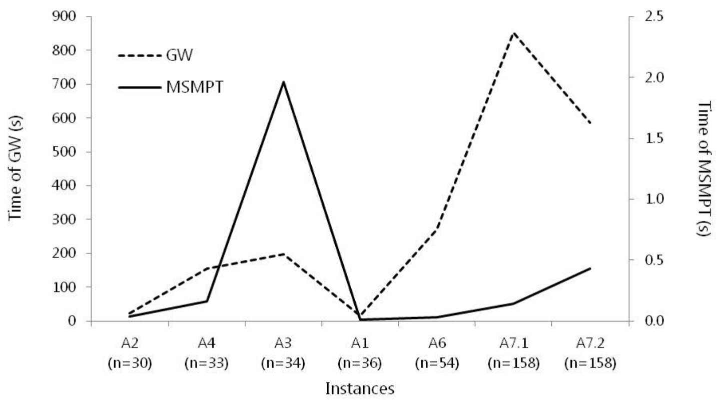

Figure 1. Comparison of the MSMPT with the cutting plane algorithm described in Grötschel and Wakabayashi [8] (GW for short)

Table 2. Result for the random instances ($n = 30, 50, 100$)

	MSMPT				Xpress	
instance	relative gap	$\%^{\blacktriangle}$	$\%^{\bigstar}$	time(s)	ratio	time(s)
R30a	4.E-02	1.158	57.701	0.25	1.000	1.01
R30b	2.E-02	1.026	84.368	0.11	1.000	1.03
R30c	2.E-02	0.591	82.989	0.10	1.000	1.01
R30d	0	0.435	96.552	0.02	1.000	1.03
R30e	0	1.190	0.000	0.00	1.000	1.01
R50a	3.E-02	0.777	32.163	0.51	1.000	4.76
R50b	0	0.855	60.327	0.08	1.000	4.90
R50c	1.E-01	2.214	0.000	0.49	1.000	4.76
R50d	3.E-02	0.968	32.082	0.52	1.000	4.77
R50e	3.E-02	1.776	52.653	1.02	1.000	4.77
R100a	2.E-01	2.648	0.000	4.35	1.000	43.96
R100b	2.E-01	2.616	0.000	4.47	1.000	46.32
R100c	2.E-01	2.362	0.000	5.83	0.996	47.55
R100d	2.E-01	2.806	0.000	4.52	1.000	45.94
R100e	2.E-01	2.653	0.000	4.82	1.000	43.93

For each n = 30, 50, 100, we generated five problems, all of which the number of equivalence relations q is fixed to 10. The result is summarized in Table 2. As mentioned in Section 3, since one of the possible competitors of the MSM is the dual of the LP relaxation, here we also presented the result of the dual of the LP relaxation. To solve it, we used Xpress Optimizer 21.01.06 to solve them and run on a PC with an Intel i7, 2.80GHz processor and 6 GB of memory. The column "ratio" shows the ratio of the best upper bound obtained by the MSMPT and the optimal value of the dual of the LP relaxation.

We observe that the MSMPT fails to find an optimal solution for almost all of the random instances. In addition, the pegging test does not work very well especially for the instances with $n = 100$. However, we see that the MSMPT excels the dual of the LP relaxation in computational times and upper bounds. This shows that the MSMPT has the potential to replace the linear programming relaxation in exact methods such as the branch-and-bound method.

Table 3. Similarity of the instances

A1	A2	A3	A4	A6	A71	A72
-	0.765	0.738	0.780	0.696	0.907	0.861

The real-world instances would be different from randomly generated instances. To examine the difference quantitatively, we define the degree of similarity of given equivalence relations $\sim^1, \sim^2, \dots, \sim^q$. Let $sim(s,t)$ denote the cosine similarity of $\sim^s$ and $\sim^t$ with $1 \le s < t \le q$, i.e.,

$$sim(s,t) := \frac{\displaystyle\sum_{(i,j)\in N^2_<} r^s_{ij} r^t_{ij}}{\sqrt{\displaystyle\sum_{(i,j)\in N^2_<} (r^s_{ij})^2}\sqrt{\displaystyle\sum_{(i,j)\in N^2_<} (r^t_{ij})^2}},$$

and sim_{all} denote the geometric mean of the cosine similarities, i.e.,

$$sim_{all} := (sim(1,2) \times sim(1,3) \times \cdots \times sim(q-1,q))^{\frac{1}{q(q-1)/2}}.$$

Note that, in this case, $0 \le sim(s,t) \le 1$ holds for all $1 \le s < t \le q$. If the value sim_{all} is close to 1, then one could say that the given equivalence relations are similar. The value sim_{all} of the random instances we solved is about 0.575 on average. Table 3 shows the value sim_{all} of the instances in Grötschel and Wakabayashi [8]. Similarity is not available for `A1` since it has some missing values.

Judging from the similarity defined above, the real-world instances are likely to be easier than the random instances. Keeping this point in mind, we generate random instances which imitates the real-world instances in the following way.

Step 1 Decide the problem size n and the number of equivalence relations q.

Step 2 Make an n-dimensional vector (d_{i1}) by choosing d_{i1} randomly from the uniform distribution on $\{0,1,2\}$ (this vector will be used as a reference equivalence relation).

Step 3 For each $k \in \{2,3,\dots,q\}$ and for each $i \in \{1,2,\dots,n\}$, choose p randomly from the uniform distribution in $[0,1]$, and if $p \ge \bar{p}$ then set $d_{ik} \leftarrow d_{i0}$, otherwise set randomly from the uniform distribution in $\{0,1,2\}$.

Step 4 Make equivalence relations $\sim^1, \sim^2, \dots, \sim^q$ by setting

$$r^k_{ij} := \begin{cases} 1 & d_{ik} = d_{jk}, \\ 0 & \text{otherwise}, \end{cases} \quad \text{for all } k \in \{1,2,\dots,q\}.$$

Step 5 Set $c_{ij} \leftarrow -\displaystyle\sum_{k=1}^{q}(1 - 2r^k_{ij})$ for all $(i,j) \in N^2_<$.

The parameter $\bar{p}$ controls the value sim_{all}. Namely, the smaller $\bar{p}$ we set, the greater the value sim_{all} will be. We set $\bar{p}$ to 0.4 and generated five problems for each $n = 100$, 200, 300. The result is summarized in Table 4. Note that the instances with $n = 300$ have 44,850 decision variables and 13,365,300 inequality constraints. We observe that the MSMPT finds an optimal solution for almost all these instances. This result shows that the MSMPT has a potential to solve larger real-world instances. Even when the MSMPT fails to find an optimal solution, we have problems of reduced size at hand. Then we could solve them by an appropriate solver. In the next section, we will discuss the performance of the MSMPT as a preprocessing for an exact method.

Table 4. Result for the random instances with similarity ($n = 100, 200, 300$)

instance	sim_{all}	relative gap	$\%^{\blacktriangle}$	$\%^{\bigstar}$	time (s)
RS100a	0.65	0	1.125	0.000	0.17
RS100b	0.68	0	1.199	9.354	0.50
RS100c	0.66	0	1.193	0.000	0.06
RS100d	0.67	0	1.372	0.000	0.49
RS100e	0.66	0	1.145	98.101	0.93
RS200a	0.67	0	1.164	7.367	4.65
RS200b	0.68	0	1.090	0.000	5.52
RS200c	0.67	0	1.228	0.000	6.85
RS200d	0.68	0	1.087	99.030	16.80
RS200e	0.66	6.E-05	1.118	99.467	140.34
RS300a	0.67	0	1.143	0.000	40.47
RS300b	0.67	0	1.133	100.000	159.57
RS300c	0.66	0	1.221	98.370	240.66
RS300d	0.67	0	1.171	0.000	53.48
RS300e	0.67	3.E-05	1.160	99.802	469.64

Table 5. Comparison of (n', m') and $(|N^2_<|, |N^3_<|)$

instance	n	$\lvert N^2_<\rvert$	$\lvert N^3_<\rvert$	$n'/\lvert N^2_<\rvert$	$m'/\lvert N^3_<\rvert$
A3	34	561	17,952	9.E-02	5.E-02
RS200e	200	19,900	3,940,200	5.E-03	9.E-04
RS300e	300	44,850	13,365,300	2.E-03	5.E-05

Table 6. Result of Xpress for the reduced problems

	MSMPT		Xpress		total
instance	ω_{low}	ω_{up}	optimal value	time(s)	time(s)
A3	1042.00	1043.00	1042.00	0.04	2.00
RS200e	17238.00	17239.00	17238.00	0.35	140.69
RS300e	36714.00	36715.00	36714.00	0.73	470.37

7.3. **MSMPT as a preprocessing.** Once some variables are pegged, some of the transitivity constraints become redundant. Take the transitivity constraint $x_{ij} + x_{jk} - x_{ik} \leq 1$ of type U for instance. If x_{ik} is pegged at one, this inequality is redundant. It is also redundant if x_{ij} or x_{jk} is pegged at zero. If x_{ij} is pegged at one and x_{ik} is pegged at zero, the third variable x_{jk} must take zero to meet the transitivity constraint. Then we can peg it at zero and remove the constraint. Repeating this procedure, we can reduce the size of the problem. Let n' and m' denote the number of the decision variables and the transitivity constraints of the reduced problem, respectively. Table 5 shows how the MSMPT reduces the size of the instances which we reported the MSMPT failed to solve to optimality.

We observe that the reduction ratio of the transitivity constraints is better than that of the decision variables. To solve the reduced problems exactly, we used Xpress Optimizer 21.01.06 and run on a PC with an Intel i7, 2.80GHz processor and 6 GB of memory. Table 6 shows the result.

Although the MSMPT terminates with a small duality gap to close up for instance RS200e and RS300e, the best solutions found by the MSMPT turned out to be optimal solutions. We observe that the computation time of Xpress is very short due to the problem reduction. We conclude that the MSMPT deserves a good preprocessing for exact methods.

8. Conclusion

In order to manage the high dimensionality of the Lagrangian multipliers arising from the $O(n^3)$ number of constraints, we proposed in this paper a modification of the conventional subgradient method. This enables us to apply the Lagrangian relaxation method to large instances that exceed the capability of the conventional method. Since the method reached the optimal value of the Lagrangian dual problem in almost all instances that we tested, this modification is practical as a tool for finding a good upper bound of the clique partitioning problem.

Taking advantage of the structural property of the clique partitioning problem, we made an improvement on the ordinary pegging test, which was experimentally confirmed to peg more variables than the ordinary one. We showed that a slightly complicated computation of the improved pegging test is performed efficiently.

Our numerical experiments showed that the algorithm works well for moderate-sized instances of the aggregation problem of equivalence relations, and for even larger instances when the equivalence relations are correlated.

References

[1] S. G. de Amorim, J-P. Barthélemy and C. C. Riberio, “Clustering and clique partitioning: simulated annealing and tabu search approaches”, *Journal of Classification* **9** (1992) 17–41.

[2] E. Balas and M. C. Carrera, “A dynamic subgradient-based branch-and-bound procedure for set covering”, *Operations Research* **44** (1996) 875–890.

[3] M. J. Brusco and H. F. Köhn, “Clustering qualitative data based on binary equivalence relations: neighborhood search heuristics for the clique partitioning problem”, *PSYCHOMETRIKA* **74** (2009) 685–703.

[4] I. Charon and O. Hudry, “Noising methods for a clique partitioning problem”, *Discrete Applied Mathematics* **154** (2006) 754–769.

[5] U. Dorndorf, F. Jaehn and E. Pesch, “Modeling robust flight-gate scheduling as a clique partitioning problem”, *Transportation Science* **42** (2008) 292–301.

[6] M.L. Fisher, “The Lagrangian relaxation method for solving integer programming problems”, *Management Science* **27** (1981) 1–18.

[7] A.M. Geoffrion, “Lagrangian relaxation for integer programming”, *Mathematical Programming Study* **2** (1974) 82–114.

[8] M. Grötschel and Y. Wakabayashi, “A cutting plane algorithm for a clustering problem”, *Mathematical Programming* **45** (1989) 59–96.

[9] G. Kochenberger, F. Glover, B. Alidaee and H. Wang, “Clustering of microarray data via clique partitioning”, *Journal of Combinatorial Optimization* **10** (2005) 77–92.

[10] T. Larsson, M. Patriksson and A.-B. Strömberg, “Conditional subgradient optimization - Theory and applications”, *European Journal of Operational Research* **88** (1996) 382–403.

[11] A. Mehrotra and M. A. Trick, “Cliques and clustering: A combinatorial approach”, *Operations Research Letters* **22** (1998) 1–12

[12] R. M. Nauss “An efficient algorithm for the 0-1 knapsack problem”, *Management Science* **23** (1976) 27–31.

[13] M. Oosten, J. H. G. C. Rutten and F. C. R. Spieksma, “The clique partitioning problem: Facets and patching facets”, *Networks* **38** (2001) 209–226.

[14] B. You and T. Yamada “A pegging approach to the precedence-constrained knapsack problem”, *European Journal of Operational Research* **183** (2007) 618–632

Faculty of Engineering, Information and Systems, University of Tsukuba, Tsukuba, Ibaraki 305-8573, Japan

E-mail address: sukegawa.n.aa@m.titech.ac.jp, yamamoto@sk.tsukuba.ac.jp, zhangliyuan120@gmail.com