INFORMATICA, 2020, Vol. 31, No. 1, 1–20

DOI: https://doi.org/10.15388/20-INFOR391

A Complementary Column Generation Approach for the Graph Equipartition Problem

Salem M. AL-YKOOB[1,*], Hanif D. SHERALI[2]

[1] *Department of Mathematics, College of Science, Kuwait University, P.O. Box 5969, Safat 13060, State of Kuwait*

[2] *Grado Department of Industrial and Systems Engineering (0118), Virginia Tech, Blacksburg, VA 24061, USA*

e-mail: smalyakoob@gmail.com, hanifs@vt.edu

Received: May 2019; accepted: September 2019

Abstract. This paper investigates the problem of partitioning a complete weighted graph into complete subgraphs, each having the same number of vertices, with the objective of minimizing the sum of edge weights of the resulting subgraphs. This NP-complete problem arises in many applications such as assignment and scheduling-related group partitioning problems and micro-aggregation techniques. In this paper, we present a mathematical programming model and propose a complementary column generation approach to solve the resulting model. A dual based lower bounding feature is also introduced to curtail the notorious tailing-off effects often induced when using column generation methods. Computational results are presented for a wide range of test problems.

Key words: graph partitioning, column generation, complementary column generation, mixed-integer programming.

1. Introduction and Motivation

In this paper, we study the problem of partitioning a complete weighted graph into complete subgraphs, each having the same number of vertices, with the objective of minimizing the total edge weights of the resulting subgraphs. This problem, denoted by GPP, is formally stated in Section 1.1 below, and Section 1.2 then presents some motivating examples.

1.1. *Statement of Problem GPP*

Consider a complete-weighted graph $G(V, E)$, where V and E, respectively, denote the set of vertices and edges of the graph G. Let $v = 1, 2, \ldots, |V|$ index the vertices of V, and for $v_1, v_2 \in V$ with $v_1 \neq v_2$, let $(v_1, v_2) \in E$ denote the edge joining v_1 and v_2 in G. Let $w(v_1, v_2) > 0$ denote the weight associated with the edge (v_1, v_2). Let nn be a positive integer and suppose that $\alpha = \frac{|V|}{n}$ is integer-valued. Hence, the set V can be partitioned into n subsets, each of which is composed of α vertices. Let P denote the set

*Corresponding author.

of all distinct subsets of V, each of which has $\alpha\alpha$ vertices, and let V_p denote the pth such vertex subset, $\forall p = 1, 2, \ldots, |P|$. An *n-partition* of V is a collection of n vertex subsets from P, say $V^{p_1}, V^{p_2}, \ldots, V^{p_n}$, satisfying $\bigcup_{i=1}^{n} V_{p_i} = V$, and $V_{p_i} \cap V_{p_j} = \emptyset$, $\forall i, j \in \{1, \ldots, n\}$ with $i \neq j$. Let Q denote the set of all such *n-partitions*, indexed by $q = 1, \ldots, |Q|$, where the qth *n-partition* is given by $\{V_{p_k(q)}, k = 1, \ldots, n\}$. For any V_p, $p \in P$, let $w_p = \sum_{\substack{v_i, v_j \in V_p \\ i<j}} w(v_i, v_j)$, and accordingly, let $c_q \equiv \sum_{k=1}^{n} w_{p_k(q)}$ represent the cost of the qth *n-partition* for any $q \in Q$. Problem GPP then seeks an *n-partition* $q^* \in Q$ such that $c_{q^*} \leqslant c_q$, $\forall q \in Q$.

1.2. *Motivating Examples*

We provide two motivating examples for Problem GPP, where we have used specific numbers in lieu of generic notation for the purpose of illustration.

Example 1. Consider a firm that operates four work centres and needs to assign three employees to each centre (from a total of 12 available employees). For $i = 1, \ldots, 12$, let e_i denote the ith employee. Each employee quantifies ranked preferences for working with the other 11 employees from the set $\{1, \ldots, 11\}$, where a lower number rank indicates a higher preference. We construct a complete weighted graph having 12 vertices, where vertex v_i corresponds to employee e_i, $\forall i = 1, \ldots, 12$, and where the weight associated with the edge joining vertices v_i and v_j, $i, j \in \{1, \ldots, 12\}$, $i \neq j$, represents the sum of the preferences of employees e_i and e_j to work with each other. The problem of interest, then, is to partition the underlying graph into four complete subgraphs, each having three vertices, so that the total weight of the resulting complete subgraphs is minimal, thereby achieving a best aggregate preference.

Example 2. Consider a firm having 15 business branches that seeks to assign one of the available supervisors to each cluster of five branches. Since a supervisor assigned to any given cluster needs to frequently travel between the branches within the clusters, it is desired that the sum of the distances between the branches of a given cluster should be small. This problem can likewise be modelled as a complete graph partitioning problem having 15 vertices, each of which represents a branch, and with $n = 3$, where the edge weight associated with any pair of vertices is given by the distance between the corresponding branches.

1.3. *Contribution and Organization*

This paper proposes a column generation framework to solve Problem GPP with three enhancing features: (a) a complementary column generation scheme that uses a pricing problem to generate batches of columns; (b) a dual-based lower bound that can be employed to curtail the notorious tailing-off effects typically associated with column generation, and (c) the generation of a collection of vertex partitions that serves to determine a starting

basis for the proposed column generation framework, as well as assists in computing good quality feasible solutions.

The remainder of this paper is organized as follows. Section 2 presents literature related to the studied problem. In Section 3, we develop an integer mathematical programming model, denoted by GPM, for Problem GPP, which attempts to directly select a minimal cost *n-partition*. We then design an enhanced column generation approach (ECGH) in Section 4 to solve the linear relaxation of Model GPM, based on which we propose a heuristic procedure in Section 5 to solve Model GPM. Computational results are presented in Section 6, and we conclude the paper in Section 7 with a summary and some remarks, as well as future research extensions.

2. Related Literature

Several graph partitioning problems have been studied in the literature, which are motivated by applications in microaggregation (Domingo-Ferrer and Mateo-Sanz, 2002), political districting (Mehrotra *et al.*, 1998), video clustering (Schaeffer, 2007), telecommunication and VLSI design (Karypis *et al.*, 1999), biological or social networks (Fan *et al.*, 2009), and data mining (Zha *et al.*, 2001). Typically such problems arise in the context of clustering, which is an unsupervised classification and the clusters must sometimes satisfy certain additional threshold criteria (Fan and Pardalos, 2012). The general graph partitioning problem aims to partition the vertex set of a graph into several disjoint subsets with the objective of minimizing the sum of edge weights between the disjoint subsets (Fan and Pardalos, 2010). This is an NP-complete combinatorial optimization problem (Garey *et al.*, 1976) and different techniques were employed to solve it (Hager and Krylyuk, 1999). The case when the graph is partitioned into equal or different by 1 cardinalities for all partitions was solved either by linear programming (Lisser and Rendl, 2003) or semidefinite programming (Karisch and Rendl, 1998; Lisser and Rendl, 2003). Quadratic programming (Hager and Krylyuk, 1999, 2002) and semidefinite programming (Wolkowicz and Zhao, 1996) requires that the cardinalities of all partitions are known *a priori*. Fan and Pardalos (2010) extended this work by formulating a zero-one quadrating programming problem without the input of cardinalities of the required partitions. The objective of the problem studied in the current paper is to minimize the sum of edge weights of the resulting partitions while that in the general graph partitioning problem is to minimize the sum of edge weights between the disjoint partitions.

In the following, we discuss a number of applications that are addressed using different types of graph partitioning paradigms. Micro-aggregation is a technique used by statistical agencies, where some statistical information needs to be disclosed, while the related specific individual information must remain classified. Published data needs to be therefore presented in a manner such that: (a) the classified data cannot be concluded from the published data, and (b) the deleted unclassified data is minimized. Domingo-Ferrer and Mateo-Sanz (2002) used a graph partitioning approach to solve this problem. Political redistricting is another application of graph partitioning, where boundaries of districts

need to be drawn within the states to attain certain characteristics and to avoid partisan political goals. Mehrotra *et al.* (1998) designed a graph partitioning political redistricting model with the motivation that (a) differences in populations for any two different districts should be minimized in order to adhere to the one-person-one-vote principle; (b) districts should be contiguous, and (c) districts should be geographically compact. Graph partitioning is also used in video scene clustering (Tan and Lu, 2003) to index, browse, and retrieve video data. In this context, a graph $G(V, E)$ is constructed as follows, where each vertex $v \in V$ represents a scene and an edge $e \in E$ between two vertices indicates the similarity obtained from some defined relations of the colours of two scenes. The objective is to partition G with the goal of maximizing similarity in the individual partitions where the number of partitions is not restricted.

In telecommunication technology, graph partitioning is employed to subdivide a transmission network into clusters in order to maximize the routed traffic within the clusters (Laguna, 1994). Park *et al.* (2000) addressed the problem of clustering a telecommunication network into local networks and hub locations. Xiao *et al.* (2007) developed a graph partitioning model to cluster mobile units within mobile servers. In this case, a graph $G(V, E)$ is constructed where $v \in V$ represents a mobile unit and each edge $e \in E$ represents a communication link between two units, and where the weight assigned to the edge depends on some technical parameters including the bandwidth and the distance between the two vertices. Laguna (1994) used a graph partitioning model to enhance several design features and to overcome limitations of optical fiber networks within the telecommunication industry.

Graph partitioning has also been used to tackle scheduling problems. Carlson and Nemhauser (1966) developed a clustering model for a scheduling problem that involves several activities and facilities, where the problem is to cluster activities and then to assign them to the facilities so as to minimize interaction costs, given the cost of assigning pairs of activities to a facility. Salido *et al.* (2007) employed graph partitioning in railway scheduling to generate optimal schedules for trains, taking into consideration connection points, railway types, and train capacities, among other restrictions.

There exist several other such examples of graph partitioning problems that have been studied in the literature, e.g. see Ji (2004). These include the clique partitioning problem (Grotschel and Wakabayashi, 1989; 1990), the graph equipartitioning problem (Conforti and Rao, 1990a, 1990b), the capacitated graph partitioning problem (Mehrotra and Trick, 1998), the maximum balanced connected q-partition (Chlebikova, 1996; Salgado and Wakabayashi, 2004), and the minimum edge-cut graph partitioning problem (Donath and Hoffman, 1973; Goldschmidt and Hochbaum, 1988), among others. All of the above graph partitioning problems are NP-hard (Ji, 2004). In particular, similar to the problem considered in the present paper, the k-way graph equipartitioning problem is to partition the vertex set V into k subsets of equal size, with the objective of minimizing the total weight of edges that have both end-points in the same partitioned subset. Mitchell (2001) formulated a mathematical model for the k-way graph equipartitioning problem, investigated its polyhedral structure and presented a branch-and-cut algorithm to solve the resulting model. The algorithm in Mitchell (2001) was used to realign the National Football League (NFL). Results for partitioning 32 teams into eight groups with the objective

of minimizing the overall travel time among teams within each group were reported in Mitchell (2003). The sports realignment problem studied in Mitchell (2001, 2003) was revisited later along with other similar contextual problems by Xiaoyun and Mitchell (2005) who modelled the realignment of NBA, NHL, and NFL as k-way equipartition problems. A branch-and-price scheme with cutting plane features was designed and implemented to solve the resulting k-way equipartitioning problems, where the pricing problem was modelled as an integer program. Computational results indicated that the branch-and-price-and-cut scheme of Xiaoyun and Mitchell (2005) performed well on small-sized instances (with about 40 vertices). However, for larger test instances, the solution of the pricing problem turned out to be cumbersome to solve. Nonetheless, for such problems, the root algorithm was found to yield relatively good quality feasible solutions. The algorithm in Xiaoyun and Mitchell (2005) was also used to solve certain micro aggregation problems. The authors concluded that the performance of their proposed branch-and-price-and-cut approach was comparable to that of the price-and-cut method of Mitchell (2001, 2003).

3. Formulation of Model GPM

In this section, we formulate a model for Problem GPP, denoted GPM, which directly attempts to select a minimum-cost collection of n valid *partitions* from the set P in order to constitute an *n-partition*.

Model formulation

Define the following set of binary decision variables:

$$x_p = \begin{cases} 1 & \text{if partition } p \in P \text{ is selected,} \\ 0 & \text{otherwise.} \end{cases}$$

For a given partition $p \in P$, we define the following set of *parameters* that indicate whether a vertex $v \in V$ belongs to the associated vertex subset V_p or not:

$$\lambda_{v,p} = \begin{cases} 1 & \text{if } v \in V_p, \\ 0 & \text{otherwise.} \end{cases}$$

Note that the values of the parameters $\lambda_{v,p}$ are known *a priori* based on information derived from the corresponding subset V_p. Then, the following model determines a minimum-cost *n-partition*:

GPM: Minimize $\sum_{p \in P} w_p x_p,$

subject to

$$\sum_{p \in P} \lambda_{v,p} x_p = 1, \quad \forall v \in V, \tag{3.1}$$

$$\sum_{p \in P} x_p = n, \tag{3.2}$$

$$x_p \in \{0, 1\}, \quad \forall p \in P.$$

The objective function of GPM minimizes the overall weight of edges associated with the selected *n-partition*. Constraint (3.1) assures that each $v \in V$ belongs to exactly one valid partition. The required number of valid partitions (n) is enforced by Constraint (3.2). The continuous relaxation of Model GPM, denoted by $\overline{\text{GPM}}$, is given as follows:

$$\text{Minimize} \left\{ \sum_{p \in P} w_p x_p : \text{ (3.1) and (3.2), where } x_p \geqslant 0, \forall p \in P \right\}.$$

Note that $x_p \leqslant 1$ is implied by (3.1). The following structural result indicates that Constraint (3.2) can be deleted from the above model without affecting the solution, even in the continuous sense.

Proposition 1. *Constraint* (3.2) *is implied by Constraint* (3.1) *within Model GPM.*

Proof. Since $|V| = \alpha n$, summing (3.1) over all $v \in V$, we get $\sum_{v \in V} \sum_{p \in P} \lambda_{v,p} x_p = \alpha n$. But since $\sum_{v \in V} \lambda_{v,p} = \alpha$ because $|V_p| = \alpha$, $\forall\, p \in P$, this implies that $\sum_{p \in P} \alpha x_p = \alpha n$, or that (3.2) holds. □

Notwithstanding Proposition 1, we retain Constraint (3.2) in the model because of the lower bounding facility it provides for $\overline{\text{GPM}}$, which enables a useful practical stopping criterion when solving the latter problem (see Proposition 2 below).

Note that Model GPM attempts to directly select a minimal cost n-partition, i.e. a minimal cost collection of n valid partitions from the set P. An alternative modelling approach to solve Problem GPP is to designate decision variables that assign vertices to different subsets and to designate constraints to ensure the cardinality of each subsets. This modelling approach is the subject of a follow-on paper, where we will attempt to employ a Lagrangean-based decomposition scheme in concert with symmetry defeating strategies to solve Problem GPP. As it will be seen later, the formulation of Model GPM enabled us to devise a column generation algorithm to heuristically solve Problem GPP, which is the focus of the current paper.

4. An Enhanced Column Generation Approach to Solve Model $\overline{\text{GPM}}$

In this section, we exploit the special column structure of GPM in order to solve its continuous LP relaxation $\overline{\text{GPM}}$ via a column generation procedure (e.g. see Barnhart *et al.*, 1998), along with three enhancing features as discussed below. Suppose that at some iteration of the revised simplex method for solving $\overline{\text{GPM}}$, we have a basic feasible solution. Let $\{\xi \equiv (\xi_v, v \in V), \xi_0\}$ denote the corresponding complementary dual solution, where

ξ and ξ_0 are the dual variables associated with Constraints (3.1) and (3.2), respectively. We can then find a candidate entering nonbasic variable x_p that has the smallest (most negative) reduced cost by solving the following auxiliary subproblem, where π_v equals one if vertex $v \in V$ is selected for inclusion within V_p and is zero otherwise:

$$\text{SP}: \textbf{Minimize} \sum_{\substack{v_i, v_j \in V \\ i<j}} w(v_i, v_j)\pi_{v_i}\pi_{v_j} - \xi^T \pi - \xi_0,$$

$$\textbf{subject to:} \sum_{v \in V} \pi_v = \alpha, \quad \text{and} \quad \pi_v \in \{0, 1\}, \ \forall v \in V.$$

Note that the resulting vector $\pi\pi \equiv (\pi_v, v \in V)$ corresponds to a valid partition, say $p \in P$, where $\lambda_{v,p} = \pi_v, \forall v \in V$, and where the first term of the objective function represents w_p for the generated entering column. To solve Problem SP, each product relationship $\pi_{v_i}\pi_{v_j}$ that appears in the objective function can be linearized by substituting a continuous variable γ_{v_i,v_j} instead, while incorporating the following constraints, noting that π_{v_i} and π_{v_j} are required to be binary-valued and that the w-parameters are positive (see Sherali and Warren, 1998):

$$\gamma_{v_i,v_j} \geqslant \pi_{v_i} + \pi_{v_j} - 1 \quad \text{and} \quad \gamma_{v_i,v_j} \geqslant 0, \quad \forall\, v_i, v_j \in V, i < j. \tag{4.1}$$

Hence, letting $\tau^*(M)$ denote the optimal objective function value of any model M, if $M\tau^*(\text{SP}) \geqslant 0$, then no nonbasic variable is a candidate to enter the basis, and an optimal solution to Problem $\overline{\text{GPM}}$ is at hand. Otherwise, if $\tau^*(\text{AP}_{lb}) < 0$, we will have obtained a candidate entering variable x_p for $\overline{\text{GPM}}$ from the optimal solution obtained for SP as noted above, and we then introduce this column into the basis and reiterate.

4.1. *Enhancing Features*

Next, we discuss three enhancing features that can improve the solvability of Problem $\overline{\text{GPM}}$ by mitigating the tailing-off effect that is often induced by the classical column generation approach.

A) Duality based lower bounding termination criterion

The following proposition, whose proof readily follows from Proposition 1 in Ghoniem and Sherali (2009) (we include a specialized proof below for the sake of completeness), portends an optimality gap via the solution of Problem SP that will enable us to conveniently terminate the solution of Problem $\overline{\text{GPM}}$ within some percentage of optimality.

Proposition 2. *At any iteration of the column generation process to solve Problem* $\overline{\text{GPM}}$*, the solution to Problem SP provides a dual feasible solution to* $\overline{\text{GPM}}$ *with a duality gap of* $-n\tau^*(\text{SP}) \geqslant 0$.

Proof. Let $(\overline{\xi}, \overline{\xi_0})$ be the complementary dual solution to the restricted version of $\overline{\text{GPM}}$ at any iteration, where this restricted problem provides an upper bound of

$$\sum_{v \in V} \overline{\xi_v} + n\overline{\xi_0} \tag{4.2}$$

on the value $\tau^*(\overline{\text{GPM}})$. Moreover, from the corresponding problem SP, we get

$$\tau^*(\text{SP}) = \min_{p \in P} \left\{ w_p - \sum_{v \in V} \lambda_{v,p} \overline{\xi}_v - \overline{\xi}_0 \right\},$$

which implies that

$$\sum_{v \in V} \lambda_{v,p} \overline{\xi}_v + [\overline{\xi}_0 + \tau^*(\text{SP})] \leqslant w_p, \quad \forall\, p \in P,$$

or that $(\overline{\xi}, \overline{\xi}_0 + \tau^*(\text{SP}))$ is dual feasible to $\overline{\text{GPM}}$, thus establishing a lower bound on the value $\tau^*(\overline{\text{GPM}})$, with a dual objective function value of

$$\sum_{v \in V} \overline{\xi}_v + n\big[\overline{\xi}_0 + \tau^*(\text{SP})\big]. \tag{4.3}$$

From (4.2) and (4.3), we therefore infer that this dual feasible solution yields a duality gap of $-n\tau^*(\text{SP}) \geqslant 0$. □

B) Generation of complementary columns

Instead of generating a single negative reduced-cost column as done above in the classical column generation (CG), the complementary column generation (CCG) of Ghoniem and Sherali (2009) advocates the generation of multiple columns at each iteration to form a feasible *n-partition* (as possible, unless an infeasible subproblem is encountered) as described next. Let $\overline{\pi} = \{\overline{\pi}_v, v \in V\}$ be a solution to Problem SP, based on which, let $\Omega = \{v : \overline{\pi}_v = 1\}$. Let V^Ω be initialized as a set that contains the partition that includes the vertices in Ω, and let X^Ω be initialized as a set that contains the variable in $\overline{\text{GPM2}}$ corresponding to the partition in V^Ω. We then resolve Problem SP with the additional requirements that $\pi_v = 0, \forall v \in \Omega$. Let $\overline{\pi}^{new} = \{\overline{\pi_v}^{new}, v \in V\}$ denote the resulting solution. Next, the set of prohibited indices Ω is augmented by setting $\Omega \leftarrow \Omega \cup \{v : \overline{\pi_v}^{new} = 1\}$ and the sets V^Ω and X^Ω are updated accordingly. The foregoing step is repeated until $\Omega = |V|$ or an infeasible subproblem is encountered. The variables in X^Ω along with their respective partitions from V^Ω will serve to augment a restricted version of Model $\overline{\text{GPM}}$ that will be used in the next subsection within a column generation framework to solve $\overline{\text{GPM}}$. Note that when $\Omega = |V|$, the set V^Ω consists of a batch of columns that collectively constitute a feasible solution to Model GPM. Moreover, even if an infeasible subproblem is encountered in the foregoing process, the set of partitions generated thus far can be used to fruitfully augment the current restricted version of Model $\overline{\text{GPM}}$.

C) Determining a starting basis for Model $\overline{\text{GPM}}$

Note that Model $\overline{\text{GPM}}$ is highly degenerate because it has $|V|+1$ rows, but any basic feasible solution corresponding to a feasible binary solution involves only $\frac{|V|}{\alpha} = n$ non-zero binary x-variables. This will likely exacerbate the initial oscillations of dual solutions within the column generation procedure, which typically slows the convergence of the algorithm. Various dual stabilization approaches have been discussed in the literature to mitigate this phenomenon, see, for example, Bazaraa *et al.* (2010). Instead of using such dual stabilization techniques, we simply try to diminish the occurrence of oscillations by the generation of additional columns (as discussed next) in order to restrict the dual solution space, which thereby essentially contributes toward the dual stabilization process.

With this motivation, we determine $3n + |V|$ partitions of V as follows, which can then be used along with suitable artificial variables (at zero values) to determine a starting basis for $\overline{\text{GPM}}$:

a) Set $V^1 = \{v_1, v_2, \ldots, v_\alpha\}, V^2 = \{v_{\alpha+1}, v_{\alpha+2}, \ldots, v_{2\alpha}\}, \ldots, V^n = \{v_{(n-1)\alpha+1}, v_{(n-1)\alpha+2}, \ldots, v_{n\alpha}\}$, noting that $\{V^1, V^2, \ldots, V^n\}$ constitutes a valid *n-partition*.

b) For $v = 1, \ldots, |V|$, we construct $|V|$ partitions, denoted by V^{n+v}, as follows. Consider the subgraph of G that contains only the least weight $(\alpha - 1)$ edges incident to v. Let V^{n+v} be the set of vertices that contains v as well as the other $(\alpha - 1)$ vertices adjacent to v via the selected $(\alpha - 1)$ edges. Note that the vertex partitions $V^{n+1}, \ldots, V^{n+|V|}$ are not necessarily mutually exclusive.

c) Pick $v \in \{1, \ldots, |V|\}$ and initialize $V^{n+|V|+1} = \{v\}$. Find $v_1 \in V/V^{n+|V|+1}$ such that $w(v, v_1) = \min_{\overline{v} \in V/V^{n+|V|+v}} \{w(\overline{v}, v)\}$. Let $V^{n+|V|+1} = \{v, v_1\}$. Find $v_2 \in V/V^{n+|V|+1}$ such that $w(v, v_2) + w(v_1, v_2) = \min_{\overline{v} \in V/V^{n+|V|+v_1}} (w(v, \overline{v}) + w(v_1, \overline{v}))$ and proceed in this fashion until $V^{n+|V|+1}$ contains α vertices. Pick $v^1 \in V/V^{n+|V|+1}$ and let $V^{n+|V|+2}$ be the set of vertices from $V/V^{n+|V|+1}$ that contains v^1 along with $(\alpha - 1)$ vertices chosen as discussed in the process of constructing $V^{n+|V|+1}$. Continue in this manner until n vertex partitions are constructed, which collectively constitute an *n-partition*.

d) Determine the least cost vertex partition by solving Problem SP with a modified objective function given by Minimize $\sum_{\substack{v_i, v_j \in V \\ i<j}} w(v_i, v_j)\pi_{v_i}\pi_{v_j}$. The resulting solution determines a valid partition; denote this $V^{2n+|V|+1}$. Next, we resolve Problem SP with the aforementioned modified objective function while updating the set of vertices to exclude all the vertices in $V^{2n+|V|+1}$. We proceed in this manner until we construct n vertex partitions that collectively constitute a valid *n-partition*.

Hereafter, we will refer to the foregoing three enhancing complementary column generation features discussed in this section as CCG features.

4.2. *Enhanced Column Generation Algorithm to Solve* $\overline{\text{GPM}}$

An enhanced column generation algorithm that incorporates the above three features, denoted by ECGA, is presented next to solve $\overline{\text{GPM}}$. The best known lower and upper bounds

derived in this process for solving $\overline{\text{GPM}}$ as described below are denoted by LB^* and UB^*, respectively.

Algorithm ECGA

Initialization Step

Let X_0 be the set of x-variables associated with the vertex partitions V^i, $i = 1, \ldots, 3n + |V|$, as determined above, along with suitable artificial variables incorporated within the constraints of Model GPM (to determine a starting basis). Consider a restricted version of GPM, denoted by GPM_0, which contains the variables in X_0. Solve $\overline{\text{GPM}_0}$ directly using CPLEX, to determine an initial basic feasible solution to Model $\overline{\text{GPM}}$. If the set of basic variables contains any artificial variables at optimality, then iteratively solve Problem SP to generate columns for $\overline{\text{GPM}}$ until the residual artificial variables are eliminated. Set $i = 1$ and let X_1 be the set of (non-artificial) variables in the current column generation master program. Select a suitable optimality tolerance ε_1, set $LB^* = -\infty$, $UB^* = \infty$, $i = 1$ and proceed to the Main Step.

Main Step

Construct a restricted version of $\overline{\text{GPM}}$, denoted by $\overline{\text{GPM}_i}$, which only involves the variables in X_i and solve $\overline{\text{GPM}_i}$. Let $\{\overline{\xi}, \overline{\xi_0}\}$ denote the corresponding complementary dual solution for $\overline{\text{GPM}_i}$. Set $UB^* = \tau^*(\overline{\text{GPM}_i}) = \{\sum_{v \in V} \overline{\xi_v} + n\overline{\xi_0}\}$. Solve the subproblem SP, and let π^* be the optimal solution obtained. Set $LB^* = \max\{LB^*, \sum_{v \in V} \overline{\xi_v} + n[\overline{\xi_0} + \tau^*(\text{SP})]\}$.

a) If $\tau^*(\text{SP}) \geqslant 0$, stop; the optimal solution obtained for $\overline{\text{GPM}_i}$ $\overline{\text{GPM}_i}$ is also optimal for $\overline{\text{GPM}}$.
b) Trigger the CCG feature to determine X^Ω. Set $X_{i+1} = X_i \cup X^\Omega$ and let $i \leftarrow i + 1$. If $(100\frac{UB^* - LB^*}{UB^*}) \leqslant \varepsilon_1$, stop; we have an optimal solution for $\overline{\text{GPM}}$ within an optimality tolerance of ε_1 %. Otherwise, repeat the Main Step.

4.3. *Analysis of Algorithm ECGH*

a) Algorithm ECGA establishes an upper bound UB^* on $\overline{\text{GPM}}$ that decreases monotonically at each iteration of ECGA. Also, at each iteration of ECGA, the lower bound LB^* for $\overline{\text{GPM}}$ is updated based on the solution to Problem SP. At the end of Algorithm ECGA, the best provable lower bound on Model $\overline{\text{GPM}}$ is given by

$$\begin{aligned} &LB^*(\overline{\text{GPM}}) = \tau^*(\overline{\text{GPM}}) \quad \text{if } \tau^*(\text{SP}) \geqslant 0, \quad \text{and} \\ &LB^*(\overline{\text{GPM}}) = LB^*, \qquad \text{otherwise.} \end{aligned}$$

b) When the CCG Features is triggered at a given iteration of Algorithm ECGA, the set X^Ω consists of a batch of columns that collectively lend themselves toward composing a feasible solution to Model $\overline{\text{GPM}}$. The process of generating complementary columns judiciously includes multiple sets of feasible solutions whose composition is likely to enhance the possibility of encompassing optimal or near-optimal solutions when

solving the last restricted problem of GPM as a binary restricted problem as used in the procedure described in Section 5 below. Moreover, the additional columns generated provide a further restriction on the dual search space, which induces dual stabilization.

c) The duality-based lower bound established in Proposition 2 offers a helpful ε_1-based termination criterion that serves to circumvent the notorious tailing-off trend often associated with column generation procedures. Hence, both of the above features, along with the generation of $3n + |V|$ columns to initialize Algorithm ECGA, are instrumental in designing an effective heuristic approach for solving GPM with a provable optimality tolerance at termination as described in the next section.

d) Note that at any iteration of ECGA columns that are already (explicitly) present in the restricted master program ($\overline{\text{GPM}_i}$) price out with nonnegative reduced costs, and therefore these columns are automatically excluded from the solution to Problem SP (except at termination when $\tau^*(\text{SP}) \geqslant 0$). However, the inclusion of constraints in Problem SP that explicitly exclude all such columns provides valid cuts that might serve to tighten the continuous relaxation of this problem and, hence, enhance its solvability. For this purpose, letting $\overline{V} \subseteq V$ be the set of vertices that characterize the partition corresponding to any column that is currently in GPM_i, we add the following constraint to Problem SP for each such column:

$$\sum_{v \in \overline{V}} \pi_v \leqslant (\alpha - 1). \tag{4.4}$$

e) Although, the formulation of Model GPM incorporates all potential partitions as represented by the x_p variables, the initial step of the proposed column generation heuristic (ECGH) contains only a small subset of the x_p variables, then more valid partitions (x_p variables) are added iteratively until a heuristic solution is attained. In fact, this is the main advantage of adopting a column generation framework, where initially only a subset of the columns are present in the solution, and more columns are added subsequently until a heuristic solution is obtained.

5. A Heuristic Procedure for Solving GPM

In this section, we present a heuristic approach to solve Model GPM, denoted by ECGH, which is a sequential variable-fixing procedure that constructs a feasible *n-partition* in a sequential fashion in order to solve Problem GPP. Essentially, this procedure generates an *n-partition* by augmenting fixed partitions from solutions to $\overline{\text{GPM}}$ obtained via the ECGA method outlined in the foregoing section.

To describe this procedure, cconsider an optimal solution to $\overline{\text{GPM}}$ obtained via ECGA. Let S_b^1 be the index set of the basic variables that equal one when ECGA terminates, and let S_b^f be the set of fractional basic variables at optimality. (Note that if $S_b^f = \emptyset$, we have an *n-partition* at hand, and we stop with this solution as optimal for GPM.) Initialize the set $\Gamma = S_b^1$. Hence, $V^{p^i} \cap V^{p^j} = \emptyset, \forall\, p^i, p^j \in \Gamma$ with $i \neq j$, because otherwise, if there exists some $v \in V^{p^i} \cap V^{p^j}$, then equation (3.1) corresponding to vertex v would be violated.

Note that $|\Gamma| \leqslant n$, or else, we would have an $\overline{n}$-*partition*, where $\overline{n} > n$, which involves $\alpha\overline{n}$ vertices from V, contradicting the fact that $|V| = \alpha n$.

The following Variable-Fixing Step will be used with our proposed enhanced column generation heuristic described subsequently.

Variable-Fixing Step

Let $\overline{x}$ be the optimal solution obtained for $\overline{\text{GPM}}$ and let $\overline{x}_{\max} = \max_{p \in S_b^f}\{\overline{x}_p\}$, and determine $\Lambda = \{p \in S_b^f : \overline{x}_p = \overline{x}_{\max}\}$, $w_{\min} = \min\{w_p : p \in \Lambda\}$, and $\overline{\Lambda} = \{p \in \Lambda : w_p = w_{\min}\}$. Pick $\hat{p} \in \overline{\Lambda}$ and update $\Gamma \leftarrow \Gamma \cup \{\hat{p}\}$. Let $\Pi = \{p \in P : V^p \cap V^{p^1} = \emptyset, \forall p^1 \in \Gamma\}$, and correspondingly, define $V^{\Pi} = \{v \in V : v \notin \bigcup_{p \in \Gamma} V^p\}$.

Based on the above variable-fixing step, we let GPM_{Π} to be a modified version of GPM obtained by: (a) restricting the set of partitions P to Π, and (b) replacing V by V^{Π} in (3.1). This problem is given as follows:

$$\text{GPM}_{\Pi}: \textbf{Minimize} \sum_{p \in \Pi} w_p x_p$$

subject to

$$\sum_{p \in \Pi} \lambda_{v,p} x_p = 1, \quad \forall v \in V^{\Pi}, \tag{5.1}$$

$$\sum_{p \in \Pi} x_p = n - |\Gamma|, \tag{5.2}$$

$$x_p \in \{0, 1\}, \quad \forall p \in \Pi.$$

We can now solve $\overline{\text{GPM}_{\Pi}}$ using ECGA as before, based on the remnant set of vertices, and repeat the process until an n-*partition* is obtained. The proposed heuristic (ECGH) for generating an n-*partition* is stated formally below, noting that whenever $|\Gamma| = n$, we have at hand an n-*partition* that is described by the set of valid partitions $\{p \in \Gamma\}$.

Heuristic ECGH

Initialization

Set $\Gamma = \emptyset\Gamma = \emptyset$, $\Pi = P$, and $V^{\Pi} = V$.

LP-Step

Solve $\overline{\text{GPM}_{\Pi}}$ using ECGA, and let $\overline{X}$ denote the resulting solution. Determine the index sets S_b^1 and S_b^f. If $S_b^f = \emptyset S_b^f = \emptyset$, then update $\Gamma \leftarrow \Gamma \cup S_b^1$ and stop, since $|\Gamma| = n$, then stop; otherwise, proceed to the Variable-Fixing Step.

Variable-Fixing Step (This step is described above).

Final Step

Let $\Gamma \leftarrow \Gamma \cup \{\hat{p}\}$. If $|\Gamma| = n$; otherwise, update Π and V^{Π}, and repeat the LP-Step.

Table 1
Test problems for model GPM.

Test Problem	TP_1	TP_2	TP_3	TP_4	TP_5	TP_6
$(\|V\|, n)$	(12, 4)	(18, 3)	(20, 5)	(20, 2)	(24, 2)	(30, 3)
Test Problem	TP_7	TP_8	TP_9	TP_{10}	TP_{11}	TP_{12}
$(\|V\|, n)$	(30, 2)	(36, 3)	(36, 2)	(40, 8)	(40, 5)	(40, 4)
Test Problem	TP_{13}	TP_{14}	TP_{15}	TP_{16}	TP_{17}	TP_{18}
$(\|V\|, n)$	(50, 25)	(50, 10)	(50, 5)	(60, 15)	(60, 10)	(72, 18)
Test Problem	TP_{19}	TP_{20}	TP_{21}	TP_{22}	TP_{23}	TP_{24}
$(\|V\|, n)$	(72, 12)	(84, 28)	(84, 21)	(84, 14)	(100, 25)	(100, 20)

REMARK 3.

a) At each iteration of ECGH, the set Γ is augmented by at least one valid partition in a feasible fashion, and the number of elements in Γ cannot exceed n. Consequently, the algorithm terminates finitely whenever $|\Gamma| = n$, yielding a desired *n-partition*.
b) In the "Variable-Fixing Step", note that $V^{\hat{p}} \cap V^{p} = \emptyset V^{\hat{p}} \cap V^{p}$, $\forall p \in \Gamma$, because otherwise, (3.1) would be violated for some $v \in V$. Hence, we maintain a partitioning of the vertices while augmenting the set Γ according to $\Gamma \leftarrow \Gamma \cup \{\hat{p}\}$.

6. Computational Results

In this section, we present computational results for the proposed complementary column generation approach for solving Model GPM. We use a set of 24 test problems described in Table 1 for GPM, where TP_i, $i = 1, \ldots, 24$, represents the i-th test problem. For all the test instances, the weights associated with the edges are randomly generated using a random function within the interval [1, 1000]. All computational results have been performed on a Core™ i7 Processor, CPU 4.00 GHz computer having 4 GB of RAM and using the CPLEX Commercial Package (version 12) as the optimization solver.

Below, we summarize the notation that will be used in this section.

- ε_1: Optimality gap tolerance for implementing Algorithm ECGH to solve $\overline{\text{GPM}}$.
- ε_2: Optimality gap tolerance for solving SP.
- $LB^*(\overline{\text{GPM}}) = \begin{cases} \tau^*(\overline{\text{GPM}}) & \text{if } \tau^*(\text{SP}) \geqslant 0, \\ LB^*, & \text{otherwise.} \end{cases}$
- $\tau^*(\text{ECGH})$: The best objective function value obtained for Model GPM using Heuristic ECGH.
- ρ: $100\big(\frac{\tau^*(\text{ECGH}) - LB^*(\overline{\text{GPM}})}{\tau^*(\text{ECGH})}\big)\% =$ Percentage optimality gap for the best solution obtained for Model GPM via Heuristic ECGH.
- T_{ECGA}: The total solution time for solving $\overline{\text{GPM}}$ using Algorithm ECGA.
- T_{ECGH}: The total solution time for solving GPM using Heuristic ECGH.

We begin by presenting computational results pertaining to solving Model GPM using Heuristic ECGH based on a set of 24 test problems having up to 100 vertices with different partitioning requirements. Tables 2 and 3, respectively, present our computational experience in solving Model GPM with and without the CCG Features. Note that because the

Table 2
Results for solving $\overline{\text{GPM}}$ and GPM using the CCG features.

Test problem TP_i I	$LB^*(\overline{\text{GPM}})$	T_{ECGA} (seconds)	$\tau^*(\text{GPM})$	T_{ECGH} (seconds)	ρ %	$(\varepsilon_1, \varepsilon_2)$
1	48.0	0.35	51.0	1.48	5.882353	(0, 0)
2	1379.0	3.05	1392.0	4.65	0.933908	(0, 0)
3	2879.0	4.01	2883.0	4.84	0.138744	(0, 0)
4	3144.0	5.75	3145.0	6.59	0.031797	(0, 0)
5	12459.0	14.36	12459.0	16.67	0.000000	(0, 0)
6	51806.0	35.34	52526.0	50.83	1.370750	(0, 0)
7	91242.0	1042.42	92273.0	1123.08	1.117337	(0, 0)
8	66484.0	16.31	66484.0	17.84	0.000000	(0, 0)
9	85758.0	23.14	85760.0	25.28	0.002332	(0, 0)
10	92042.0	23.27	93059.0	24.35	1.092855	(0.00001, 0)
11	10825.79	1795.3	11473.34	1825.81	5.643954	(0.00001, 0)
12	73011.0	25.84	73063.0	27.67	0.071171	(0.001, 0)
13	3614.0	24.09	3820.0	26.05	5.392670	(0.005, 0)
14	44231.0	23	44555.0	24.12	0.727191	(0.005, 0)
15	42106.0	24.08	43611.63	25.48	3.452359	(0.005, 0)
16	61479.0	25.14	63020.0	28.8	2.445255	(0.005, 0)
17	23002.0	26.06	23737.0	28.64	3.096432	(0.005, 0)
18	3418.0	26.94	3674.0	29.06	6.967882	(0.005, 0)
19	5239.0	28.45	5417.0	30.59	3.285952	(0.05, 0)
20	16812.09	1904.34	17521.0	1987.16	4.045895	(0.2, 0)
21	65132.0	13060	67245.0	13239.3	3.142241	(0.2, 0)
22	51076.0	6462.82	58177.2	9974.89	12.206171	(0.2, 0.1)
23	3671.0	16462.8	3894.0	16591.9	5.726759	(0.2, 0.1)
24	48552.0	17328.5	52278.0	18148.5	7.127281	(0.2, 0.1)
AVG	**35808.75**	**2432.72**	**36729.92**	**2635.98**	**3.08**	

weights associated with Problem GPP are randomly generated, repeated implementations of Heuristic ECGH might produce different objective function values, optimality gaps and run-times for a given test problem. Also, observe that in Proposition 2, we have assumed that Problem SP is solved using $\varepsilon_2 = 0$, in which case we have the provable lower bound given either by $\tau^*(\overline{\text{GPM}})$ if $\tau^*(\text{SP}) \geqslant 0$ or otherwise by LB^*. However, when using a tolerance $\varepsilon_2 > 0$ while solving Problem SP, we can modify Proposition 2 to assert that the established duality gap is given by $-n\tau^*(\text{SP}) + n\varepsilon_2 \geqslant 0$, with $LB^* \leftarrow LB^* - n\varepsilon_2$. The use of this lower bounding scheme is instrumental in curtailing the tailing-off effect associated with column generation. Hence, for each test problem, we first set $\varepsilon_1 = \varepsilon_2 = 0$ and try to solve Model GPM within some specified time. In case a solution is not obtainable, we then set $\varepsilon_2 = 0$ and set ε_1 to some sufficiently small value and gradually increment it until we reach a solution within the specified time. For test instances that remain unsolved, we set ε_2 to a sufficiently small positive value and increment it until we obtain a solution within the specified time. From our preliminary computational experiments, we noticed that solving Problem SP even with the default cutting plane feature of CPLEX was cumbersome in some test instances, especially those having a relatively large number of vertices, while the time to update the solution to GPM was in most cases just a few seconds.

Table 3
Results for solving $\overline{\text{GPM}}$ and GPM without the CCG features.

Test problem TP_i I	$LB^*(\overline{\text{GPM}})$	T_{ECGA} (seconds)	τ^*(GPM)	T_{ECGH} (seconds)	ρ %
1	24.0	2	57.0	5	57.89
2	1409.0	26	1417.0	28	0.56
3	3466.0	42	3466.0	42	0
4	10837.5	1756	10837.5	1756	0
5	46625.0	805	46625.0	805	0
6	44930.0	1183	44930.0	1183	0
7	90099.0	571	90099.0	571	0
8	69459.0	10819	69459.0	10819	0
9	136174.0	5379	136174.0	5379	0
10	16922.66	284	17068.0	492	0.85
11	40307.99	2656	40802.0	4025	1.21
12	58437.42	9349	59157.0	10691	1.22
13	1837.0	15	1837.0	15	0
14	21389.0	1015	21703.44	2055	1.45
15	70301.44	476272	71545.0	481129	1.74
16	16571.33	838	17094.0	1042	3.06
17	37442.45	7014	38060.0	8222	1.62
18	3304.0	1178	3528.0	1693	6.35
19	3410.13	10640	3528.0	14012	3.34
20	11292.95	1207	12492.0	1376	9.60
21	23051.92	6769	23824.0	17679	3.24
22	53427.27	85003	54761.0	225112	2.44
23	145.59	8323	145.59	9457	0
24	271.41	34284	290.91	36578	6.70
AVG	**31714.00**	**27726.25**	**32037.52**	**34756.92**	**4.22**

In order to better understand the efficiency of the CCG Features and its contribution toward enhancing the solution quality for Model GPM, we experimented with solving this model using Heuristic ECGH both with and without the CCG Features. As shown in Tables 2 and 3, the incorporation of the CCG Features reduced the average solution time for solving Model $\overline{\text{GPM}}$ by 11-fold and for solving Model GPM by 12-fold. This substantial reduction in the average solution times is attained by virtue of mitigating the tailing-off effect at termination. The average optimality gap at termination when solving Model GPM with and without the enhancing features are given by 3.08% and 4.22%, respectively. Although the higher average optimality gap obtained for the latter is skewed because of the relatively high optimality gap of 57.89% obtained when solving problem TP_1 (without this test case, the average optimality gap is given by 1.88%), but still the significant reduction in the average solution time when using the CCG Features strengthens the robustness of the proposed column generation approach.

In the remainder of this section, we focus and analyse results obtained using the CCG Features. To provide insights into the performance of our approach for solving Model GPM, we partition our test problems into three sets, denoted by S_1, S_2 and S_3, based on the number of vertices ranging up to 36, 84, and 100, respectively. Tables 4 and 5 provide results for these partitioned subsets of problems. As expected, with an increase in

Table 4
Ssummary of run-times and optimality gaps for model GPM.

Problem set	Test problems	Max number of vertices	Least T_{ECGH}	Largest T_{ECGH}	Ave T_{ECGH}	Least ρ %	Largest ρ %	Ave ρ %
S_1	$TP1_1, \ldots, TP1_{10}$	36	1.48	1825.81	260.75	0	5.88	1.35
S_2	$TP1_{11}, \ldots, TP1_{21}$	84	24.12	13239.30	2348.93	0.07	6.96	3.48
S_3	$TP1_{22}, \ldots, TP1_{24}$	100	9974.89	18148.50	13587.56	5.72	12.22	8.35

Table 5
Tolerances for model $\overline{\text{GPM}}$ and problem SP.

Problem set	Least ε_1	Largest ε_1	Average ε_1	Least ε_2	Largest ε_2	Average ε_2
S_1	0	0	0	0	0	0
S_2	0.00001	0.02	0.0400	0	0	0
S_3	0.2	0.2	0.2	0.1	0.1	0.1

the number of vertices, both the total run-time for solving Model GPM using ECGH and the resulting optimality gap at termination increased. Test problems from the set S_1 (with $n \leqslant 36$) were solvable without using any termination tolerances for solving either $\overline{\text{GPM}}$ or Problem SP, and in this case the least, largest, and average optimality gaps obtained were given by 0%, 5.88%, and 1.35%, respectively. For test problems from S_2 (having $n \leqslant 84$), we solved Model $\overline{\text{GPM}}$ using a range of incrementally increasing gap tolerances. In this case, the least, largest, and average optimality gaps obtained were given by 0.7%, 6.96%, and 3.45%, respectively. For test problems in S_3 (having $n \leqslant 100$), using the aforementioned modified lower bounding result, the least, largest, and average optimality gaps obtained were given by 5.72%, 12.22%, and 8.35%, respectively.

7. Summary, Conclusions and Future Research

This paper examines a graph partitioning problem that is concerned with the partitioning of a complete weighted graph $G(V, E)$ into n complete subgraphs each having the same number of α vertices, with the objective of minimizing the total weight of edges included in the subgraphs. This problem has many applications in various contexts such as assignment-related group partitioning problems, micro aggregation in statistics, telecommunication, and political redistricting. To solve this problem, we formulated a mixed-integer program, denoted by GPM, which directly attempts to select a minimum-cost collection of n valid partitions from the entire set of valid partitions in order to constitute an *n-partition*. Exploiting the structure of Model GPM, we then designed a column generation heuristic (ECGH) that incorporates the following three enhancing features for solving this model: (a) a lower bounding facility based on solving the pricing subproblem, which helps to curtail the tailing off effect typically associated with column generation; (b) a complementary column generation feature that attempts to generate multiple columns at each

iteration to constitute a feasible *n-partition*, and (c) the generation of initial columns for Model GPM that serve to provide a starting basis as well as to restrict the dual solution space, thereby contributing toward dual stabilization.

Detailed computational results were presented for solving Model GPM. These results demonstrated that the CCG Features proposed for enhancing the traditional column generation framework yielded comparable quality solutions (3% optimality on average) with respect to the standard classical column generation approach while reducing the average run-time for solving Models $\overline{\text{GPM}}$ and GPM by 11-fold and 12-fold, respectively. Based on our computational results, we propose investigating further algorithmic strategies for dealing with relatively larger problems. In particular, solving Problem SP within algorithm ECGH was especially cumbersome in test instances of Problem GPP that involve a relatively large number of vertices. In fact, we were able to obtain reasonable solutions for up to 100 vertices for Model GPM, but for problems having 150 vertices, we were unable to solve even the linear programming relaxation of Problem SP after two days of run-time. Hence, we recommend exploring some alternative ways for solving Problem SP, including a polyhedral analysis coupled with more effective heuristic solution approaches.

Another extension worth exploring for solving Model GPM is as follows. Let $\overline{\text{GPM}}_{\text{LR}}$ be the current restricted version of Model $\overline{\text{GPM}}$ obtained from the final iteration of Algorithm ECGH. We can then solve $\overline{\text{GPM}}_{\text{LR}}$ to optimality as a 0-1 program directly using a commercial package such as CPLEX by utilizing some suitable specialized decomposition scheme as necessary, in order to obtain a good quality feasible solution to Model GPM. This might be particularly attractive because of the complementary column generation strategy implemented within the solution process (Ghoniem and Sherali, 2009). Moreover, this solution approach can further provide facility to consider equity issues within the vertex partitioning scheme through the addition of suitable side-constraints, which are difficult otherwise to accommodate within the column generation modelling and solution process. In the future, we also aim to explore alternative modelling approaches for Problem GPP that attempt to directly generate the required partitions and use decomposition schemes such as Lagrangian relaxation while incorporating necessary equity.

Acknowledgements. The authors gratefully acknowledge the assistance of Ms. Renju Lekshmi in implementing the developed procedures.

Funding

This work was supported by *Kuwait University* under Research Grant No. [SM03/14].

References

Barnhart, C., Johnson, E.L., Nemhauser, G.L., Savelsbergh, M.W., Vance, P.H. (1998). Branch-and-price column generation for solving huge integer programs. *Operation Research*, 46(3), 316–329.

Bazaraa, M.S., Jarvis, J.J., Sherali, H.D. (2010). *Linear Programming and Network Flows*. 4th ed. John Wiley and Sons, Hoboken, New Jersey.

Carlson, R.C., Nemhauser, G.L. (1966). Scheduling to minimize interaction costs. *Operations Research*, 14, 52–58.

Chlebikova, J. (1996). Approximating the maximally balanced connected partition problem in graphs. *Information Processing Letters*, 60, 225–230.

Conforti, M., Rao, M.R. (1990a). The equipartition polytope. I: formulations, dimension and basic facets. *Mathematical Programming*, 49, 49–70.

Conforti, M., Rao, M.R. (1990b). The equipartition polytope. II: valid inequalities and facets. *Mathematical Programming*, 49, 71–90.

Domingo-Ferrer, J., Mateo-Sanz, J.M. (2002). Practical data oriented microaggregation for statistical disclosure control. *IEEE Transactions on Knowledge and Data Engineering*, 14(1), 189–201.

Donath, W.E., Hoffman, A.J. (1973). Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development*, 17(5), 420–425.

Fan, N., Pardalos, P.M. (2010). Linear and quadratic programming approaches for the general graph partitioning problem. *Journal of Global Optimization*, 48, 57–71.

Fan, N., Pardalos, P.M. (2012). Multi-way clustering and biclustering by the ratio cut and normalized cut in graphs. *Journal of Combinatorial Optimization*, 23, 224–251.

Fan, N., Pardalos, P.M., Chinchuluun, A., Pistikopoulos, E.N. (2009). Graph partitioning approaches for analyzing biological networks. In: *BIOMAT 2009 – International Symposium on Mathematical and Computational Biology*.

Garey, M.R., Johnson, D.S., Stockmeyer, L. (1976). Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1, 237–267.

Ghoniem, A., Sherali, H.D. (2009). Complementary column generation and bounding approaches for set partitioning formulations. *Optimization Letters*, 3(1), 123–136.

Goldschmidt, O., Hochbaum, D. (1988). Polynomial algorithm for the k-cut problem. In: *29th Annual IEEE Symposium on Foundations of Computer Science*. IEEE Computer Society.

Grotschel, M., Wakabayashi, Y. (1989). A cutting plane algorithm for a clustering problem. *Mathematical Programming*, 45, 59–96.

Grotschel, M., Wakabayashi, Y. (1990). Facets of the clique partitioning polytope. *Mathematical Programming*, 47, 367–387.

Hager, W., Krylyuk, Y. (1999). Graph partitioning and continuous quadratic programming. *SIAM J. Discret. Math*, 12(4), 500–523.

Hager, W., Krylyuk, Y. (2002). Multiset graph partitioning. *Math. Meth. Oper Res*, 55, 1–10.

Ji, X. (2004). *Graph partitioning problem with minimum weight constraints*. Technical Report, Rensselaer Polytechnic Institute, Graduate Faculty, New York, NY.

Karisch, S.E., Rendl, F. (1998). Semidefinite programming and graph equipartition. In: Pardalos, P.M., Wolkowicz, H. (Eds.), *Topics in Semidefinite and Interior-Point Methods*. American Mathematical Society, USA, pp. 77–95.

Karypis, G., Aggarwal, R., Kumar, V., Shekhar, S. (1999). Multilevel hypergraph partitioning: applications in VLSI domain. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 7(1), 69–79.

Laguna, M. (1994). Clustering for the design of SONET rings in interoffice telecommunications. *Management Science*, 40(11), 1533–1541.

Lisser, A., Rendl, F. (2003). Graph partitioning using linear and semidefinite programming. *Mathematical Programming Series B*, 95, 91–101.

Mehrotra, A., Trick, M.A. (1998). Cliques and clustering: a combinatorial approach. *Operation Research Letters*, 22, 1–12.

Mehrotra, A., Johnson, E.L., Nemhauser, G.L. (1998). An optimization based heuristic for political districting. *Management Science*, 44(8), 1100–1114.

Mitchell, J.E. (2001). *Branch-and-cut for the k-way equipartition problem*. Technical Report, Rensselaer Polytechnic Institute Troy, NY.

Mitchell, J.E. (2003). Realignment in the National Football League: did they do it right? *Naval Research Logistics*, 50(7), 683–701.

Park, K., Lee, K., Park, S., Lee, H. (2000). Telecomunication node clustering with node compatibility and network survivability requirements. *Management Science*, 46(3), 363–374.

Salgado, L.R., Wakabayashi, Y. (2004). Approximation results on balanced connected partitions of graphs. *Electronic Notes in Discrete Mathematics*, 18, 207–212.

Salido, M.A., Abril, M., Barber, F., Ingolotti, L., Tormos, P., Lova, A. (2007). Domain-dependent distributed models for railway scheduling. *Knowledge Based Systems*, 20, 186–194.

Schaeffer, S.E. (2007). Survey: graph clustering. *Computer Science Review*, 1, 27–64.

Sherali, H., Warren, A. (1998). Reformulation-linearization techniques for discrete optimization problems. In: Du, D.Z., Pardalos, P.M. (Eds.), *Handbook of Combinatorial Optimization*. Springer, Boston, MA, pp. 479–532.

Tan, Y.P., Lu, H. (2003). Video scene clustering by graph partitioning. *Electronics Letters*, 39(11), 841–842.

Wolkowicz, H., Zhao, Q. (1996). Semidefinite programming relaxations for the graph partitioning problem. *Discrete Applied Mathematics*, 96–97, 461–479.

Xiao, B., Cao, J., Shao, Z., Zhuge, Q., Shao, E.H.M. (2007). Analysis and algorithms design for the partition of large-scale adaptive mobile wireless networks. *Computer Communications*, 30, 1899–1912.

Xiaoyun, J., Mitchell, J. (2005). Finding optimal realighment in sports leagues using a branch-and-cut-and-price approach. *International Journal of Operational Research*, 1(1–2), 101–122.

Zha, H., He, X., Ding, C., Simon, H., Gu, M. (2001). Bipartite graph partitioning and data clustering. In: *CIKM'01: Proceedings of the Tenth International Conference on Information and Knowledge Management*. ACM Press, New York, NY, pp. 25–32.

S.M. Al-Yakoob is an associate professor at the Department of Mathematics, Kuwait University. His research interests include mathematical programming and optimization with applications to real world problems such as location, transportation, scheduling, and timetabling problems.

H.D. Sherali is a University Distinguished Professor Emeritus at the Industrial and Systems Engineering Department, Virginia Polytechnic Institute and State University. His areas of research interest are in mathematical optimization modelling, analysis, and design of algorithms for specially structured linear, nonlinear, and continuous and discrete nonconvex programs, with applications to transportation, location, engineering and network design, production, economics, and energy systems. He has published over 351 refereed articles in various operations research journals and has (co-)authored nine books, with a total Google Scholar citation count of over 35,700 and an H-index of 75. He is an elected member of the National Academy of Engineering, a fellow of both INFORMS and IIE, and a member of the Virginia Academy of Science Engineering and Medicine.