Comput Optim Appl (2018) 69:189–223
https://doi.org/10.1007/s10589-017-9945-2

A multilevel bilinear programming algorithm for the vertex separator problem

William W. Hager[1] · **James T. Hungerford**[2] · **Ilya Safro**[3]

Received: 1 September 2017 / Published online: 4 October 2017

Abstract The Vertex Separator Problem for a graph is to find the smallest collection of vertices whose removal breaks the graph into two disconnected subsets that satisfy specified size constraints. The Vertex Separator Problem was formulated in the paper 10.1016/j.ejor.2014.05.042 as a continuous (non-concave/non-convex) bilinear quadratic program. In this paper, we develop a more general continuous bilinear program which incorporates vertex weights, and which applies to the coarse graphs that are generated in a multilevel compression of the original Vertex Separator Problem. We develop a method for improving upon a given vertex separator by applying a Mountain Climbing Algorithm to the bilinear program using an incidence vector for the separator as a starting guess. Sufficient conditions are developed under which the algorithm

September 1st, 2017. The research was supported by the Office of Naval Research under Grants N00014-11-1-0068 and N00014-15-1-2048 and by the National Science Foundation under Grants 1522629 and 1522751. Part of the research was performed while the second author was a Givens Associate at Argonne National Laboratory.

✉ James T. Hungerford
jamesthungerford@gmail.com

William W. Hager
hager@ufl.edu
http://people.clas.ufl.edu/hager/

Ilya Safro
isafro@clemson.edu
http://www.cs.clemson.edu/~isafro

1 Department of Mathematics, University of Florida, PO Box 118105, Gainesville, FL 32611-8105, USA

2 RaceTrac Store Support Center, 200 Galleria Pkwy SE, Atlanta, GA 30339, USA

3 School of Computing, Clemson University, 228 McAdams Hall, Clemson, SC 29634, USA

can improve upon the starting guess after at most two iterations. The refinement algorithm is augmented with a perturbation technique to enable escapes from local optima and is embedded in a multilevel framework for solving large scale instances of the problem. The multilevel algorithm is shown through computational experiments to perform particularly well on communication and collaboration networks.

Keywords Vertex separator · Continuous formulation · Graph partitioning · Multilevel · Weighted edge contractions · Multilevel algorithm

Mathematics Subject Classification 90C35 · 90C27 · 90C20 · 90C06

1 Introduction

Let $G = (\mathcal{V}, \mathcal{E})$ be a graph on vertex set $\mathcal{V} = \{1, 2, \ldots, n\}$ and edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. We assume G is simple and undirected; that is for any vertices i and j we have $(i, i) \notin \mathcal{E}$ and $(i, j) \in \mathcal{E}$ if and only if $(j, i) \in \mathcal{E}$ (note that this implies that $|\mathcal{E}|$, the number of elements in $\mathcal{E}$, is twice the total number of edges in G). For each $i \in \mathcal{V}$, let $c_i \in \mathbb{R}$ denote the cost and $w_i > 0$ denote the weight of vertex i. If $\mathcal{Z} \subseteq \mathcal{V}$, then

$$\mathcal{C}(\mathcal{Z}) = \sum_{i \in \mathcal{Z}} c_i \quad \text{and} \quad \mathcal{W}(\mathcal{Z}) = \sum_{i \in \mathcal{Z}} w_i$$

denote the total cost and weight of the vertices in $\mathcal{Z}$, respectively.

If the vertices $\mathcal{V}$ are partitioned into three disjoint sets $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{S}$, then $\mathcal{S}$ *separates* $\mathcal{A}$ and $\mathcal{B}$ if there is no edge $(i, j) \in \mathcal{E}$ with $i \in \mathcal{A}$ and $j \in \mathcal{B}$. The Vertex Separator Problem (VSP) is to minimize the cost of $\mathcal{S}$ while requiring that $\mathcal{A}$ and $\mathcal{B}$ have approximately the same weight. We formally state the VSP as follows:

$$\begin{aligned} &\min_{\mathcal{A},\mathcal{S},\mathcal{B} \subseteq \mathcal{V}} \quad \mathcal{C}(\mathcal{S}) \\ &\text{subject to} \quad \mathcal{S} = \mathcal{V} \setminus (\mathcal{A} \cup \mathcal{B}), \quad \mathcal{A} \cap \mathcal{B} = \emptyset, \quad (\mathcal{A} \times \mathcal{B}) \cap \mathcal{E} = \emptyset, \\ &\ell_a \le \mathcal{W}(\mathcal{A}) \le u_a, \quad \text{and} \quad \ell_b \le \mathcal{W}(\mathcal{B}) \le u_b, \end{aligned} \tag{1}$$

where ℓ_a, ℓ_b, u_a, and u_b are given nonnegative real numbers less than or equal to $\mathcal{W}(\mathcal{V})$. The constraints $\mathcal{S} = \mathcal{V} \setminus (\mathcal{A} \cup \mathcal{B})$ and $\mathcal{A} \cap \mathcal{B} = \emptyset$ ensure that $\mathcal{V}$ is partitioned into disjoint sets $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{S}$, while the constraint $(\mathcal{A} \times \mathcal{B}) \cap \mathcal{E} = \emptyset$ ensures that there are no edges between the sets $\mathcal{A}$ and $\mathcal{B}$. Throughout the paper, we assume (1) is feasible. In particular, if $\ell_a, \ell_b \ge 1$, then there exist at least two distinct vertices i and j such that $(i, j) \notin \mathcal{E}$; that is, G is not a complete graph.

Vertex separators have applications in VLSI design [26,30,39], finite element methods [32], parallel processing [12], sparse matrix factorizations ([9, Sect. 7.6], [16, Chapter 8], and [34]), hypergraph partitioning [25], and network security [7,27,31]. The VSP is NP-hard [5,15]. However, due to its practical significance, many heuristics have been developed for obtaining approximate solutions, including node-swapping heuristics [29], spectral methods [34], semidefinite programming methods [13], and recently a breakout local search algorithm [3].

Early methods [17,34], for computing vertex separators were based on computing edge separators (bipartitions of $\mathcal{V}$ with low cost edge-cuts). In these algorithms, vertex separators are obtained from edge separators by selecting vertices incident to the edges in the cut. More recently, [1] gave a method for computing vertex separators in a graph by finding low cost net-cuts in an associated hypergraph. Some of the most widely used heuristics for computing edge separators are the node swapping heuristics of Fiduccia–Mattheyses [14] and Kernighan–Lin [26], in which vertices are exchanged between sets until the current partition is considered to be locally optimal.

It has been demonstrated repeatedly that for problems on large-scale graphs, such as finding minimum k-partitionings [6,20,24] or minimum linear arrangements [36,38], optimization algorithms can be much more effective when carried out in a multilevel framework. In a multilevel framework, a hierarchy of increasingly smaller graphs is generated which approximate the original graph, but with fewer degrees of freedom. The problem is solved for the coarsest graph in the hierarchy, and the solution is gradually uncoarsened and refined to obtain a solution for the original graph. During the uncoarsening phase, optimization algorithms are commonly employed locally to make fast improvements to the solution at each level in the algorithm. Although multilevel algorithms are inexact for most NP-hard problems on graphs, they typically produce very high quality solutions and are very fast (often linear in the number of vertices plus the number of edges with no hidden coefficients). Many multilevel edge separator algorithms have been developed and incorporated into graph partitioning packages (see survey in [6]). In [2], a Fiduccia–Mattheyses type heuristic is used to find vertex separators directly. Variants of this algorithm have been incorporated into the multilevel graph partitioners METIS [23,24] and BEND [21].

In [19], the authors make a departure from traditional discrete-based heuristics for solving the VSP, and present the first formulation of the problem as a continuous optimization problem. In particular, when the vertex weights are identically one, conditions are given under which the VSP is equivalent to solving a continuous bilinear quadratic program.

The preliminary numerical results of [19] indicate that the bilinear programming formulation can serve as an effective tool for making local improvements to a solution in a multilevel context. The current work makes the following contributions:

1. The bilinear programming model of [19] is extended to the case where vertex weights are possibly greater than one. This generalization is important since each vertex in a multilevel compression corresponds to a collection of vertices in the original graph. The bilinear formulation of the compressed graph is not exactly equivalent to the VSP for the compressed graph, but it very closely approximates the VSP as we show.
2. We develop an approach to improve upon a given vertex separator by applying a Mountain Climbing Algorithm to the bilinear program using the incidence vector for the separator as a starting guess. Sufficient conditions are developed under which the algorithm can improve upon the separator after at most two iterations.

3. We investigate a technique for escaping local optima encountered by the Mountain Climbing Algorithm based on relaxing the constraint that there are no edges between the sets in the partition. Since this constraint is enforced by a penalty in the objective, we determine the smallest possible relaxation of the penalty for which a gradient descent step moves the iterate to a new location where the separator could be smaller.
4. A multilevel algorithm is developed which incorporates the weighted bilinear program in the refinement phase along with the perturbation technique. Computational results are given to compare the quality of the solutions obtained with the bilinear programming approach to a multilevel vertex separator routine in the METIS package. The algorithm is shown to be especially effective on communication and collaboration networks.

The outline of the paper is as follows. Section 3 reviews the bilinear programming formulation of the VSP in [19] and develops the weighted formulation which is suitable for the coarser levels in the algorithm. In Sect. 4 we develop the Mountain Climbing Algorithm and examine some sufficient conditions under which a separator can be improved by the algorithm. The conditions are derived in the appendix. In addition, a perturbation technique is developed for escaping local optima. Section 5 summarizes the multilevel framework, while Sect. 6 gives numerical results comparing our algorithm to METIS. Conclusions are drawn in Sect. 7.

2 Notation

Vectors or matrices whose entries are all 0 or all 1 are denoted by $\mathbf{0}$ or $\mathbf{1}$ respectively, where the dimension will be clear from the context. The set of integers is $\mathbb{Z}$, while $\mathbb{Z}^+$ is the set of strictly positive integers. If $\mathbf{x} \in \mathbb{R}^n$ and $f : \mathbb{R}^n \to \mathbb{R}$, then $\nabla f(\mathbf{x})$ denotes the gradient of f at $\mathbf{x}$, a row vector, and $\nabla^2 f(\mathbf{x})$ is the Hessian. If $f : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, then $\nabla_{\mathbf{x}} f(\mathbf{x}, \mathbf{y})$ is the row vector corresponding to the first n entries of $\nabla f(\mathbf{x}, \mathbf{y})$, while $\nabla_{\mathbf{y}} f(\mathbf{x}, \mathbf{y})$ is the row vector corresponding to the last n entries. If $\mathbf{A}$ is a matrix, then $\mathbf{A}_i$ denotes the i-th row of $\mathbf{A}$. If $\mathbf{x} \in \mathbb{R}^n$, then $\mathbf{x} \geq \mathbf{0}$ means $x_i \geq 0$ for all i, and $\mathbf{x}^\mathsf{T}$ denotes the transpose, a row vector. Let $\mathbf{I} \in \mathbb{R}^{n\times n}$ denote the $n \times n$ identity matrix, let $\mathbf{e}_i$ denote the i-th column of $\mathbf{I}$, and let $|\mathcal{A}|$ denote the number of elements in the set $\mathcal{A}$. If $\mathcal{Z} \subseteq \mathcal{V}$, then

$$\mathcal{N}(\mathcal{Z}) = \{j \in \mathcal{V}\backslash\mathcal{Z} \ : \ \exists\, i \in \mathcal{Z} \text{ s.t. } (i, j) \in \mathcal{E}\}$$

is the set of neighbors of $\mathcal{Z}$ and $\overline{\mathcal{N}}(\mathcal{Z}) = \mathcal{N}(\mathcal{Z}) \cup \mathcal{Z}$. Let $\mathbb{B} = \{0, 1\}$ denote the binary numbers. If $\mathbf{x} \in \mathbb{B}^n$, then the support of $\mathbf{x}$ is defined by

$$\text{supp}(\mathbf{x}) = \{i \in \mathcal{V} : x_i = 1\},$$

and if $\mathcal{A} \subseteq \mathcal{V}$, then $\text{supp}^{-1}(\mathcal{A})$ is the vector $\mathbf{x} \in \mathbb{B}^n$ whose support is $\mathcal{A}$. Hence, $\text{supp}^{-1}(\text{supp}(\mathbf{x})) = \mathbf{x}$.

3 Bilinear programming formulation

Since minimizing $\mathcal{C}(\mathcal{S})$ in (1) is equivalent to maximizing $\mathcal{C}(\mathcal{A} \cup \mathcal{B})$, we may view the VSP as the following maximization problem:

$$\begin{aligned} &\max_{\mathcal{A},\mathcal{B}\subseteq\mathcal{V}} \quad \mathcal{C}(\mathcal{A}\cup\mathcal{B}) \\ &\text{subject to} \quad \mathcal{A}\cap\mathcal{B}=\emptyset, \quad (\mathcal{A}\times\mathcal{B})\cap\mathcal{E}=\emptyset, \\ &\ell_a \le \mathcal{W}(\mathcal{A}) \le u_a, \quad \text{and} \quad \ell_b \le \mathcal{W}(\mathcal{B}) \le u_b. \end{aligned} \tag{2}$$

Let $\mathbf{A}$ be the $n \times n$ adjacency matrix for the graph G defined by $a_{ij} = 1$ if $(i, j) \in \mathcal{E}$ and $a_{ij} = 0$ otherwise, and let $\mathbf{I}$ be the $n \times n$ identity matrix. For any pair of subsets $\mathcal{A}, \mathcal{B} \subseteq \mathcal{V}$, let $\mathbf{x} = \mathrm{supp}^{-1}(\mathcal{A})$ and $\mathbf{y} = \mathrm{supp}^{-1}(\mathcal{B})$ be the associated binary support vectors. Observe that

$$\mathbf{x}^{\mathsf{T}}(\mathbf{A}+\mathbf{I})\mathbf{y} = \sum_{i=1}^{n}\sum_{j=1}^{n} x_i a_{ij} y_j + \sum_{i=1}^{n} x_i y_i = \sum_{x_i=1}\sum_{y_j=1} a_{ij} + \sum_{x_i=y_i=1} 1.$$

By definition, $x_i = 1$ or $y_j = 1$ if and only if $i \in \mathcal{A}$ or $j \in \mathcal{B}$ respectively, and $a_{ij} = 1$ if and only $(i, j) \in \mathcal{E}$. Consequently, we have

$$\mathbf{x}^{\mathsf{T}}(\mathbf{A}+\mathbf{I})\mathbf{y} = \sum_{i\in\mathcal{A}}\sum_{j\in\mathcal{B}} a_{ij} + \sum_{i\in\mathcal{A}\cap\mathcal{B}} 1 = |(\mathcal{A}\times\mathcal{B})\cap\mathcal{E}| + |\mathcal{A}\cap\mathcal{B}|.$$

So, the constraints $\mathcal{A} \cap \mathcal{B} = \emptyset$ and $(\mathcal{A} \times \mathcal{B}) \cap \mathcal{E} = \emptyset$ in (2) hold if and only if

$$\mathbf{x}^{\mathsf{T}}(\mathbf{A}+\mathbf{I})\mathbf{y} = 0, \quad \text{where } \mathbf{x} = \mathrm{supp}(\mathcal{A}) \text{ and } \mathbf{y} = \mathrm{supp}(\mathcal{B}). \tag{3}$$

Moreover, for these support vectors, we have $\mathcal{C}(\mathcal{A}\cup\mathcal{B}) = \mathbf{c}^{\mathsf{T}}(\mathbf{x}+\mathbf{y})$ and $\mathcal{W}(\mathcal{A}) = \mathbf{w}^{\mathsf{T}}\mathbf{x}$, where $\mathbf{c}$ and $\mathbf{w}$ are the n-dimensional vectors which store the costs c_i and weights w_i of vertices, respectively. Hence, a binary formulation of (2) is

$$\begin{aligned} &\max_{\mathbf{x},\mathbf{y}\in\mathbb{B}^n} \quad \mathbf{c}^{\mathsf{T}}(\mathbf{x}+\mathbf{y}) \\ &\text{subject to } \mathbf{x}^{\mathsf{T}}(\mathbf{A}+\mathbf{I})\mathbf{y} = 0, \\ &\ell_a \le \mathbf{w}^{\mathsf{T}}\mathbf{x} \le u_a, \quad \text{and} \quad \ell_b \le \mathbf{w}^{\mathsf{T}}\mathbf{y} \le u_b. \end{aligned} \tag{4}$$

Now, consider the following problem in which the quadratic constraint of (4) has been relaxed:

$$\begin{aligned} &\max_{\mathbf{x},\mathbf{y}\in\mathbb{B}^n} \quad f(\mathbf{x},\mathbf{y}) := \mathbf{c}^{\mathsf{T}}(\mathbf{x}+\mathbf{y}) - \gamma\mathbf{x}^{\mathsf{T}}(\mathbf{A}+\mathbf{I})\mathbf{y} \\ &\text{subject to} \quad \ell_a \le \mathbf{w}^{\mathsf{T}}\mathbf{x} \le u_a \quad \text{and} \quad \ell_b \le \mathbf{w}^{\mathsf{T}}\mathbf{y} \le u_b. \end{aligned} \tag{5}$$

Here, $\gamma \in \mathbb{R}$. Notice that $\gamma\mathbf{x}^{\mathsf{T}}(\mathbf{A}+\mathbf{I})\mathbf{y}$ acts as a penalty term in (5) when $\gamma \ge 0$, since $\mathbf{x}^{\mathsf{T}}(\mathbf{A}+\mathbf{I})\mathbf{y} \ge 0$ for every $\mathbf{x}, \mathbf{y} \in \mathbb{B}^n$. Moreover, (5) gives a relaxation of (4), since the constraint (3) is not enforced. Problem (5) is feasible since the VSP (2) is feasible by assumption. The following proposition gives conditions under which (5) is essentially equivalent to (4) and (2).

Proposition 1 *If* $\mathbf{w} \geq \mathbf{1}$ *and* $\gamma > 0$ *with* $\gamma \geq \max\{c_i : i \in \mathcal{V}\}$, *then for any feasible point* $(\mathbf{x}, \mathbf{y})$ *in* (5) *satisfying*

$$f(\mathbf{x}, \mathbf{y}) \geq \gamma(\ell_a + \ell_b), \tag{6}$$

there is a feasible point $(\overline{\mathbf{x}}, \overline{\mathbf{y}})$ *in* (5) *such that*

$$f(\overline{\mathbf{x}}, \overline{\mathbf{y}}) \geq f(\mathbf{x}, \mathbf{y}) \quad \textit{and} \quad \overline{\mathbf{x}}^{\mathsf{T}}(\mathbf{A} + \mathbf{I})\overline{\mathbf{y}} = 0. \tag{7}$$

Hence, if the optimal objective value in (5) *is at least* $\gamma(\ell_a + \ell_b)$, *then there exists an optimal solution* $(\mathbf{x}^*, \mathbf{y}^*)$ *to* (5) *such that an optimal solution to* (2) *is given by*

$$\mathcal{A} = \mathrm{supp}(\mathbf{x}^*), \quad \mathcal{B} = \mathrm{supp}(\mathbf{y}^*), \quad \textit{and} \;\; \mathcal{S} = \mathcal{V} \setminus \mathcal{A} \cup \mathcal{B}. \tag{8}$$

Proof Let $(\mathbf{x}, \mathbf{y})$ be a feasible point in (5) satisfying (6). Since $\mathbf{x}$, $\mathbf{y}$, and $(\mathbf{A} + \mathbf{I})$ are nonnegative, we have $\mathbf{x}^{\mathsf{T}}(\mathbf{A} + \mathbf{I})\mathbf{y} \geq 0$. If $\mathbf{x}^{\mathsf{T}}(\mathbf{A} + \mathbf{I})\mathbf{y} = 0$, then we simply take $\overline{\mathbf{x}} = \mathbf{x}$ and $\overline{\mathbf{y}} = \mathbf{y}$, and (7) is satisfied. Now suppose instead that

$$\mathbf{x}^{\mathsf{T}}(\mathbf{A} + \mathbf{I})\mathbf{y} > 0. \tag{9}$$

Then,

$$\begin{aligned} \gamma(\ell_a + \ell_b) \leq f(\mathbf{x}, \mathbf{y}) &= \mathbf{c}^{\mathsf{T}}(\mathbf{x} + \mathbf{y}) - \gamma\mathbf{x}^{\mathsf{T}}(\mathbf{A} + \mathbf{I})\mathbf{y} && (10)\\ &< \mathbf{c}^{\mathsf{T}}(\mathbf{x} + \mathbf{y}) && (11)\\ &\leq \gamma\mathbf{1}^{\mathsf{T}}(\mathbf{x} + \mathbf{y}). && (12) \end{aligned}$$

Here, (10) is due to (6), (11) is due to (9) and the assumption that $\gamma > 0$, and (12) holds by the assumption that $\gamma \geq \max\{c_i : i \in \mathcal{V}\}$. It follows that either $\mathbf{1}^{\mathsf{T}}\mathbf{x} > \ell_a$ or $\mathbf{1}^{\mathsf{T}}\mathbf{y} > \ell_b$.

Assume without loss of generality that $\mathbf{1}^{\mathsf{T}}\mathbf{x} > \ell_a$. Since $\mathbf{x}$ is binary and ℓ_a is an integer, we have

$$\mathbf{1}^{\mathsf{T}}\mathbf{x} \geq \ell_a + 1.$$

Since the entries in $\mathbf{x}$, $\mathbf{y}$, and $(\mathbf{A} + \mathbf{I})$ are all non-negative integers, (9) implies that there exists an index i such that $(\mathbf{A} + \mathbf{I})_i\mathbf{y} \geq 1$ and $x_i = 1$ (recall that subscripts on a matrix correspond to the rows). If $\hat{\mathbf{x}} = \mathbf{x} - \mathbf{e}_i$, then $(\hat{\mathbf{x}}, \mathbf{y})$ is feasible in problem (5) since $u_a \geq \mathbf{w}^{\mathsf{T}}\mathbf{x} > \mathbf{w}^{\mathsf{T}}\hat{\mathbf{x}}$ and

$$\mathbf{w}^{\mathsf{T}}\hat{\mathbf{x}} \geq \mathbf{1}^{\mathsf{T}}\hat{\mathbf{x}} = \mathbf{1}^{\mathsf{T}}\mathbf{x} - 1 \geq \ell_a.$$

Input: A binary feasible point $(\mathbf{x}, \mathbf{y})$ for (5) satisfying (6)
while ($\mathbf{x}^{\mathsf{T}}(\mathbf{A}+\mathbf{I})\mathbf{y} > 0$)
    if ($\mathbf{1}^{\mathsf{T}}\mathbf{x} > \ell_a$)
        Choose i such that $x_i = 1$ and $(\mathbf{A}+\mathbf{I})_i\mathbf{y} \geq 1$.
        Set $x_i = 0$.
    else if ($\mathbf{1}^{\mathsf{T}}\mathbf{y} > \ell_b$)
        Choose i such that $y_i = 1$ and $(\mathbf{A}+\mathbf{I})_i\mathbf{x} \geq 1$.
        Set $y_i = 0$.
    end if
end while

Algorithm 1 Convert a binary feasible point for (5) into a vertex separator without decreasing the objective function value.

Here the first inequality is due to the assumption that $\mathbf{w} \geq \mathbf{1}$. Furthermore,

$$f(\hat{\mathbf{x}}, \mathbf{y}) = f(\mathbf{x}, \mathbf{y}) - c_i + \gamma(\mathbf{A}+\mathbf{I})_i\mathbf{y} \geq f(\mathbf{x}, \mathbf{y}) - c_i + \gamma \geq f(\mathbf{x}, \mathbf{y}), \tag{13}$$

since $(\mathbf{A}+\mathbf{I})_i\mathbf{y} \geq 1$, $\gamma \geq 0$, and $\gamma \geq c_i$. We can continue to set components of $\mathbf{x}$ and $\mathbf{y}$ to 0 until reaching a binary feasible point $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ for which $\bar{\mathbf{x}}^{\mathsf{T}}(\mathbf{A}+\mathbf{I})\bar{\mathbf{y}} = 0$ and $f(\bar{\mathbf{x}}, \bar{\mathbf{y}}) \geq f(\mathbf{x}, \mathbf{y})$. This completes the proof of the first claim in the proposition.

Now, if the optimal objective value in (5) is at least $\gamma(\ell_a + \ell_b)$, then by the first part of the proposition, we may find an optimal solution $(\mathbf{x}^*, \mathbf{y}^*)$ satisfying (3); hence, $(\mathbf{x}^*, \mathbf{y}^*)$ is feasible in (4). Since (5) is a relaxation of (4), $(\mathbf{x}^*, \mathbf{y}^*)$ is optimal in (4). Hence, the partition $(\mathcal{A}, \mathcal{S}, \mathcal{B})$ defined by (8) is optimal in (2). This completes the proof. □

Algorithm 1 represents the procedure used in the proof of Proposition 1 to move from a feasible point in (5) to a feasible point $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ satisfying (7).

Remark 1 There is typically an abundance of feasible points in (5) satisfying (6). For example, in the common case where $\gamma = c_i = w_i = 1$ for each i, (6) is satisfied whenever $\mathbf{x} = \mathrm{supp}^{-1}(\mathcal{A})$ and $\mathbf{y} = \mathrm{supp}^{-1}(\mathcal{B})$ for a pair of feasible sets $\mathcal{A}$ and $\mathcal{B}$ in (2), since in this case

$$f(\mathbf{x}, \mathbf{y}) = \mathbf{c}^{\mathsf{T}}(\mathbf{x}+\mathbf{y}) = \mathbf{w}^{\mathsf{T}}\mathbf{x} + \mathbf{w}^{\mathsf{T}}\mathbf{y} = \mathcal{W}(\mathcal{A}) + \mathcal{W}(\mathcal{B}) \geq \ell_a + \ell_b = \gamma(\ell_a + \ell_b).$$

The algorithms developed in the next section find and improve upon vertex separators by generating approximate solutions to the binary program (5). A key step towards generating these solutions is in solving the following continuous relaxation, which provides an upper bound on the optimal objective value of (5):

$$\begin{aligned} &\max_{\mathbf{x},\mathbf{y}\in\mathbb{R}^n} \quad f(\mathbf{x}, \mathbf{y}) := \mathbf{c}^{\mathsf{T}}(\mathbf{x}+\mathbf{y}) - \gamma\mathbf{x}^{\mathsf{T}}(\mathbf{A}+\mathbf{I})\mathbf{y} \\ &\quad \text{subject to } \mathbf{0} \leq \mathbf{x} \leq \mathbf{1}, \quad \mathbf{0} \leq \mathbf{y} \leq \mathbf{1}, \quad \ell_a \leq \mathbf{w}^{\mathsf{T}}\mathbf{x} \leq u_a, \quad \text{and} \quad \ell_b \leq \mathbf{w}^{\mathsf{T}}\mathbf{y} \leq u_b. \end{aligned} \tag{14}$$

The upper bound provided by (14) is typically very tight in practice. In fact, in [19] the authors showed that (14) is in a sense equivalent to (5) in the case where $\mathbf{c} \geq \mathbf{0}$, $\mathbf{w} = \mathbf{1}$, and the upper and lower bounds on $\mathcal{A}$ and $\mathcal{B}$ are integers. In particular, the following result was proved:

Theorem 1 *(*see [19, Theorem 2.1, Part 1]*) Suppose that* $u_a, \ell_a, u_b, \ell_b \in \mathbb{Z}$, $\mathbf{w} = \mathbf{1}$, $\mathbf{c} \geq \mathbf{0}$, *and* $\gamma \geq \max\{c_i : i \in \mathcal{V}\} > 0$. *If* (2) *is feasible and the optimal objective value in* (2) *is at least* $\gamma(\ell_a + \ell_b)$, *then* (14) *has a binary optimal solution* $(\mathbf{x}, \mathbf{y}) \in \mathbb{B}^{2n}$ *satisfying* (3).

In the proof of Theorem 1, a step-by-step procedure is given for moving from any feasible point $(\mathbf{x}, \mathbf{y})$ in (14) to a binary point $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ satisfying $f(\bar{\mathbf{x}}, \bar{\mathbf{y}}) \geq f(\mathbf{x}, \mathbf{y})$. Thus, when the assumptions of Theorem 1 are satisfied the VSP may be solved with a 4-step procedure:

1. Obtain an optimal solution to the continuous bilinear program (14).
2. Move to a binary optimal solution using the algorithm of [19, Theorem 2.1, Part 1].
3. Convert the binary solution of (14) to a separator using Algorithm 1.
4. Construct an optimal partition via (8).

When G has a small number of vertices, the dimension of the bilinear program (14) is small, and the above approach may be effective, depending mainly on the continuous optimization algorithm one chooses for Step 1. However, since the objective function in (14) is non-concave, the number of local maximizers in (14) grows quickly as $|\mathcal{V}|$ becomes large and solving the bilinear program becomes increasingly difficult.

In order to find good approximate solutions to (14) when G is large, we will incorporate the 4-step procedure (with some modifications) into a multilevel framework (see Sect. 5). The basic idea is to coarsen the graph into a smaller graph having a similar structure to the original graph; the VSP is then solved for the coarse graph via a procedure similar to the one above, and the solution is uncoarsened to give a solution for the original graph.

At the coarser levels of our algorithm, each vertex represents an aggregate of vertices from the original graph. Hence, in order to keep track of the sizes of the aggregates, weights must be assigned to the vertices in the coarse graphs, which means the assumption of Theorem 1 that $\mathbf{w} = \mathbf{1}$ does not hold at the coarser levels. However, in the general case where $\mathbf{w} > \mathbf{0}$ and $\mathbf{c} \in \mathbb{R}^n$, the following weaker result is obtained:

Definition 1 A point $(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{2n}$ is called *mostly binary* if $\mathbf{x}$ and $\mathbf{y}$ each have at most one non-binary component.

Proposition 2 *If the VSP* (2) *is feasible and* $\gamma \in \mathbb{R}$, *then* (14) *has a mostly binary optimal solution.*

Proof We show that the following stronger property holds:

(P) For any $(\mathbf{x}, \mathbf{y})$ feasible in (14), there exists a piecewise linear path to a feasible point $(\bar{\mathbf{x}}, \bar{\mathbf{y}}) \in \mathbb{R}^{2n}$ which is mostly binary and satisfies $f(\bar{\mathbf{x}}, \bar{\mathbf{y}}) \geq f(\mathbf{x}, \mathbf{y})$.

Let $(\mathbf{x}, \mathbf{y})$ be any feasible point of (14). If $\mathbf{x}$ and $\mathbf{y}$ each have at most one non-binary component, then we are done. Otherwise, assume without loss of generality there exist indices $k \neq l$ such that

$$0 < x_k \leq x_l < 1.$$

Since $\mathbf{w} > \mathbf{0}$, we can define

$$\mathbf{x}(t) := \mathbf{x} + t\left(\frac{1}{w_k}\mathbf{e}_k - \frac{1}{w_l}\mathbf{e}_l\right)$$

for $t \in \mathbb{R}$. Substituting $\mathbf{x} = \mathbf{x}(t)$ in the objective function yields

$$f(\mathbf{x}(t), \mathbf{y}) = f(\mathbf{x}, \mathbf{y}) + td, \quad \text{where } d = \nabla_{\mathbf{x}} f(\mathbf{x}, \mathbf{y})\left(\frac{1}{w_k}\mathbf{e}_k - \frac{1}{w_l}\mathbf{e}_l\right).$$

If $d \geq 0$, then we may increase t from zero until either $x_k(t) = 1$ or $x_l(t) = 0$. In the case where $d < 0$, we may decrease t until either $x_k(t) = 0$ or $x_l(t) = 1$. In either case, the number of non-binary components in $\mathbf{x}$ is reduced by at least one, while the objective value does not decrease by the choice of the sign of t. Feasibility is maintained since $\mathbf{w}^{\mathsf{T}}\mathbf{x}(t) = \mathbf{w}^{\mathsf{T}}\mathbf{x}$. We may continue moving components to bounds in this manner until $\mathbf{x}$ has at most one non-binary component. The same procedure may be applied to $\mathbf{y}$. In this way, we will arrive at a feasible point $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ such that $\bar{\mathbf{x}}$ and $\bar{\mathbf{y}}$ each have at most one non-binary component and $f(\bar{\mathbf{x}}, \bar{\mathbf{y}}) \geq f(\mathbf{x}, \mathbf{y})$. This proves (P), which completes the proof. □

The proof of Proposition 2 was constructive. A nonconstructive proof goes as follows: Since the quadratic program (14) is bilinear, there exists an optimal solution lying at an extreme point [28]. At an extreme point of the feasible set of (14), exactly $2n$ linearly independent constraints are active. Since there can be at most n linearly independent constraints which are active at $\mathbf{x}$, and similarly for $\mathbf{y}$, there must exist exactly n linearly independent constraints which are active at $\mathbf{x}$; in particular, at least $n - 1$ components of $\mathbf{x}$ must lie at a bound, and similarly for $\mathbf{y}$. Therefore, $(\mathbf{x}, \mathbf{y})$ is mostly binary. In the case where $\mathbf{w} \neq \mathbf{1}$, there may exist extreme points of the feasible set which are not binary; for example, consider $n = 3$, $\ell_a = \ell_b = 1$, $u_a = u_b = 2$, $\mathbf{w} = (1, 1, 2)$, $\mathbf{x} = (1, 0, 0.5)$, and $\mathbf{y} = (0, 1, 0.5)$.

Often, the conclusion of Proposition 2 can be further strengthened to assert the existence of a solution $(\mathbf{x}, \mathbf{y})$ of (14) for which either $\mathbf{x}$ or $\mathbf{y}$ is completely binary, while the other variable has at most one nonbinary component. The rationale is the following: Suppose that $(\mathbf{x}, \mathbf{y})$ is a mostly binary optimal solution and without loss of generality x_i is a nonbinary component of $\mathbf{x}$. Substituting $\mathbf{x}(t) = \mathbf{x} + t\mathbf{e}_i$ in the objective function we obtain

$$f(\mathbf{x}(t), \mathbf{y}) = f(\mathbf{x}, \mathbf{y}) + td, \quad d = \nabla_{\mathbf{x}} f(\mathbf{x}, \mathbf{y})\mathbf{e}_i.$$

Input: A feasible point $(\mathbf{x}, \mathbf{y})$ for the continuous bilinear program (14).
while ($\mathbf{x}$ has at least 2 nonbinary components)
  Choose $i, j \in \mathcal{V}$ such that $x_i, x_j \in (0, 1)$.
  Update $\mathbf{x} \leftarrow \mathbf{x} + t(\frac{1}{w_i}\mathbf{e}_i - \frac{1}{w_j}\mathbf{e}_j)$, choosing t to ensure that:
    (a) $f(\mathbf{x}, \mathbf{y})$ does not decrease,
    (b) either $x_i \in \mathbb{B}$ or $x_j \in \mathbb{B}$,
    (c) $\mathbf{x}$ feasible in (14).
end while
while ($\mathbf{y}$ has at least 2 nonbinary components)
  Choose $i, j \in \mathcal{V}$ such that $y_i, y_j \in (0, 1)$.
  Update $\mathbf{y} \leftarrow \mathbf{y} + t(\frac{1}{w_i}\mathbf{e}_i - \frac{1}{w_j}\mathbf{e}_j)$, choosing t to ensure that:
    (a) $f(\mathbf{x}, \mathbf{y})$ does not decrease,
    (b) either $y_i \in \mathbb{B}$ or $y_j \in \mathbb{B}$,
    (c) $\mathbf{y}$ feasible in (14).
end while

Algorithm 2 Convert a feasible point for (14) into a mostly binary feasible point without decreasing the objective value.

If $d \geq 0$, we increase t, while if $d < 0$, we decrease t; in either case, the objective function $f(\mathbf{x}(t), \mathbf{y})$ cannot decrease. If

$$\ell_a + w_i \leq \mathbf{w}^\mathsf{T}\mathbf{x} \leq u_a - w_i, \tag{15}$$

then we can let t grow in magnitude until either $x_i(t) = 0$ or $x_i(t) = 1$, while complying with the bounds $\ell_a \leq \mathbf{w}^\mathsf{T}\mathbf{x}(t) \leq u_a$. In applications, either the inequality (15) holds, or an analogous inequality $\ell_b + w_j \leq \mathbf{w}^\mathsf{T}\mathbf{y} \leq u_a - w_j$ holds for $\mathbf{y}$, where y_j is a nonbinary component of $\mathbf{y}$. The reason that one of these inequalities holds is that we typically have $u_a = u_b > \mathcal{W}(\mathcal{V})/2$, which implies that the upper bounds $\mathbf{w}^\mathsf{T}\mathbf{x} \leq u_a$ and $\mathbf{w}^\mathsf{T}\mathbf{y} \leq u_b$ cannot be simultaneously active. On the other hand, the lower bounds $\mathbf{w}^\mathsf{T}\mathbf{x} \geq \ell_a$ and $\mathbf{w}^\mathsf{T}\mathbf{y} \geq \ell_b$ are often trivially satisfied when ℓ_a and ℓ_b are small numbers like one.

Algorithm 2 represents the procedure used in the proof of Proposition 2 to convert a given feasible point for (14) into a mostly binary feasible point without decreasing the objective function value. In the case where $\mathbf{w} = \mathbf{1}$, the final point returned by Algorithm 2 is binary.

Although the continuous bilinear problem (14) is not necessarily equivalent to the discrete VSP (2) when $\mathbf{w} \neq \mathbf{1}$, it closely approximates (2) in the sense that it has a mostly binary optimal solution. Since (14) is a relaxation of (4), the objective value at an optimal solution to (14) gives an upper bound on the optimal objective value in (4), and therefore on the optimal objective value in (2). On the other hand, given a mostly binary solution to (14), we can typically push the remaining fractional components to bounds without violating the constraints on $\mathbf{w}^\mathsf{T}\mathbf{x}$ and $\mathbf{w}^\mathsf{T}\mathbf{y}$. Then we apply Algorithm 1 to this binary point to obtain a feasible point in (4), giving a *lower* bound on the optimal objective value in (4) and (2). In the case where $\mathbf{w} = \mathbf{1}$, the upper and lower bounds are equal.

Input: A feasible point $(\mathbf{x}, \mathbf{y})$ for (14) and $\eta > 0$.
while ($(\mathbf{x}, \mathbf{y})$ not stationary point for (14))
    $\hat{\mathbf{x}} \leftarrow \text{argmax}\ \{f(\mathbf{x}, \mathbf{y}) : \mathbf{x} \in \mathcal{P}_a\}$
    $\hat{\mathbf{y}} \leftarrow \text{argmax}\ \{f(\mathbf{x}, \mathbf{y}) : \mathbf{y} \in \mathcal{P}_b\}$
    if ($f(\hat{\mathbf{x}}, \hat{\mathbf{y}}) > \max\ \{f(\hat{\mathbf{x}}, \mathbf{y}), f(\mathbf{x}, \hat{\mathbf{y}})\}(\eta + 1)$)
        $(\mathbf{x}, \mathbf{y}) \leftarrow (\hat{\mathbf{x}}, \hat{\mathbf{y}})$
    else if ($f(\hat{\mathbf{x}}, \mathbf{y}) > f(\mathbf{x}, \hat{\mathbf{y}})(\eta + 1)$)
        $\mathbf{x} \leftarrow \hat{\mathbf{x}}$
    else
        $\mathbf{y} \leftarrow \hat{\mathbf{y}}$
    end if
end while
return $(\mathbf{x}, \mathbf{y})$

Algorithm 3 MCA: A modified version of Konno's Mountain Climbing Algorithm for generating a stationary point for (14).

4 Finding and improving upon vertex separators

Due to the bilinear structure of the objective in (14), one strategy for generating an approximate solution is to start from any feasible point and successively optimize over $\mathbf{x}$ and then over $\mathbf{y}$ while leaving the other variable fixed. This is a specific instance of the general Mountain Climbing Algorithm of Konno [28] for solving bilinear programs. Let $\mathcal{P}_a$ and $\mathcal{P}_b$ denote the feasible sets (polyhedra) associated with $\mathbf{x}$ and $\mathbf{y}$ in (14); that is,

$$\mathcal{P}_i = \{\mathbf{z} \in \mathbb{R}^n : \mathbf{0} \le \mathbf{z} \le \mathbf{1} \ \text{ and } \ \ell_i \le \mathbf{w}^{\mathsf{T}}\mathbf{z} \le u_i\}, \quad i = a, b\,. \tag{16}$$

In Algorithm 3, we employ a variant of the Mountain Climbing Algorithm (MCA) that includes a diagonal step. For a given $(\mathbf{x}, \mathbf{y})$, let $\hat{\mathbf{x}}$ denote the maximizer of $f(\mathbf{x}, \mathbf{y})$ over $\mathbf{x} \in \mathcal{P}_a$, and let $\hat{\mathbf{y}}$ denote the maximizer over $\mathbf{y} \in \mathcal{P}_b$. The diagonal steps to $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ are only taken during the initial iterations when they provide an improvement at least $1 + \eta$ times better than either an individual $\hat{\mathbf{x}}$ or $\hat{\mathbf{y}}$ step, where η is a small constant (10^{-10} in our experiments). After an $\hat{\mathbf{x}}$ or $\hat{\mathbf{y}}$ step is taken, the iterates alternate between $(\hat{\mathbf{x}}, \mathbf{y})$ and $(\mathbf{x}, \hat{\mathbf{y}})$, and hence only one linear program is solved at each iteration.

At each iteration of MCA, computing $\hat{\mathbf{x}}$ or $\hat{\mathbf{y}}$ amounts to solving a linear program where the feasible set is of the form (16). For example, the computation of $\hat{\mathbf{x}}$ in MCA amounts to solving the linear program

$$\max\{\mathbf{g}^{\mathsf{T}}\mathbf{z} : \mathbf{0} \le \mathbf{z} \le \mathbf{1},\ \ell_a \le \mathbf{w}^{\mathsf{T}}\mathbf{z} \le u_a\}, \tag{17}$$

where $\mathbf{g}$ corresponds to the gradient of the objective $f(\mathbf{x}, \mathbf{y})$ with respect to $\mathbf{y}$. We note that (17) is a special case of the convex quadratic knapsack problem studied in [11]. In most VSP applications, the constraint $\ell_a \le \mathbf{w}^{\mathsf{T}}\mathbf{z}$ is satisfied trivially. In this case, the algorithm in [11] for solving (17) introduces a Lagrange multiplier λ for the constraint $\mathbf{w}^{\mathsf{T}}\mathbf{z} \le u_a$ and defines

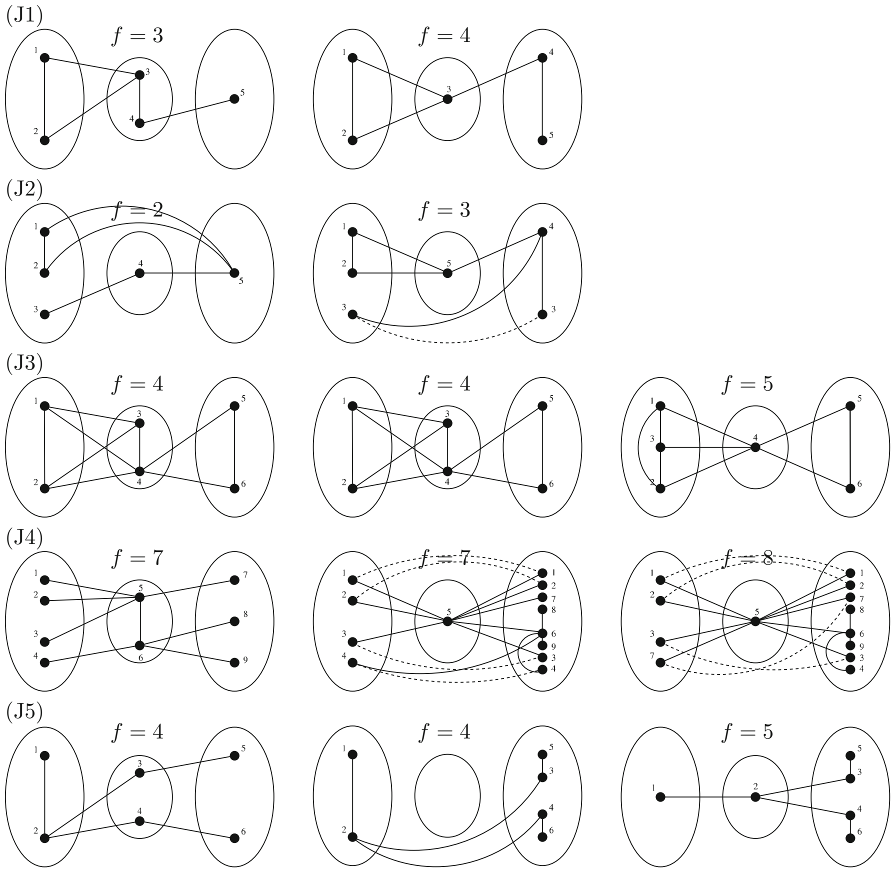

Fig. 1 Examples of the cases (J1)–(J5) in the Appendix where MCA is guaranteed to strictly improve the objective value. The initial partition $\mathcal{A}$, $\mathcal{S}$, $\mathcal{B}$, appears on the left, followed by the next one or two iterates of MCA

$$z_i(\lambda) = \begin{cases} 1 & \text{if } g_i + \lambda w_i > 0, \\ 0 & \text{if } g_i + \lambda w_i < 0, \end{cases}$$

where $z_i(\lambda)$ can be chosen arbitrarily in the interval [0, 1] when $g_i + \lambda w_i = 0$. To solve (17), we start from $\lambda = 0$ and decrease λ until reaching the first $\mathbf{z}(\lambda)$ for which $\mathbf{w}^\mathsf{T}\mathbf{z}(\lambda) \leq u_a$. Since the set $\mathcal{Z} = \{i : g_i + \lambda w_i = 0\}$ could have more than one element, there could be multiple solutions to (17). In order to explore a larger swath of the solution space, we make at most one component $z_i(\lambda)$ fractional for $i \in \mathcal{Z}$, and make the remaining components of $z_i(\lambda)$ for $i \in \mathcal{Z}$ binary; the ones in these remaining binary components are assigned randomly. MCA will typically converge to a stationary point of (14) in a small number of iterations. Moreover, using the implementation described above, the stationary point is almost binary.

In the Appendix, we develop 5 different cases, (J1)–(J5), where MCA is guaranteed to strictly improve the objective value. Specific instances of these cases are shown in Fig. 1, where $\mathbf{c} = \mathbf{w} = \mathbf{1}$, $\ell_a = \ell_b = 1$, and u_a and u_b are sufficiently large. The figure gives for each case the initial partition in the order $\mathcal{A}$, $\mathcal{S}$, $\mathcal{B}$, followed by the next one or two iterations of MCA obtained by maximizing (14) first with respect to $\mathbf{y}$ and then possibly with respect to $\mathbf{x}$ while holding the other variable fixed. In two of these cases, MCA leads to iterates for which one or two vertices satisfy $x_i = y_i = 1$, so the disjointness condition $\mathbf{x}^\mathsf{T}\mathbf{y} = 0$ does not hold. Dotted lines connect vertices that appear in both $\mathcal{A}$ and $\mathcal{B}$. In this case, Proposition 1 and Algorithm 1 can be used to obtain a feasible partition for (1) where $\mathcal{A}$ and $\mathcal{B}$ are disjoint. The objective value in (14) appears above each partition. In the cases (J2), (J4), and (J5) a traditional Fidducia-Matheyses type algorithm based on iteratively moving vertices from $\mathcal{S}$ to one of the two shores and then moving the neighbors in the opposite shore into $\mathcal{S}$ would fail to find an improvement in the initial partition. Indeed, applying the routine METISRefine (from METIS 5.1.0 [24]) to these partitions fails to reduce the size of the separator. Observe that in cases (J4) and (J5), MCA is able to reduce the size of $\mathcal{S}$ by essentially removing "false hub" vertices from the separator and replacing them with a single better vertex. One of the mechanisms by which MCA finds these improvement opportunities (eg. (J4)) is by temporarily violating the disjointness condition, effectively delaying the assignment of a vertex to a particular set until the "optimal" choice can be determined. We remark that the structures described in (J4) and (J5) often arise in communication networks and in networks having a centralized hub of vertices. This suggests that MCA may be particularly effective on graphs coming from communication applications. As we will see, this hypothesis is supported by our computational results in Sect. 6.

There are also many cases in which a vertex separator *cannot* be improved by MCA, but *can* be improved by a Fidduccia-Mattheyses type algorithm. In order to improve the performance of MCA in these cases, we enhance it to include a technique which we call γ-perturbations. Let us also denote the objective function in (14) as f_γ to emphasize its dependence on the penalty parameter γ.

According to our theory, we need to take $\gamma \geq \max\{c_i : i \in \mathcal{V}\}$ to ensure an (approximate) equivalence between the discrete (2) and the continuous VSP (14). The penalty term $-\gamma\mathbf{x}^\mathsf{T}(\mathbf{A}+\mathbf{I})\mathbf{y}$ in the objective function of (14) enforces the constraints $\mathcal{A} \cap \mathcal{B} = \emptyset$ and $(\mathcal{A} \times \mathcal{B}) \cap \mathcal{E} = \emptyset$ of (2). Thus, by decreasing γ, we relax our enforcement of these constraints and place greater emphasis on the cost of the separator. The next proposition will determine the amount by which we must decrease γ in order to ensure that a first-order optimal point $(\mathbf{x}, \mathbf{y})$ of f_γ no longer satisfies the first-order optimality conditions for the perturbed problem $f_{\tilde{\gamma}}$. A standard statement of the first-order optimality conditions for non-linear programs is given in [33]. The following theorem is a special case [18, Proposition 3.1].

Theorem 2 *If* $\gamma \in \mathbb{R}$ *and* $(\mathbf{x}, \mathbf{y})$ *is feasible in* (14)*, then* $(\mathbf{x}, \mathbf{y})$ *satisfies the first-order optimal condition if and only if the following hold:*

(C1) $\nabla_\mathbf{x} f(\mathbf{x}, \mathbf{y})\mathbf{d} \leq 0$ *for every* $\mathbf{d} \in \mathcal{F}_a(\mathbf{x}) \cap \mathcal{D}$,
(C2) $\nabla_\mathbf{y} f(\mathbf{x}, \mathbf{y})\mathbf{d} \leq 0$ *for every* $\mathbf{d} \in \mathcal{F}_b(\mathbf{y}) \cap \mathcal{D}$,

where

$$\mathcal{F}_i(\mathbf{z}) = \left\{ \mathbf{d} \in \mathbb{R}^n : \begin{array}{l} d_j \le 0 \textit{ for all } j \textit{ such that } z_j = 1 \\ d_j \ge 0 \textit{ for all } j \textit{ such that } z_j = 0 \\ \mathbf{w}^{\mathsf{T}}\mathbf{d} \le 0 \textit{ if } \mathbf{w}^{\mathsf{T}}\mathbf{z} = u_i \\ \mathbf{w}^{\mathsf{T}}\mathbf{d} \ge 0 \textit{ if } \mathbf{w}^{\mathsf{T}}\mathbf{z} = l_i \end{array} \right\}, \quad i = a, b, \ \mathbf{z} \in \mathbb{R}^n,$$

and

$$\mathcal{D} = \bigcup_{i,j=1}^{n} \left\{ \mathbf{e}_i, -\mathbf{e}_i, w_j \mathbf{e}_i - w_i \mathbf{e}_j \right\}. \tag{18}$$

The sets $\mathcal{F}_a$ and $\mathcal{F}_b$ are the cones of first-order feasible directions at $\mathbf{x}$ and $\mathbf{y}$. The set $\mathcal{D}$ is a reflective edge description (introduced in [18]) of each of the sets $\mathcal{P}_a$ and $\mathcal{P}_b$; that is, each edge of $\mathcal{P}_a$ is parallel to an element of $\mathcal{D}$. Since $\mathcal{D}$ is a finite set, checking the first-order optimality conditions reduces to testing the conditions (C1) and (C2) for the finite collection of elements from $\mathcal{D}$ that are in the cone of first-order feasible directions.

Proposition 3 *Let $\gamma \in \mathbb{R}$, let $(\mathbf{x}, \mathbf{y})$ be a feasible point in (14) which satisfies the first-order optimality condition* (C1) *for $f = f_\gamma$, and let $\tilde{\gamma} \le \gamma$. Suppose that $\mathbf{w}^{\mathsf{T}}\mathbf{x} > \ell_a$. Then* (C1) *holds at $(\mathbf{x}, \mathbf{y})$ for $f = f_{\tilde{\gamma}}$ if and only if*

$$\tilde{\gamma} \ \ge \ \alpha \ := \begin{cases} \alpha_1, \textit{ if } & \mathbf{w}^{\mathsf{T}}\mathbf{x} < u_a, \\ \alpha_2, \textit{ if } & \mathbf{w}^{\mathsf{T}}\mathbf{x} = u_a, \end{cases}$$

where

$$\alpha_1 = \max \left\{ \frac{c_j}{(\mathbf{A} + \mathbf{I})_j \mathbf{y}} : \ x_j < 1 \textit{ and } (\mathbf{A} + \mathbf{I})_j \mathbf{y} > 0 \right\}, \quad \textit{and}$$

$$\alpha_2 = \inf \left\{ \alpha \in \mathbb{R} : \frac{1}{w_i} \frac{\partial f_\alpha}{\partial x_i}(\mathbf{x}, \mathbf{y}) \le \frac{1}{w_j} \frac{\partial f_\alpha}{\partial x_j}(\mathbf{x}, \mathbf{y}) \ \forall \ x_i < 1 \textit{ and } x_j > 0 \right\}.$$

Proof First, we consider the case where $\mathbf{w}^{\mathsf{T}}\mathbf{x} < u_a$. In this case, the cone of first-order feasible directions at $\mathbf{x}$ is given by

$$\mathcal{F}_a(\mathbf{x}) = \left\{ \mathbf{d} \in \mathbb{R}^n : d_i \ge 0 \text{ when } x_i = 0 \text{ and } d_i \le 0 \text{ when } x_i = 1, \ i = 1, \ldots, n \right\}.$$

It follows that for each $i = 1, \ldots, n$,

$$\begin{array}{rll} \mathbf{e}_i & \in \mathcal{F}_a(\mathbf{x}) & \text{if and only if } \ x_i < 1, \\ -\mathbf{e}_i & \in \mathcal{F}_a(\mathbf{x}) & \text{if and only if } \ x_i > 0, \\ (w_j \mathbf{e}_i - w_i \mathbf{e}_j) & \in \mathcal{F}_a(\mathbf{x}) & \text{if and only if } \ x_i < 1 \text{ and } x_j > 0. \end{array}$$

Hence, the first-order optimality condition (C1) for $f_{\tilde{\gamma}}$ can be expressed as follows:

$$\nabla_{\mathbf{x}} f_{\tilde{\gamma}}(\mathbf{x}, \mathbf{y})\mathbf{e}_i \leq 0 \quad \text{when} \quad x_i < 1, \tag{19}$$

$$\nabla_{\mathbf{x}} f_{\tilde{\gamma}}(\mathbf{x}, \mathbf{y})\mathbf{e}_i \geq 0 \quad \text{when} \quad x_i > 0, \tag{20}$$

$$\nabla_{\mathbf{x}} f_{\tilde{\gamma}}(\mathbf{x}, \mathbf{y})(w_j \mathbf{e}_i - w_i \mathbf{e}_j) \leq 0 \quad \text{when} \quad x_i < 1 \text{ and } x_j > 0. \tag{21}$$

Since (21) is implied by (19) and (20), it follows that (C1) holds if and only if (19) and (20) hold. Since (C1) holds for f_γ, we know that

$$\nabla_{\mathbf{x}} f_\gamma(\mathbf{x}, \mathbf{y})\mathbf{e}_i = c_i - \gamma(\mathbf{A} + \mathbf{I})_i \mathbf{y} \geq 0 \;\; \text{when} \;\; x_i > 0.$$

Hence, since $\tilde{\gamma} \leq \gamma$ and $(\mathbf{A} + \mathbf{I})_i \mathbf{y} \geq 0$,

$$\nabla_{\mathbf{x}} f_{\tilde{\gamma}}(\mathbf{x}, \mathbf{y})\mathbf{e}_i = c_i - \tilde{\gamma}(\mathbf{A} + \mathbf{I})_i \mathbf{y} \geq 0 \quad \text{when } x_i > 0.$$

Hence, (C1) holds with respect to $\tilde{\gamma}$ if and only if (19) holds. Since (C1) holds for $f = f_\gamma$, we have

$$c_j - \gamma(\mathbf{A} + \mathbf{I})_j \mathbf{y} \leq 0 \text{ when } x_j < 1. \tag{22}$$

Hence, for every j such that $x_j < 1$ and $(\mathbf{A} + \mathbf{I})_j \mathbf{y} = 0$, we have

$$c_j - \tilde{\gamma}(\mathbf{A} + \mathbf{I})_j \mathbf{y} = c_j = c_j - \gamma(\mathbf{A} + \mathbf{I})_j \mathbf{y} \leq 0.$$

So, (19) holds if and only if $c_j - \tilde{\gamma}(\mathbf{A} + \mathbf{I})_j \mathbf{y} \leq 0$ for every $j \in \mathcal{J}$; that is, if and only if $\tilde{\gamma} \geq \alpha_1$.

Next, suppose that $\mathbf{w}^\mathsf{T}\mathbf{x} = u_a$. Then, the cone of first-order feasible directions at $\mathbf{x}$ has the constraint $\mathbf{w}^\mathsf{T}\mathbf{d} \leq 0$. Consequently, $\mathbf{e}_i \notin \mathcal{F}_a(\mathbf{x}) \cap \mathcal{D}$ for any i, and the first-order optimality condition (C1) for $f_{\tilde{\gamma}}$ reduces to (20)–(21). As in Part 1, (20) holds, since $\tilde{\gamma} \leq \gamma$. Condition (21) is equivalent to $\tilde{\gamma} \in \Gamma$, where Γ is the set on the right hand side of the definition of α_2. Since (21) holds for $f = f_\gamma$, we have $\gamma \in \Gamma$. Since $\nabla_{\mathbf{x}} f_{\tilde{\gamma}}(\mathbf{x}, \mathbf{y})$ is a affine function of $\tilde{\gamma}$, the set of $\tilde{\gamma}$ satisfying (21) for some i and j such that $x_j > 0$ and $x_i < 1$ is a closed interval, and the intersection of the intervals over all i and j for which $x_j > 0$ and $x_i < 1$ is also a closed interval. Hence, since $\tilde{\gamma} \leq \gamma \in \Gamma$, we have $\tilde{\gamma} \in \Gamma$ if and only if $\tilde{\gamma} \geq \alpha_2$, which completes the proof. □

Remark 4.1 Of course, Proposition 3 also holds when the variables $\mathbf{x}$ and $\mathbf{y}$ and the bounds (ℓ_a, u_a) and (ℓ_b, u_b) are interchanged. Hence, to ensure that the current iterate is not a stationary point of of f_γ, we only need to choose γ strictly smaller than the largest of the two lower bounds gotten from Proposition 3. Given a stationary point $(\mathbf{x}, \mathbf{y})$ for (14), let the function minreduce $(\mathbf{x}, \mathbf{y})$ denote the smallest γ for which $(\mathbf{x}, \mathbf{y})$ remains a stationary point of f_γ. In most applications, u_a and $u_b > \mathcal{W}(\mathcal{V})/2$, $\ell_a = \ell_b = 1$, which implies that at most one of the constraints $\mathbf{w}^\mathsf{T}\mathbf{x} \leq u_a$ or $\mathbf{w}^\mathsf{T}\mathbf{y} \leq u_b$ is active. Note that the bound given by α_2 in Proposition 3 often provides no useful information in the following sense: In a multilevel implementation, the vertex costs (and weights) are often 1 at the finest level, and at coarser levels, the vertex costs may not differ greatly. When the vertex costs are equal, (21) holds when $\tilde{\gamma}$ has the same

Input: A feasible point $(\mathbf{x}, \mathbf{y})$ for (14), $\delta > 0$, and $K \geq 1$.
$(\mathbf{x}, \mathbf{y}) \leftarrow$ MCA $(\mathbf{x}, \mathbf{y})$
$\tilde{\gamma} \leftarrow$ minreduce $(\mathbf{x}, \mathbf{y})$, $k \leftarrow 1$
while ($k \leq K$)
  $\tilde{\gamma} \leftarrow$ reduce $(\tilde{\gamma}, k)$
  $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}) \leftarrow$ MCA $(\mathbf{x}, \mathbf{y}, \tilde{\gamma})$
  $(\mathbf{x}^*, \mathbf{y}^*) \leftarrow$ MCA $(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \gamma)$
  if ($f(\mathbf{x}^*, \mathbf{y}^*) > f(\mathbf{x}, \mathbf{y})(\delta + 1)$)
    $(\mathbf{x}, \mathbf{y}) \leftarrow (\mathbf{x}^*, \mathbf{y}^*)$
    $\tilde{\gamma} \leftarrow$ minreduce $(\mathbf{x}, \mathbf{y})$
  end if
  $k \leftarrow k + 1$
end while
return $(\mathbf{x}, \mathbf{y})$

Algorithm 5 MCA_GR: A refinement algorithm for (14) which incorporates γ-refinements.

sign as γ; that is, as long as $\tilde{\gamma} \geq 0$, which implies that $\alpha_2 = 0$. Since α_1 is typically positive, minreduce $(\mathbf{x}, \mathbf{y})$ often corresponds to α_1.

Algorithm 5, denoted MCA_GR, approximately solves (14), using MCA in conjunction with γ-refinements. Here, the notation MCA$(\mathbf{x}, \mathbf{y}, \tilde{\gamma})$ indicates that the MCA algorithm is applied to the point $(\mathbf{x}, \mathbf{y})$ using $\tilde{\gamma}$ in place of γ as the penalty parameter. In our implementation of MCA_GR, we used the function *reduce* which was defined as follows:

$$\text{reduce } (\tilde{\gamma}, k) = \begin{cases} \tilde{\gamma} - 10^{-6} & \text{when } k = 1, \\ \tilde{\gamma}/2 & \text{when } k > 1. \end{cases}$$

5 Multilevel algorithm

We now give an overview of a multilevel algorithm, which we call BLP, based on the bilinear programming formulation of the vertex separator problem. The algorithm consists of three phases: coarsening, solving, and uncoarsening.

Coarsening Vertices are visited one by one and each vertex is matched with an unmatched adjacent vertex, whenever one exists. Matched vertices are merged together to form a single vertex having a cost and weight equal to the sum of the costs and weights of the constituent vertices. This coarsening process repeats until the graph has fewer than 75 vertices or fewer than 10 edges.

The goal of the coarsening phase is to reduce the number of degrees of freedom in the problem, while preserving its structure so that the solutions obtained for the coarse problems give a good approximation to the solution for the original problem. We consider two matching rules: random and heavy-edge. In random matching, we randomly combine pairs of vertices in the graph. To implement heavy-edge matching, we associated weights with each edge in the graph. When two edges are combined during the coarsening process, the new edge weight is the sum of the weights of the combined edges. In the coarsening, each vertex is matched with an unmatched neighbor such that the edge between them has the greatest weight over all unmatched neighbors. Heavy edge matching rules have been used in multilevel algorithms such as [20,24], and were originally developed for edge-cut problems. In our initial experiments, we also considered a third rule based on an *algebraic distance* [8] between vertices. However, the results were not significantly different from heavy-edge matching, which is not surprising, since (like heavy-edge rules) the algebraic distance was originally developed for minimizing edge-cuts.

Solving For each of the graphs in the multilevel hierarchy, we approximately solve (14) using MCA_GR. For the coarsest graph, the starting guess is $x_i = u_a/\mathcal{W}(\mathcal{V})$ and $y_i = u_b/\mathcal{W}(\mathcal{V})$, $i = 1, 2, \ldots, n$. For the finer graphs, a starting guess is obtained from the next coarser level using the uncoarsening process described below. After MCA_GR terminates, Algorithm 3.2 along with the modification discussed after Proposition 2 are used to obtain a mostly binary approximation to a solution of (14). Algorithm 3.1 is then used to convert the binary solution into a vertex separator.

Uncoarsening We use the solution for the vertex separator problem computed at any level in the multilevel hierarchy as a starting guess for the solution at the next finer level. Sophisticated cycling techniques like the W- or F-cycle [4] were not implemented. A starting guess for the next finer graph is obtained by unmatching vertices in the coarser graph. Suppose that we are uncoarsening from level l to $l-1$ and $(\mathbf{x}^l, \mathbf{y}^l)$ denotes the solution computed at level l. If vertex i at level l is obtained by matching vertices j and k at level $l-1$, then our starting guess for $(\mathbf{x}^{l-1}, \mathbf{y}^{l-1})$ is simply $(x_j^{l-1}, y_j^{l-1}) = (x_i^l, y_i^l)$ and $(x_k^{l-1}, y_k^{l-1}) = (x_i^l, y_i^l)$.

6 Numerical results

The multilevel algorithm BLP was programmed in C++ and compiled using g++ with optimization O3 on a Dell Precision T7610 Workstation with a Dual Intel Xeon Processor E5-2687W v2 (16 physical cores, 3.4GHz, 25.6MB cache, 192GB memory). When solving the linear programs (17) arising in MCA, it was convenient to sort the ratios g_i/w_i; the sorting phase was carried out by calling $std :: sort$, the $(O(n \log n))$ sorting routine implemented in the C++ standard library. The number of iterations of the main loop in MCA_GR was capped at $K = 10$. On each level in the multilevel hierarchy, after the uncoarsening phase, MCA_GR was called repeatedly until the solution failed to improve, but at most 10 times. We used the parameters $\eta = 10^{-10}$ and $\delta = 10^{-13}$ for MCA and MCA_GR, respectively.

We evaluated the performance of BLP by making comparisons with the routine

METIS_ComputeVertexSeparator,

which is an implementation of a multilevel Fidduccia-Mattheyses-like algorithm for the VSP available from METIS 5.1.0 [24]. The following options were used:

METIS_IPTYPE_NODE (coarsest problem solved with node growth scheme),
METIS_RTYPE_SEP2SIDED (Fiduccia–Mattheyses-like refinement scheme),
METIS_OPTION_NITER = 1,000,000 (maximum refinement iterations).

In a preliminary experiment, we also considered the refinement option

METIS_RTYPE_SEP1SIDED,

but the results obtained were not significantly different. On average, the option

METIS_RTYPE_SEP2SIDED

provided slightly better results.

For our experiments, we compiled a test set of 52 sparse graphs with n ranging from $n = 1,000$ to $n = 139,752$ and with sparsities ranging from 1.32×10^{-2} to 5.65×10^{-5}, where sparsity is defined as the ratio $\frac{|\mathcal{E}|}{n(n-1)}$ (recall that $|\mathcal{E}|$ is equal to twice the number of edges). Our graphs were taken from the Stanford SNAP database [22], the University of Florida Sparse Matrix Library [10], and from a test set from [37] consisting of graphs which were specifically designed to be challenging for graph partitioning. Nine of the graphs from our test set represent 2D or 3D meshes, 15 represent either a communication, collaboration, or social network, and the remaining 31 come from miscellaneous applications. We label these groups with the letters M (mesh), C (communication), and O (other), respectively. In order to enable future comparisons with our solver, we have made this benchmark set available at http://people.cs.clemson.edu/~isafro/data.html as well as in the supplementary material for this paper.

After our experiments, we found that it was useful to further subdivide the O instances into two groups based on the frequency of structures of the form (J4) (see Appendix) in the final partition. We considered (J4), and not (J5), only because the structure appearing in (J5) seemed more difficult to detect. One necessary condition for the structure (J4) to arise is for there to exist a vertex $i \in \mathcal{A}$ such that $\mathcal{N}(i) \cap \mathcal{A} = \emptyset$ or a vertex $j \in \mathcal{B}$ such that $\mathcal{N}(j) \cap \mathcal{B} = \emptyset$. Hence, we (upper-)estimated the number of (J4) structures in each graph by finding an approximate vertex separator (via a call to METIS_ComputeVertexSeparator), and calculating the percentage $p(G)$ of vertices i lying in $\mathcal{A} \cup \mathcal{B}$ for which the containment $\mathcal{N}(i) \subseteq \mathcal{S}$ held. A graph G in group O was moved to OC if $p(G) \geq 5\%$ and to OO otherwise. Table 1 gives the statistics for each graph: number of vertices, number of edges, sparsity, min/max/average degree, and $p(G)$. Note that $p(G) \geq 5\%$ for all but two C graphs, and $p(G) < 5\%$ for all but two of the M graphs.

Vertex costs c_i and weights w_i were assumed to be 1 at the finest level for all graphs. For the bounds on the shores of the separator, we took $\ell_a = \ell_b = 1$ and $u_a = u_b = \lfloor 0.6n \rfloor$, which are the default bounds used by METIS. (Here $\lfloor r \rfloor$ denotes the largest integer not greater than r.) We considered both random matching and heavy-edge matching schemes for both algorithms. Since both algorithms contain random elements, the algorithms were run using 100 different random seeds for each instance.

Tables 2, 3, 4 and 5 give the average, minimum, and maximum cardinalities of the separators obtained by the algorithms for the 100 different random seeds. For each test problem, the smallest separator size obtained across all four algorithms (METIS or BLP with random or heavy edge matching) is highlighted in bold. Here, METIS_HE and METIS_RM refer to the METIS algorithm with heavy-edge matching and with random matching, respectively. Tables 6 and 7 compare the average separator sizes

Table 1 Test set graphs with their statistics

Graph	$\|\mathcal{V}\|$	$\|\mathcal{E}\|/2$	Sparsity	Degree			p (%)
				Min	Max	Ave	
Type M							
bcsstk17	10,974	208,838	3.47E−03	0	149	38.06	4.78
delaunay_n13	8192	24,547	7.32E−04	3	12	5.99	0.00
jagmesh7	1138	3156	4.88E−03	3	6	5.55	0.00
lshp3466	3466	10,215	1.70E−03	3	6	5.89	0.00
minnesota	2642	3303	9.47E−04	1	5	2.5	0.04
nasa4704	4704	50,026	4.52E−03	5	41	21.27	0.00
netz4504	1961	2578	1.34E−03	2	8	2.63	0.57
sherman1	1000	1375	2.75E−03	0	6	2.75	32.47
sstmodel	3345	9702	1.73E−03	0	17	5.8	18.54
Type C							
ca-HepPh	7241	20,2194	7.71E−03	2	982	55.85	2.41
email-Enron	9660	224,896	4.82E−03	2	2532	46.56	2.03
email-EuAll	16805	76156	5.39E−04	1	3360	9.06	40.68
Erdos992	6100	7515	4.04E−04	0	61	2.46	33.42
netscience	1589	2742	2.17E−03	0	34	3.45	8.06
oregon2_010505	5441	19,505	1.32E−03	1	1888	7.17	31.94
p2p-Gnutella04	10879	39,994	6.76E−04	0	103	7.35	21.48
p2p-Gnutella05	8846	31,839	8.14E−04	1	88	7.2	22.06
p2p-Gnutella06	8717	31,525	8.30E−04	1	115	7.23	20.88
p2p-Gnutella08	6301	20,777	1.05E−03	1	97	6.59	24.79
p2p-Gnutella09	8114	26,013	7.90E−04	1	102	6.41	27.70
p2p-Gnutella24	26,518	65,369	1.86E−04	1	355	4.93	26.80
p2p-Gnutella25	22,687	54,705	2.13E−04	1	66	4.82	27.63
p2p-Gnutella30	36,682	88,328	1.31E−04	1	55	4.82	28.30
p2p-Gnutella31	62586	147,892	7.55E−05	1	95	4.73	25.09
Type OO							
bcspwr09	1723	2394	1.61E−03	1	14	2.78	0.18
c-38	8127	34781	1.05E−03	1	888	8.56	0.16
c-43	11125	56275	9.09E−04	1	2619	10.12	0.05
crystm01	4875	50,232	4.23E−03	7	26	20.61	0.00
fxm3_6	5026	44,500	3.52E−03	3	128	17.71	0.00
G42	2000	11779	5.89E−03	4	249	11.78	0.45
net25	9520	195,840	4.32E−03	2	138	41.14	0.00
Peko01	7849	533,03	1.73E−03	2	62	13.58	0.00
USpowerGrid	4941	6594	5.40E−04	1	19	2.67	0.16

Table 1 continued

Graph	$\vert\mathcal{V}\vert$	$\vert\mathcal{E}\vert/2$	Sparsity	Degree Min	Max	Ave	p (%)
barth5_1Ksep_50in_5Kout	32212	101,805	1.96E−04	1	22	6.32	0.00
bcsstk30_500sep_10in_1Kout	58348	2,016,578	1.18E−03	0	219	69.12	0.02
bump2_e18_aa01_model1_crew1	56,438	300,801	1.89E−04	1	604	10.66	2.51
c-30_data_data	11,023	62,184	1.02E−03	1	2109	11.28	0.11
c-60_data_cti_cs4	85,830	241,080	6.55E−05	1	2207	5.62	0.44
data_and_seymourl	9167	55,866	1.33E−03	1	229	12.19	2.59
msc10848_300sep_100in_1Kout	21,996	1,221,028	5.05E−03	1	722	111.02	0.01
p0291_seymourl_iiasa	10,498	53,868	9.78E−04	1	229	10.26	0.24
web-NotreDame	56,429	235,285	1.48E−04	1	6852	8.34	2.56
web-Stanford	122,749	1,409,561	1.87E−04	1	35,053	22.97	1.16
wiki-Vote	3809	95,996	1.32E−02	1	1167	50.4	3.00
Type OC							
soc-Epinions1	22,908	389,439	1.48E−03	1	3026	34	10.38
befref_fxm_2_4_air02	14,109	98,224	9.87E−04	1	1531	13.92	41.80
finan512_scagr7-2c_rlfddd	139,752	552,020	5.65E−05	1	669	7.9	30.05
model1_crew1_cr42_south31	45101	189,976	1.87E−04	1	17,663	8.42	14.01
sctap1-2b_and_seymourl	40,174	140,831	1.75E−04	1	1714	7.01	9.19
south31_slptsk	39,668	189,914	2.41E−04	1	17,663	9.58	14.49
vibrobox_scagr7-2c_rlfddd	77,328	435,586	1.46E−04	1	669	11.27	56.09
yeast	2361	6646	2.39E−03	0	64	5.63	21.05

obtained by the algorithms for 100 different random seeds. The column labeled "Wins" gives the percentage of problems where the average separator size for BLP was strictly smaller than the average separator size for METIS. The column labeled "Ave" takes these average separator sizes, forms the relative ratio $\frac{|\mathcal{S}_{METIS}|-|\mathcal{S}_{BLP}|}{|\mathcal{V}|}$ of the average separator sizes, and then computes the average over a test set. The columns labeled "Min" and "Max" replace the average over a test set by either the minimum or maximum. Regardless of the matching scheme used, the average separator size obtained by BLP was smaller than that of METIS for all of the C graphs, with an average improvement of almost 3%, with similar (though slightly worse) results for the OC graphs. We believe that the strong performance of BLP on these graphs is likely due to the relatively large $p(G)$ values, which suggest that improvement opportunities of type (J4) may arise frequently. Likely for the same reason, the performance of BLP was much poorer on the M and OO graphs.

METIS exhibited a clear favorability towards the heavy-edge-based matching scheme (which out-performed the random matching scheme in approximately 82% of the instances). On the other hand, BLP was relatively indifferent to the choice of the matching scheme (the heavy-edge-based scheme performed better in approximately 48% of the instances). However, when each algorithm was run with its optimal matching scheme, BLP still out-performed METIS in approximately 58% of the instances.

Table 2 Comparison of sizes of separators obtained by METIS and BLP on M graphs

	METIS_RM			BLP_RM		
Graph	Ave	Min	Max	Ave	Min	Max
bcsstk17	147.29	138	168	142.99	**126**	296
delaunay_n13	75.49	**69**	90	88.26	72	152
jagmesh7	14.14	**14**	21	20.97	**14**	42
lshp3466	55.45	52	61	65.18	52	107
minnesota	17.34	**14**	23	21.14	15	37
nasa4704	175.45	168	188	175.51	167	187
netz4504	18.29	**16**	23	21.75	**16**	40
sherman1	30.7	29	50	22.14	**18**	30
sstmodel	24.85	22	40	26.98	**20**	39
	METIS_HE			BLP_HE		
Graph	Ave	Min	Max	Ave	Min	Max
bcsstk17	143.82	132	186	163.97	**126**	237
delaunay_n13	74.05	**69**	83	83.74	72	102
jagmesh7	14.03	**14**	15	19.75	**14**	39
lshp3466	55.42	**51**	61	58.28	**51**	106
minnesota	16.79	**14**	23	21.46	**14**	42
nasa4704	176.65	**163**	206	186.93	165	268
netz4504	18.02	17	20	22	**16**	41
sherman1	29.98	28	39	21.58	**18**	27
sstmodel	24.29	22	35	28.44	**20**	42

Figure 2 gives a performance profile comparing the separator sizes obtained by each algorithm with its optimal matching scheme. The profile gives the percentage P of graphs in which each algorithm produced a solution which was within a factor of τ of the best solution (between the two algorithms). The left hand side of the plot gives the percentage of instances for which each algorithm obtained the best solution, while the center and right hand sides of the plot give an indication of the relative robustness of the algorithms. In terms of overall robustness, we conclude that the algorithms are comparable, though for about 5% of the instances the BLP solution was within a factor of between 2 and 2.25 of the METIS solution.

Since each iteration of MCA requires the solution of an LP (which is often more computationally intensive than performing swaps), the running time of BLP is considerably slower than METIS. CPU times for BLP_RM ranged from 0.02 (s) to 66.92 (s), with an average of 6.77 (s), while CPU times for METIS_RM ranged from less than 0.01 (s) to 0.84 (s), with an average of 0.13 (s).

Figure 3 gives a log–log plot of $n = |\mathcal{V}|$ versus CPU time for all 52 instances. The best fit line through the data in the log–log plot has a slope of approximately 1.62, which indicates that the CPU time of BLP is between a linear and a quadratic function of the number of vertices.

Table 3 Comparison of sizes of separators obtained by METIS and BLP on C graphs

	METIS__RM			BLP__RM		
Graph	Ave	Min	Max	Ave	Min	Max
ca-HepPh	767.56	683	851	663.44	610	709
email-Enron	709.29	650	839	488.73	**435**	531
email-EuAll	76.04	**5**	348	11.17	**5**	36
netscience	0.16	**0**	5	0.05	**0**	2
oregon2_010505	79	66	113	52.89	**42**	66
p2p-Gnutella04	2140.5	2089	2200	1602.78	1560	1732
p2p-Gnutella05	1720.54	1687	1750	1370.22	1273	1419
p2p-Gnutella06	1689.01	1641	1727	1277.15	**1183**	1345
p2p-Gnutella08	1042.3	1003	1075	807.3	**780**	830
p2p-Gnutella09	1328.33	1293	1367	1016.61	995	1053
p2p-Gnutella24	3617.52	3538	3685	2563.77	2525	2607
p2p-Gnutella25	3008.89	2931	3095	2142.97	**2090**	2198
p2p-Gnutella30	4692.64	4580	4798	3170.95	3127	3218
p2p-Gnutella31	6904.14	6619	7468	5280.04	5208	5350
	METIS__HE			BLP__HE		
Graph	Ave	Min	Max	Ave	Min	Max
ca-HepPh	753.62	668	839	659.91	**586**	706
email-Enron	687.23	604	804	497.08	447	555
email-EuAll	8.6	**5**	57	7.76	**5**	17
netscience	0.09	**0**	3	0.01	**0**	1
oregon2_010505	58.51	48	70	51.5	**42**	63
p2p-Gnutella04	2054.89	1986	2103	1599.85	**1550**	1725
p2p-Gnutella05	1666.26	1629	1724	1364.78	**1259**	1415
p2p-Gnutella06	1605.74	1568	1653	1271.4	1194	1340
p2p-Gnutella08	1009.23	976	1043	809.32	786	830
p2p-Gnutella09	1287.09	1253	1327	1018.97	**987**	1043
p2p-Gnutella24	3284.79	3203	3380	2567.34	**2519**	2623
p2p-Gnutella25	2761.55	2691	2836	2144.51	2104	2186
p2p-Gnutella30	4267.38	4094	4398	3171.29	**3121**	3252
p2p-Gnutella31	5986.29	5888	6184	5271.12	**5205**	5328

In order to close the gap in running time between BLP and METIS, the BLP code needs further development to take into account the sparse change in successive iterates. The current code takes into account the sparsity of $\mathbf{A}$ when computing a matrix vector product $\mathbf{Ax}$, but it does not take into account the fact the change $\mathbf{x}_k - \mathbf{x}_{k-1}$ and $\mathbf{y}_k - \mathbf{y}_{k-1}$ in successive iterates is often sparse. Instead of computing the product $\mathbf{Ax}_k$ from scratch in each iteration, we could store the prior product $\mathbf{Ax}_{k-1}$ and update it with the very sparse change $\mathbf{A}(\mathbf{x}_k - \mathbf{x}_{k-1})$ to obtain $\mathbf{Ax}_k$. Similarly, when solving the linear programs in MCA, a solution obtained at iteration $k-1$ could be updated to

Table 4 Comparison of sizes of separators obtained by METIS and BLP on OO graphs

	METIS_RM			BLP_RM		
Graph	Ave	Min	Max	Ave	Min	Max
bcspwr09	7.47	**6**	11	11.43	**6**	20
c-38	24.82	**12**	72	47.81	14	108
c-43	140.86	117	156	138.74	115	160
crystm01	66.5	**65**	90	79.5	**65**	85
fxm3_6	60.82	**42**	90	79.67	**42**	143
G42	441.48	427	458	439.54	427	454
net25	676.32	641	714	864.71	**510**	990
Peko01	29.36	**23**	47	45.82	**23**	82
USpowerGrid	9.13	**8**	16	19.37	9	42
vsp_barth5_1Ksep_50in_5Kout	1346.14	**1043**	1530	1551.72	1312	1804
vsp_bcsstk30_500sep_10in_1Kout	636.7	528	844	623.49	**516**	910
vsp_bump2_e18_aa01_model1_crew1	4378.39	4280	4793	3593.84	3534	3935
vsp_c-30_data_data	536.73	471	611	528.24	468	594
vsp_c-60_data_cti_cs4	2636.81	2384	2741	2465.98	**2318**	2957
vsp_data_and_seymourl	1243.14	**1091**	1347	1287.53	1210	1344
vsp_msc10848_300sep_100in_1Kout	523.7	279	715	360.2	279	693
vsp_p0291_seymourl_iiasa	535.98	531	545	519.37	**511**	535
web-NotreDame	437.77	274	611	420.98	210	614
web-Stanford	143.73	**29**	484	270.66	121	366
wiki-Vote	708.97	694	737	709.4	683	766

	METIS_HE			BLP_HE		
Graph	Ave	Min	Max	Ave	Min	Max
bcspwr09	7.52	**6**	13	11.88	7	19
c-38	14.2	**12**	22	41.02	14	99
c-43	141.6	**103**	155	137.72	118	156
crystm01	67.33	**65**	90	77.46	**65**	105
fxm3_6	53.44	**42**	88	76.41	**42**	104
G42	441.04	424	462	438.77	**417**	455
net25	598.21	**510**	915	896.52	574	1005
Peko01	31.06	**23**	45	55.91	24	111
USpowerGrid	8.99	**8**	14	18.35	**8**	34
vsp_barth5_1Ksep_50in_5Kout	1329.88	1131	1451	1573.16	1290	1875
vsp_bcsstk30_500sep_10in_1Kout	752.41	552	1228	865.21	564	1614
vsp_bump2_e18_aa01_model1_crew1	4306.43	4264	4343	3589.46	**3516**	3944
vsp_c-30_data_data	510.07	**453**	594	599.07	496	733
vsp_c-60_data_cti_cs4	2600.51	2525	2684	2516.27	2366	2998
vsp_data_and_seymourl	1252.78	1148	1341	1269.98	1169	1354

Table 4 continued

Graph	METIS__HE Ave	Min	Max	BLP__HE Ave	Min	Max
vsp_msc10848_300sep_100in_1Kout	648.16	279	929	658.07	279	1072
vsp_p0291_seymourl_iiasa	536.25	532	542	519.44	**511**	534
web-NotreDame	399.97	270	518	419.93	**209**	600
web-Stanford	133.52	**29**	575	274.87	97	617
wiki-Vote	704.86	694	731	703.6	**680**	765

Table 5 Comparison of sizes of separators obtained by METIS and BLP on OC graphs

Graph	METIS__RM Ave	Min	Max	BLP__RM Ave	Min	Max
soc-Epinions1	3072.42	2915	3224	2309.64	2270	2342
vsp_befref_fxm_2_4_air02	1464.03	1328	1584	299.6	**282**	327
vsp_finan512_scagr7-2c_rlfddd	7610.25	7400	7883	4693.56	**4584**	4848
vsp_model1_crew1_cr42_south31	2740.18	2558	2894	2348.24	2283	2425
vsp_sctap1-2b_and_seymourl	4269.77	3884	4557	3657.33	3446	3958
vsp_south31_slptsk	2416.25	2350	2498	2219.08	2139	2287
vsp_vibrobox_scagr7-2c_rlfddd	4182.86	3995	5240	2720.02	**2652**	2820
yeast	212.37	177	236	156.59	**144**	172
Graph	**METIS__HE Ave**	**Min**	**Max**	**BLP__HE Ave**	**Min**	**Max**
soc-Epinions1	2976.1	2818	3078	2305.18	**2266**	2344
vsp_befref_fxm_2_4_air02	1072.9	989	1142	298.22	283	340
vsp_finan512_scagr7-2c_rlfddd	7438.5	7216	7697	5026.48	4608	5330
vsp_model1_crew1_cr42_south31	2216.08	**2086**	2631	2346.6	2295	2407
vsp_sctap1-2b_and_seymourl	4114.48	3831	4373	3665.98	**3440**	3931
vsp_south31_slptsk	2054.39	**1982**	2116	2224.59	2168	2284
vsp_vibrobox_scagr7-2c_rlfddd	3467.55	3362	3613	2720.2	2646	2828
yeast	192.62	182	213	156.83	146	170

Table 6 Improvement of BLP over METIS when random matching was used

Category	% Wins	% Improvement Ave	Min	Max
M	22.22	−0.06	−0.60	0.86
C	100.00	2.88	0.00	4.94
OO	45.00	−0.10	−1.98	1.39
OC	100.00	2.60	0.50	8.25
Total	65.38	1.14	−1.98	8.25

Table 7 Improvement of BLP over METIS when heavy-edge matching was used

Category	% Wins	% Improvement		
		Ave	Min	Max
M	11.11	−0.09	−0.50	0.84
C	100.00	2.20	0.00	4.18
OO	30.00	−0.27	−3.13	1.27
OC	75.00	1.63	−0.43	5.49
Total	53.85	0.74	−3.13	5.49

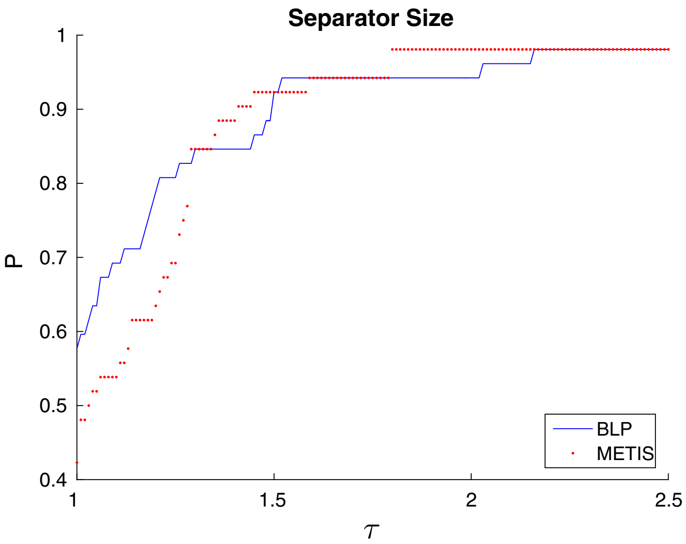

Fig. 2 Performance profile comparing BLP and METIS with their optimal matching schemes on all graphs

get a solution at iteration k. There are many potential enhancements of this nature that should provide a much faster running time for BLP.

In [35], the authors developed a semi-definite programming method for computing tight upper and lower bounds on the cardinality of an optimal vertex separator in a graph, where the cardinalities of the two shores are approximately the same. Tests were run on a set of 11 graphs having between 93 and 343 vertices. Table 8 gives the dimensions of the test graphs, the upper and lower bounds on the cardinality of an optimal separator, and the average, minimum, and maximum cardinality of the vertex separator found by BLP_HE across 100 random trials using $\ell_a = \ell_b = 1$ and $u_a = u_b = \lfloor 0.5n \rfloor$. The results for BLP_RM were similar. For all graphs, the minimum cardinality separator obtained by BLP was always less than or equal to the upper bound of [35]. The average cardinality was less than or equal to the upper bound in 5 out of the 11 instances. Due to slight differences in the way the bounds on the shores are enforced in the SDP method of [35], the BLP solution is below the lower bound in 3 of the instances.

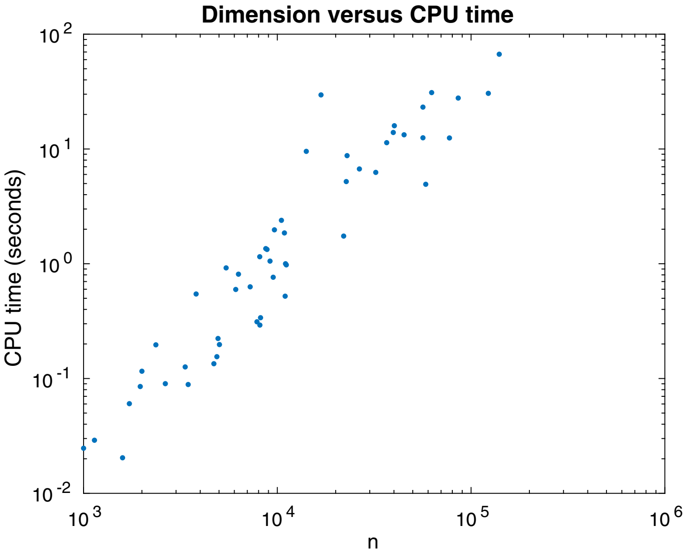

Fig. 3 Number of vertices n versus CPU time for BLP__RM

Table 8 Results for BLP__HE on test graphs from [35]

Graph	$\|\mathcal{V}\|$	$\|\mathcal{E}\|/2$	LB	UB	$\|S\|$ Ave	Min	Max
bcspwr03	118	179	4	5	5.53	4	9
can144	144	576	5	6	7.74	6	12
can161	161	608	17	18	16.95	16	25
can229	229	774	16	19	19.02	19	20
example1	93	470	11	11	11.01	10	15
grid3dt5	125	604	19	19	20.36	19	21
grid3dt6	216	1115	27	30	27.31	27	32
grid3dt7	343	1854	27	37	39.53	37	40
gridt15	120	315	9	11	11	11	11
gridt17	153	408	10	13	12.21	12	13
Smallmesh	136	354	6	6	5.32	5	19

7 Conclusion

We have developed a new algorithm (BLP) for solving large-scale instances of the Vertex Separator Problem (1). The algorithm incorporates a multilevel framework; that is, the original graph is coarsened several times; the problem is solved for the coarsest graph; and the solution to the coarse graph is gradually uncoarsened and refined to obtain a solution to the original graph. A key feature of the algorithm is the use of the continuous bilinear program (14) in both the solution and refinement phases. In the case where vertex weights are all equal to one (or a constant), (14) is an exact formulation of the VSP in the sense that there exists a binary solution satisfying (3), from which an optimal solution to the VSP can be recovered using (8). When vertex weights are not all equal, we showed that (14) still approximates the VSP in the sense that there exists a mostly binary solution.

During the solution and refinement phases of BLP, the bilinear program is solved approximately by applying the algorithm MCA_GR, a mountain climbing algorithm which incorporates a technique which we call γ-perturbations to escape from stationary points and explore more of the search space. The γ-perturbations improve the separator by relaxing the requirement that there are no edges between the sets in the partition. We determined the smallest relaxation that will generate a new partition. To our knowledge, this is the first multilevel algorithm to make use of a continuous optimization based refinement method for the family of graph partitioning problems. The numerical results of Sect. 6 indicate that BLP is capable of locating vertex separators of high quality (comparing against METIS); the algorithm is particularly effective on communication and collaboration networks, while being less effective on graphs arising in mesh applications.

8 Appendix

In this section, we analyze cases where MCA is guaranteed to strictly improve the current vertex separator. For any $k \in \mathbb{B}$ and any $\mathcal{Z} \subseteq \mathcal{V}$ we define the set $\overline{\mathcal{N}}_{\leq k}(\mathcal{Z})$ by

$$\overline{\mathcal{N}}_{\leq k}(\mathcal{Z}) := \{i \in \mathcal{V} \ : \ |\overline{\mathcal{N}}(i) \cap \mathcal{Z}| \leq k\} .$$

For simplicity, we denote $\overline{\mathcal{N}}_{\leq 0}(\mathcal{Z})$ by $\overline{\mathcal{N}}_0(\mathcal{Z}) = \mathcal{V} \backslash \overline{\mathcal{N}}(\mathcal{Z})$.

Observation 81 *Let* $\mathcal{A}, \mathcal{B} \subseteq \mathcal{V}$, $\mathbf{x} := \mathrm{supp}^{-1}(\mathcal{A})$, *and* $\mathbf{y} := \mathrm{supp}^{-1}(\mathcal{B})$. *Then,*

$$\mathbf{x}^{\mathsf{T}}(\mathbf{A} + \mathbf{I})\mathbf{y} \ = \ \textstyle\sum_{i \in \mathcal{B}} |\overline{\mathcal{N}}(i) \cap \mathcal{A}| \ = \ \textstyle\sum_{i \in \mathcal{A}} |\overline{\mathcal{N}}(i) \cap \mathcal{B}| \ .$$

Observation 82 *Let* $\mathcal{Z} \subseteq \mathcal{V}$. *Then*

$$\overline{\mathcal{N}}_0(\mathcal{Z}) \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{Z}) \ .$$

Observation 83 *Let* $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{V}$ *be such that* $\mathcal{X} \subseteq \mathcal{Y}$. *Then for any* $k \geq 0$,

$$\overline{\mathcal{N}}_{\leq k}(\mathcal{X}) \supseteq \overline{\mathcal{N}}_{\leq k}(\mathcal{Y}) \ .$$

Throughout this section, we consider the special case where $u_a, \ell_a, u_b, \ell_b \in \mathbb{Z}$, $c_i = \gamma > 0 \ \forall \ i \in \mathcal{V}$, and $\mathbf{w} = \mathbf{1}$, in which case the extreme points of (14) are binary (see [19, Lemma 3.3]).

Proposition 4 *Let* $\mathcal{A}, \mathcal{B} \subseteq \mathcal{V}$ *be such that* $\mathbf{x} := \mathrm{supp}^{-1}(\mathcal{A}) \in \mathcal{P}_a$ *and* $\mathbf{y} := \mathrm{supp}^{-1}(\mathcal{B}) \in \mathcal{P}_b$. *Suppose that* $|\overline{\mathcal{N}}_{\leq 1}(\mathcal{A})| \geq \ell_b$ *and let* $\mathbf{y}^* \in \mathcal{P}_b \cap \mathbb{B}^n$ *and* $\mathcal{B}^* := \mathrm{supp}(\mathbf{y}^*)$. *Then* $\mathbf{y}^*$ *is an optimal solution to the problem*

$$\max \{f(\mathbf{x}, \tilde{\mathbf{y}}) \ : \ \tilde{\mathbf{y}} \in \mathcal{P}_b\} \tag{23}$$

if and only if one of the following conditions holds:

(M1) $\overline{\mathcal{N}}_0(\mathcal{A}) \subseteq \mathcal{B}^* \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})$,
(M2) $\mathcal{B}^* \subset \overline{\mathcal{N}}_0(\mathcal{A})$ *and* $|\mathcal{B}^*| = u_b$.

Proof First of all, since $f(\mathbf{x}, \mathbf{y})$ is linear in $\mathbf{y}$ when $\mathbf{x}$ is fixed, (23) has an optimal solution lying at an extreme point of the set $\mathcal{P}_b$ (see, for instance Chapter 2 of [40]); that is, at a point $\mathbf{y}$ for which there are n linearly independent constraints in $\mathcal{P}_b$ which are active. If all n of these constraints are of the form $\mathbf{0} \leq \mathbf{y}$ or $\mathbf{y} \leq \mathbf{1}$, then $\mathbf{y}$ is binary. Otherwise, $\mathbf{y}$ has at least $n-1$ binary components and the constraint $\mathbf{1}^\mathsf{T}\mathbf{y}$ is active at either the lower or upper bound; that is, $\mathbf{1}^\mathsf{T}\mathbf{y} = u_b$ or $\mathbf{1}^\mathsf{T}\mathbf{y} = \ell_b$. But since $u_b, \ell_b \in \mathbb{Z}$ and $n-1$ components of $\mathbf{y}$ are binary, in fact we must have that every component of $\mathbf{y}$ is binary. Thus, (23) has an optimal solution which is binary.

Next, note that by computation we have that for any $\tilde{\mathbf{y}} \in \mathcal{P}_b \cap \mathbb{B}^n$

$$
\begin{aligned}
\frac{1}{\gamma} f(\mathbf{x}, \tilde{\mathbf{y}}) &= \frac{1}{\gamma}[\mathbf{c}^\mathsf{T}(\mathbf{x} + \tilde{\mathbf{y}}) - \gamma \mathbf{x}^\mathsf{T}(\mathbf{A} + \mathbf{I})\tilde{\mathbf{y}}] \\
&= \mathbf{1}^\mathsf{T}(\mathbf{x} + \tilde{\mathbf{y}}) - \mathbf{x}^\mathsf{T}(\mathbf{A} + \mathbf{I})\tilde{\mathbf{y}} && (24) \\
&= |\mathcal{A}| + |\tilde{\mathcal{B}}| - \sum_{i \in \tilde{\mathcal{B}}} |\overline{\mathcal{N}}(i) \cap \mathcal{A}| && (25) \\
&= |\mathcal{A}| + |\overline{\mathcal{N}}_0(\mathcal{A}) \cap \tilde{\mathcal{B}}| + |\tilde{\mathcal{B}} \backslash \overline{\mathcal{N}}_0(\mathcal{A})| - \sum_{i \in \tilde{\mathcal{B}} \backslash \overline{\mathcal{N}}_0(\mathcal{A})} |\overline{\mathcal{N}}(i) \cap \mathcal{A}| && (26) \\
&= |\mathcal{A}| + |\overline{\mathcal{N}}_0(\mathcal{A}) \cap \tilde{\mathcal{B}}| + \sum_{i \in \tilde{\mathcal{B}} \backslash \overline{\mathcal{N}}_0(\mathcal{A})} (1 - |\overline{\mathcal{N}}(i) \cap \mathcal{A}|) && (27) \\
&= |\mathcal{A}| + |\overline{\mathcal{N}}_0(\mathcal{A}) \cap \tilde{\mathcal{B}}| + \sum_{i \in \tilde{\mathcal{B}} \backslash \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})} (1 - |\overline{\mathcal{N}}(i) \cap \mathcal{A}|) && (28) \\
&\leq |\mathcal{A}| + |\overline{\mathcal{N}}_0(\mathcal{A}) \cap \tilde{\mathcal{B}}| \,, && (29)
\end{aligned}
$$

where $\tilde{\mathcal{B}} := \operatorname{supp}(\tilde{\mathbf{y}})$, (24) follows from the assumption on $\mathbf{c}$, (25) follows from Observation 81, (26) is due to the definition of $\overline{\mathcal{N}}_0(\mathcal{A})$, and (28) and (29) follow from the definition of $\overline{\mathcal{N}}_{\leq 1}(\mathcal{A})$ (which implies in particular that the terms in the final summation of (28) are negative, when they exist). The remainder of the proof is split into two cases.

Case 1 $|\overline{\mathcal{N}}_0(\mathcal{A})| > u_b$. In this case, first observe that there does not exist a point $\tilde{\mathbf{y}} \in \mathcal{P}_b \cap \mathbb{B}^n$ such that $\overline{\mathcal{N}}_0(\mathcal{A}) \subseteq \tilde{\mathcal{B}} \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})$, since otherwise $\mathbf{1}^\mathsf{T}\tilde{\mathbf{y}} = |\tilde{\mathcal{B}}| \geq |\overline{\mathcal{N}}_0(\mathcal{A})| > u_b$, contradicting $\tilde{\mathbf{y}} \in \mathcal{P}_b$. So, we claim that a point $\mathbf{y}^* \in \mathcal{P}_b \cap \mathbb{B}^n$ is an optimal solution to (23) if and only if (M2) holds. To see this, observe that if $\mathbf{y}^* \in \mathcal{P}_b \cap \mathbb{B}^n$ satisfies (M2) (with $\mathcal{B}^* = \operatorname{supp}(\mathbf{y}^*)$), then by (28) we have

$$
\begin{aligned}
\frac{1}{\gamma} f(\mathbf{x}, \mathbf{y}^*) &= |\mathcal{A}| + |\overline{\mathcal{N}}_0(\mathcal{A}) \cap \mathcal{B}^*| + \sum_{i \in \mathcal{B}^* \backslash \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})} (1 - |\overline{\mathcal{N}}(i) \cap \mathcal{A}|) \\
&= |\mathcal{A}| + |\overline{\mathcal{N}}_0(\mathcal{A}) \cap \mathcal{B}^*| && (30)
\end{aligned}
$$

$$= |\mathcal{A}| + |\mathcal{B}^*| \tag{31}$$
$$= |\mathcal{A}| + u_b \ , \tag{32}$$

where (30) and (31) follow from the fact that $\mathcal{B}^* \subset \overline{\mathcal{N}}_0(\mathcal{A}) \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})$ (by (M2)), and (32) follows from (M2). Next, if $\tilde{\mathbf{y}} \in \mathcal{P}_b \cap \mathbb{B}^n$ is a vector which does *not* satisfy (M2) (with $\mathcal{B}^*$ replaced by $\tilde{\mathcal{B}}$), observe that we cannot have $\tilde{\mathcal{B}} = \overline{\mathcal{N}}_0(\mathcal{A})$, since this case would fall under (M1), and we have already established that no points in $\mathcal{P}_b \cap \mathbb{B}^n$ satisfying (M1) exist. Hence, either $\tilde{\mathcal{B}} \backslash \overline{\mathcal{N}}_0(\mathcal{A}) \neq \emptyset$ or $|\tilde{\mathcal{B}}| < u_b$ (note that we cannot have $|\tilde{\mathcal{B}}| > u_b$, since this would contradict $\tilde{\mathbf{y}} = \text{supp}^{-1}(\tilde{\mathcal{B}}) \in \mathcal{P}_b$); and in both cases it follows that $|\overline{\mathcal{N}}_0(\mathcal{A}) \cap \tilde{\mathcal{B}}| < u_b$ (in the first case $|\overline{\mathcal{N}}_0(\mathcal{A}) \cap \tilde{\mathcal{B}}| < |\tilde{\mathcal{B}}| \leq u_b$ and in the second case $|\overline{\mathcal{N}}_0(\mathcal{A}) \cap \tilde{\mathcal{B}}| \leq |\tilde{\mathcal{B}}| < u_b$). Hence, by (29) and (32) we have

$$\frac{1}{\gamma} f(\mathbf{x}, \tilde{\mathbf{y}}) \leq |\mathcal{A}| + |\overline{\mathcal{N}}_0(\mathcal{A}) \cap \tilde{\mathcal{B}}| < |\mathcal{A}| + u_b = \frac{1}{\gamma} f(\mathbf{x}, \mathbf{y}^*) \ . \tag{33}$$

Hence, for any $\mathbf{y}^*, \tilde{\mathbf{y}} \in \mathcal{P}_b \cap \mathbb{B}^n$ such that $\mathbf{y}^*$ satisfies (M2) and $\tilde{\mathbf{y}}$ does not satisfy (M2), we have $f(\mathbf{x}, \tilde{\mathbf{y}}) < f(\mathbf{x}, \mathbf{y}^*) = (|\mathcal{A}| + u_b)\gamma$. Moreover, by our assumption $|\overline{\mathcal{N}}_0(\mathcal{A})| > u_b$, a point $\mathbf{y}^* \in \mathcal{P}_b \cap \mathbb{B}^n$ satisfying (M2) exists (let $\mathbf{y}^*$ by any binary point such that $\text{supp}(\mathbf{y}^*)$ is a subset of $\overline{\mathcal{N}}_0(\mathcal{A})$ of cardinality u_b). Hence, the binary optimal solutions to (23) are precisely the points $\mathbf{y}^* \in \mathcal{P}_b \cap \mathbb{B}^n$ satisfying (M2). This completes the proof of Case 1.

Case 2 $|\overline{\mathcal{N}}_0(\mathcal{A})| \leq u_b$. In this case, we first note that there does not exist a $\tilde{\mathbf{y}} \in \mathcal{P}_b \cap \mathbb{B}^n$ satisfying (M2), since otherwise $|\tilde{\mathcal{B}}| < |\overline{\mathcal{N}}_0(\mathcal{A})| \leq u_b = |\tilde{\mathcal{B}}|$, a contradiction (note that here we use the assumption that $|\overline{\mathcal{N}}_0(\mathcal{A})| \leq u_b$). So, we claim that a point $\mathbf{y}^* \in \mathcal{P}_b \cap \mathbb{B}^n$ is an optimal solution to (23) if and only if (M1) holds. To see this, observe that if $\mathbf{y}^*$ is a point in $\mathcal{P}_b \cap \mathbb{B}^n$ satisfying (M1), then we have from (28) that

$$\frac{1}{\gamma} f(\mathbf{x}, \mathbf{y}^*) = |\mathcal{A}| + |\overline{\mathcal{N}}_0(\mathcal{A}) \cap \mathcal{B}^*| + \sum_{i \in \mathcal{B}^* \backslash \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})} (1 - |\overline{\mathcal{N}}(i) \cap \mathcal{A}|) \tag{34}$$
$$= |\mathcal{A}| + |\overline{\mathcal{N}}_0(\mathcal{A}) \cap \mathcal{B}^*| \tag{35}$$
$$= |\mathcal{A}| + |\overline{\mathcal{N}}_0(\mathcal{A})| \ , \tag{36}$$

where (35) and (36) follow from the fact that $\mathcal{B}^* \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})$ and $\overline{\mathcal{N}}_0(\mathcal{A}) \subseteq \mathcal{B}^*$, respectively. Next, if $\tilde{\mathbf{y}} \in \mathcal{P}_b \cap \mathbb{B}^n$ is a point which does *not* satisfy (M1), then either $\overline{\mathcal{N}}_0(\mathcal{A}) \backslash \tilde{\mathcal{B}} \neq \emptyset$ or $\tilde{\mathcal{B}} \backslash \overline{\mathcal{N}}_{\leq 1}(\mathcal{A}) \neq \emptyset$. In the first case, we have $|\overline{\mathcal{N}}_0(\mathcal{A}) \cap \tilde{\mathcal{B}}| < |\overline{\mathcal{N}}_0(\mathcal{A})|$; in the second case, the inequality (29) holds strictly (since the terms appearing in the last summation in (28) are all negative) and we have trivially that $|\overline{\mathcal{N}}_0(\mathcal{A}) \cap \tilde{\mathcal{B}}| \leq |\overline{\mathcal{N}}_0(\mathcal{A})|$. And so in either case by (28) and (29) we have that

$$\frac{1}{\gamma} f(\mathbf{x}, \tilde{\mathbf{y}}) < |\mathcal{A}| + |\overline{\mathcal{N}}_0(\mathcal{A})| = \frac{1}{\gamma} f(\mathbf{x}, \mathbf{y}^*) \ . \tag{37}$$

Hence, for any $\mathbf{y}^*, \tilde{\mathbf{y}} \in \mathcal{P}_b \cap \mathbb{B}^n$ such that $\mathbf{y}^*$ satisfies (M1) and $\tilde{\mathbf{y}}$ does not satisfy (M1), we have that $f(\mathbf{x}, \tilde{\mathbf{y}}) < f(\mathbf{x}, \mathbf{y}^*) = \gamma(|\mathcal{A}| + |\overline{\mathcal{N}}_0(\mathcal{A})|)$. Moreover, since $|\overline{\mathcal{N}}_0(\mathcal{A})| \leq u_b$

(the assumption associated with Case 2) and $|\overline{\mathcal{N}}_{\le 1}(\mathcal{A})| \ge \ell_b$ (the supposition in the statement of the proposition), a point $\mathbf{y}^* \in \mathcal{P}_b \cap \mathbb{B}^n$ satisfying (M1) exists (construct $\text{supp}(\mathbf{y}^*)$, for example, by starting with $\overline{\mathcal{N}}_0(\mathcal{A})$ and gradually adding elements from $\overline{\mathcal{N}}_{\le 1}(\mathcal{A}) \backslash \overline{\mathcal{N}}_0(\mathcal{A})$ until the set has cardinality u_b). Hence, the binary optimal solutions to (23) are precisely the points $\mathbf{y}^* \in \mathcal{P}_b \cap \mathbb{B}^n$ satisfying (M1). This completes the proof. □

The following corollary, obtained by taking the contrapositive of Proposition 4, gives necessary and sufficient conditions for being able to improve upon a given separator by solving (23) (or the analogous program in $\mathbf{x}$).

Corollary 1 *Let $\mathcal{A}, \mathcal{B} \subseteq \mathcal{V}$ be such that $\mathbf{x} := \text{supp}^{-1}(\mathcal{A}) \in \mathcal{P}_a$ and $\mathbf{y} := \text{supp}^{-1}(\mathcal{B}) \in \mathcal{P}_b$.*

1. *Suppose that $|\overline{\mathcal{N}}_{\le 1}(\mathcal{A})| \ge \ell_b$. Then $\exists\, \hat{\mathbf{y}} \in \mathcal{P}_b$ such that*

$$f(\mathbf{x}, \hat{\mathbf{y}}) > f(\mathbf{x}, \mathbf{y})$$

   if and only if one of the following conditions holds:
   (J1) $|\overline{\mathcal{N}}_0(\mathcal{A}) \cap \mathcal{B}| < \min\{u_b, |\overline{\mathcal{N}}_0(\mathcal{A})|\}$,
   (J2) $\mathcal{B} \backslash \overline{\mathcal{N}}_{\le 1}(\mathcal{A}) \ne \emptyset$.
2. *Suppose that $|\overline{\mathcal{N}}_{\le 1}(\mathcal{B})| \ge \ell_a$. Then $\exists\, \hat{\mathbf{x}} \in \mathcal{P}_a$ such that*

$$f(\hat{\mathbf{x}}, \mathbf{y}) > f(\mathbf{x}, \mathbf{y})$$

   if and only if one of the following conditions holds:
   (L1) $|\overline{\mathcal{N}}_0(\mathcal{B}) \cap \mathcal{A}| < \min\{u_a, |\overline{\mathcal{N}}_0(\mathcal{B})|\}$,
   (L2) $\mathcal{A} \backslash \overline{\mathcal{N}}_{\le 1}(\mathcal{B}) \ne \emptyset$.

Proof We only prove Part 1, since Part 2 will follow by symmetry. By Proposition 4, we need only show that the negation of [(M1) or (M2)] (with $\mathcal{B}^*$ replaced by $\mathcal{B}$) holds if and only if [(J1) or (J2)] holds.

To see the forward direction, suppose that neither (M1) nor (M2) holds. Since (M1) does not hold, we have that either $\mathcal{B} \backslash \overline{\mathcal{N}}_{\le 1}(\mathcal{A}) \ne \emptyset$ or $\overline{\mathcal{N}}_0(\mathcal{A}) \backslash \mathcal{B} \ne \emptyset$. In the first case, (J2) holds and we are done. In the second case, we have

$$|\overline{\mathcal{N}}_0(\mathcal{A}) \cap \mathcal{B}| < |\overline{\mathcal{N}}_0(\mathcal{A})| \, . \tag{38}$$

Moreover, since (M2) does not hold, we have that either $\mathcal{B} \backslash \overline{\mathcal{N}}_0(\mathcal{A}) \ne \emptyset$ (note that we cannot have $\mathcal{B} = \overline{\mathcal{N}}_0(\mathcal{A})$, since this would imply that (M1) holds, a contradiction) or $|\mathcal{B}| < u_b$. In the first case, $|\overline{\mathcal{N}}_0(\mathcal{A}) \cap \mathcal{B}| < |\mathcal{B}| \le u_b$, while in the second case $|\overline{\mathcal{N}}_0(\mathcal{A}) \cap \mathcal{B}| \le |\mathcal{B}| < u_b$. Hence, in either case we have

$$|\overline{\mathcal{N}}_0(\mathcal{A}) \cap \mathcal{B}| < u_b \, . \tag{39}$$

Combining (38) with (39) we have that (J1) holds. This completes the proof of the forward direction.

To prove the backwards direction, we will prove the contrapositive; that is, we will prove that if either (M1) or (M2) holds, then neither (J1) nor (J2) holds. First, suppose that (M1) holds. Then, $|\overline{\mathcal{N}}_0(\mathcal{A}) \cap \mathcal{B}| = |\overline{\mathcal{N}}_0(\mathcal{A})| \geq \min\{u_b, |\overline{\mathcal{N}}_0(\mathcal{A})|\}$, and so (J1) does not hold. Moreover, since by (M1) $\mathcal{B} \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})$, we have that $\mathcal{B}\backslash\overline{\mathcal{N}}_{\leq 1}(\mathcal{A}) = \emptyset$, implying that (J2) does not hold. Next, suppose instead that (M2) holds. Then, $|\overline{\mathcal{N}}_0(\mathcal{A}) \cap \mathcal{B}| = |\mathcal{B}| = u_b \geq \min\{u_b, |\overline{\mathcal{N}}_0(\mathcal{A})|\}$, implying that (J1) does not hold. To see that (J2) does not hold, we need only observe that $\mathcal{B} \subset \overline{\mathcal{N}}_0(\mathcal{A}) \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})$, and so $\mathcal{B}\backslash\overline{\mathcal{N}}_{\leq 1}(\mathcal{A}) = \emptyset$. This completes the proof of the backwards direction. □

Corollary 1 may be applied recursively (first to $\mathcal{B}$, then to $\mathcal{A}$, etc.) in order to develop necessary and sufficient conditions for improvement between any two consecutive iterates in an entire sequence $\{(\mathcal{A}^k, \mathcal{B}^k)\}_{k\geq 0}$ (where $\mathcal{A}^k = \mathcal{A}^{k-1}$ when $k \geq 1$ is odd and $\mathcal{B}^k = \mathcal{B}^{k-1}$ when $k \geq 2$ is even), all stated in terms of the initial sets $\mathcal{A}^0 = \mathcal{A}$ and $\mathcal{B}^0 = \mathcal{B}$. In the next proposition, we employ Corollary 1 to derive sufficient conditions for improvement after two iterations of this process. To save space, we only state the conditions for the case where $\mathcal{B}$ is refined first. It is clear that by symmetry, an analogous set of conditions applies to the case where $\mathcal{A}$ is refined first.

Proposition 5 *Let $\mathcal{A}, \mathcal{B} \subseteq \mathcal{V}$ be such that $\mathbf{x} := \mathrm{supp}^{-1}(\mathcal{A}) \in \mathcal{P}_a$ and $\mathbf{y} := \mathrm{supp}^{-1}(\mathcal{B}) \in \mathcal{P}_b$. Suppose that $|\overline{\mathcal{N}}_{\leq 1}(\mathcal{A})| \geq \ell_b$, $|\overline{\mathcal{N}}_{\leq 1}(\overline{\mathcal{N}}_{\leq 1}(\mathcal{A}))| \geq \ell_a$, and that one of the following conditions holds:*

(J3) $|\mathcal{A}| < u_a$, $|\overline{\mathcal{N}}_0(\mathcal{A})| \leq u_b$, *and*

$$\exists\, i \in \mathcal{V}\backslash\mathcal{A} \;\text{ such that }\; \overline{\mathcal{N}}(i) \subseteq \overline{\mathcal{N}}(\mathcal{A}) \;\text{ and }\; |\overline{\mathcal{N}}_{\leq 1}(\mathcal{A})\backslash\overline{\mathcal{N}}(i)| \geq \ell_b\ ,$$

(J4) $\ell_b \leq |\overline{\mathcal{N}}_0(\mathcal{A})| + 2 \leq u_b$ *and*

$$\exists\, i \in \mathcal{A},\; j \in \mathcal{V}\backslash\mathcal{A} \;\text{ such that }\; \mathcal{N}(i) \cap \mathcal{A} = \emptyset \;\text{ and }\; \mathcal{N}(j) \cap \mathcal{A} = \{i\}\ ,$$

(J5) $\ell_b \leq |\overline{\mathcal{N}}_0(\mathcal{A})| + 2 \leq u_b$ *and*

$$\exists\, i \in \mathcal{A},\; j \neq k \in \mathcal{V}\backslash\mathcal{A} \;\text{ such that }\; \mathcal{N}(j) \cap \mathcal{A} = \mathcal{N}(k) \cap \mathcal{A} = \{i\}\ .$$

Then $\exists\, \hat{\mathbf{y}} \in \mathcal{P}_b \cap \mathbb{B}^n$ such that for some $\hat{\mathbf{x}} \in \mathcal{P}_a$

$$f(\hat{\mathbf{x}}, \hat{\mathbf{y}}) > f(\mathbf{x}, \hat{\mathbf{y}}) = \max\{f(\mathbf{x}, \tilde{\mathbf{y}}) \;:\; \tilde{\mathbf{y}} \in \mathcal{P}_b\} \geq f(\mathbf{x}, \mathbf{y})\ .$$

Proof We split the proof into three cases, depending on the particular condition which holds.

Case (J3) Suppose that (J3) holds. By Observation 82, $\overline{\mathcal{N}}_0(\mathcal{A}) \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})$ and by (J3) we have $\overline{\mathcal{N}}_0(\mathcal{A}) = \mathcal{V}\backslash\overline{\mathcal{N}}(\mathcal{A}) \subseteq \mathcal{V}\backslash\overline{\mathcal{N}}(i)$. Hence,

$$\overline{\mathcal{N}}_0(\mathcal{A}) \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})\backslash\overline{\mathcal{N}}(i)\ . \tag{40}$$

Moreover, (J3) states that

$$|\overline{\mathcal{N}}_0(\mathcal{A})| \leq u_b \quad \text{and} \quad |\overline{\mathcal{N}}_{\leq 1}(\mathcal{A})\backslash\overline{\mathcal{N}}(i)| \geq \ell_b\ . \tag{41}$$

Hence, combining (40) and (41), $\exists\, \hat{\mathcal{B}} \subseteq \mathcal{V}$ such that

$$\overline{\mathcal{N}}_0(\mathcal{A}) \subseteq \hat{\mathcal{B}} \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A}) \backslash \overline{\mathcal{N}}(i) \quad \text{and} \quad \ell_b \leq |\hat{\mathcal{B}}| \leq u_b \ . \tag{42}$$

By (42) the point $\hat{\mathbf{y}} := \operatorname{supp}^{-1}(\hat{\mathcal{B}})$ lies in $\mathcal{P}_b \cap \mathbb{B}^n$ and $\hat{\mathcal{B}}$ satisfies (M1). Hence, Proposition 4 implies that $\hat{\mathbf{y}}$ is optimal in (23), and thus

$$f(\mathbf{x}, \hat{\mathbf{y}}) \geq f(\mathbf{x}, \mathbf{y}) \ , \tag{43}$$

which proves the first half of the statement in the proposition.

Next, observe that since by (J3) we have $i \in \mathcal{V} \backslash \mathcal{A}$ and since by (42) we have $\overline{\mathcal{N}}(i) \cap \hat{\mathcal{B}} = \emptyset$, it follows that $i \in \overline{\mathcal{N}}_0(\hat{\mathcal{B}}) \backslash \mathcal{A}$, and hence $\overline{\mathcal{N}}_0(\hat{\mathcal{B}}) \cap \mathcal{A} \subset \overline{\mathcal{N}}_0(\hat{\mathcal{B}})$, implying $|\overline{\mathcal{N}}_0(\hat{\mathcal{B}}) \cap \mathcal{A}| < |\overline{\mathcal{N}}_0(\hat{\mathcal{B}})|$. In addition, $|\overline{\mathcal{N}}_0(\hat{\mathcal{B}}) \cap \mathcal{A}| \leq |\mathcal{A}| < u_a$, by (J3). Hence,

$$|\overline{\mathcal{N}}_0(\hat{\mathcal{B}}) \cap \mathcal{A}| < \min\{u_a, |\overline{\mathcal{N}}_0(\mathcal{B})|\} \ ;$$

that is, $\hat{\mathcal{B}}$ satisfies (L1) (with $\mathcal{B}$ replaced by $\hat{\mathcal{B}}$). And since by (42) $\hat{\mathcal{B}} \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})$ we have from Observation 81 that $\overline{\mathcal{N}}_{\leq 1}(\hat{\mathcal{B}}) \supseteq \overline{\mathcal{N}}_{\leq 1}(\overline{\mathcal{N}}_{\leq 1}(\mathcal{A}))$, and hence $|\overline{\mathcal{N}}_{\leq 1}(\hat{\mathcal{B}})| \geq |\overline{\mathcal{N}}_{\leq 1}(\overline{\mathcal{N}}_{\leq 1}(\mathcal{A}))| \geq \ell_a$, where the final inequality here follows from the supposition in the statement of the proposition. So, $\hat{\mathcal{B}}$ satisfies the assumptions of Part 2 of Corollary 1, in addition to (L1). Hence, $\exists\, \hat{\mathbf{x}} \in \mathcal{P}_a$ such that $f(\hat{\mathbf{x}}, \hat{\mathbf{y}}) > f(\mathbf{x}, \hat{\mathbf{y}})$. This completes the proof in the case where (J3) holds.

Case (J4) Suppose that (J4) holds. First, note that by the assumption on i and j, we have

$$\overline{\mathcal{N}}(i) \cap \mathcal{A} = \overline{\mathcal{N}}(j) \cap \mathcal{A} = \{i\} \ .$$

Hence, $|\overline{\mathcal{N}}(i) \cap \mathcal{A}| = |\overline{\mathcal{N}}(j) \cap \mathcal{A}| = 1$; that is, $\{i, j\} \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A}) \backslash \overline{\mathcal{N}}_0(\mathcal{A})$. Hence, letting

$$\hat{\mathcal{B}} := \overline{\mathcal{N}}_0(\mathcal{A}) \cup \{i, j\} \ ,$$

we have

$$\overline{\mathcal{N}}_0(\mathcal{A}) \subseteq \hat{\mathcal{B}} \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A}) \ , \tag{44}$$

so that $\hat{\mathcal{B}}$ satisfies (M1) (with $\mathcal{B}^*$ replaced by $\hat{\mathcal{B}}$). Moreover, since $|\hat{\mathcal{B}}| = |\overline{\mathcal{N}}_0(\mathcal{A})| + 2$, by (J4) we have that

$$\ell_b \leq |\hat{\mathcal{B}}| = |\overline{\mathcal{N}}_0(\mathcal{A})| + 2 \leq u_b \ . \tag{45}$$

So by (45) we have $\hat{\mathbf{y}} := \operatorname{supp}^{-1}(\hat{\mathcal{B}}) \in \mathcal{P}_b$ and by (44) $\hat{\mathcal{B}}$ satisfies (M1). Hence, by Proposition 4, $\hat{\mathbf{y}}$ is an optimal solution to (23) and therefore satisfies (43). This proves the first half of the statement in the proposition.

Next, observe that $\overline{\mathcal{N}}(i) \cap \hat{\mathcal{B}} \supseteq \{i, j\}$, so that $|\overline{\mathcal{N}}(i) \cap \hat{\mathcal{B}}| \geq 2$; that is, $i \in \mathcal{A} \backslash \overline{\mathcal{N}}_{\leq 1}(\hat{\mathcal{B}})$, implying that $\hat{\mathcal{B}}$ satisfies (L2) (with $\mathcal{B}$ replaced by $\hat{\mathcal{B}}$). And since $\hat{\mathcal{B}} \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})$ by (44), Observation 81 again implies that $\overline{\mathcal{N}}_{\leq 1}(\hat{\mathcal{B}}) \supseteq \overline{\mathcal{N}}_{\leq 1}(\overline{\mathcal{N}}_{\leq 1}(\mathcal{A}))$, so that $|\overline{\mathcal{N}}_{\leq 1}(\hat{\mathcal{B}})| \geq$

$|\overline{\mathcal{N}}_{\leq 1}(\overline{\mathcal{N}}_{\leq 1}(\mathcal{A}))| \geq \ell_a$. Thus, $\hat{\mathcal{B}}$ satisfies the assumptions of Part 2 of Corollary 1, in addition to (L2). Therefore $\exists\ \hat{\mathbf{x}} \in \mathcal{P}_a$ such that $f(\hat{\mathbf{x}}, \hat{\mathbf{y}}) > f(\mathbf{x}, \hat{\mathbf{y}})$. This completes the proof in the case where (J4) holds.

Case (J5) Suppose that (J5) holds. First, note that by the assumption on i, j, and k we have

$$\overline{\mathcal{N}}(j) \cap \mathcal{A} = \overline{\mathcal{N}}(k) \cap \mathcal{A} = \{i\}\ .$$

Hence, $|\overline{\mathcal{N}}(j) \cap \mathcal{A}| = |\overline{\mathcal{N}}(k) \cap \mathcal{A}| = 1$; that is, $\{j, k\} \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A}) \backslash \overline{\mathcal{N}}_0(\mathcal{A})$. Letting,

$$\hat{\mathcal{B}} := \overline{\mathcal{N}}_0(\mathcal{A}) \cup \{j, k\}\ ,$$

the proof that $\hat{\mathbf{y}} := \text{supp}^{-1}(\hat{\mathcal{B}})$ lies in $\mathcal{P}_b \cap \mathbb{B}^n$ and satisfies (43) is identical to the proof used in Case (J4).

Next, observe that $\overline{\mathcal{N}}(i) \cap \hat{\mathcal{B}} \supseteq \{j, k\}$, so that $|\overline{\mathcal{N}}(i) \cap \hat{\mathcal{B}}| \geq 2$; that is, $i \in \mathcal{A} \backslash \overline{\mathcal{N}}_{\leq 1}(\hat{\mathcal{B}})$, implying that $\hat{\mathcal{B}}$ satisfies (L2) (with $\mathcal{B}$ replaced by $\hat{\mathcal{B}}$). The remainder of the proof is identical to Case (J4). □

Remark 2 It can be shown that whenever the assumptions of (J4) or (J5) hold, any point $\overline{\mathbf{y}} \in \mathcal{P}_b \cap \mathbb{B}^n$ satisfying $\text{supp}(\hat{\mathbf{y}}) \subseteq \text{supp}(\overline{\mathbf{y}}) \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})$ will also satisfy

$$f(\hat{\mathbf{x}}, \overline{\mathbf{y}}) > f(\mathbf{x}, \overline{\mathbf{y}}) \geq f(\mathbf{x}, \mathbf{y})$$

for some $\hat{\mathbf{x}} \in \mathcal{P}_a$. Hence, in order to take advantage of improvement opportunities of the form (J4) or (J5), it is sufficient to take $\hat{\mathbf{y}} \in \mathcal{P}_b \cap \mathbb{B}^n$ such that $\hat{\mathcal{B}} = \text{supp}(\hat{\mathbf{y}})$ is maximal while satisfying $\{i, j\} \cup \overline{\mathcal{N}}_0(\mathcal{A}) \subseteq \hat{\mathcal{B}} \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})$ (in the case of (J4)) or $\{j, k\} \cup \overline{\mathcal{N}}_0(\mathcal{A}) \subseteq \hat{\mathcal{B}} \subseteq \overline{\mathcal{N}}_{\leq 1}(\mathcal{A})$ (in the case of (J5)). Since in practice there may be many vertices i, j, and k satisfying either (J4) or (J5) for a given partition, a pair of vectors $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ improving the partition is often not difficult to find.

References

1. Acer, S., Kayaaslan, E., Aykanat, C.: A recursive bipartitioning algorithm for permuting sparse square matrices into block diagonal form with overlap. SIAM J. Sci. Comput. **35**(1), C99–C121 (2013). doi:10.1137/120861242
2. Ashcraft, C.C., Liu, J.W.H.: A partition improvement algorithm for generalized nested dissection. Technical Report BCSTECH-94-020, Boeing Computer Services, Seattle, WA (1994)
3. Benlic, U., Hao, J.: Breakout local search for the vertex separator problem. In: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence (2013)
4. Brandt, A., Ron, D.: Chapter 1: Multigrid solvers and multilevel optimization strategies. In: Cong, J., Shinnerl, J.R. (eds.) Multilevel Optimization and VLSICAD. Kluwer, Alphen (2003)
5. Bui, T., Jones, C.: Finding good approximate vertex and edge partitions is NP-hard. Inf. Process. Lett. **42**, 153–159 (1992)
6. Buluç, A., Meyerhenke, H., Safro, I., Sanders, P., Schulz, C.: Recent advances in graph partitioning. In: Kliemann, L., Sanders, P. (eds.) Algorithm Engineering: Selected Results and Surveys, pp. 117–158. Springer, Berlin (2016)
7. Burmester, M., Desmedt, Y., Wang, Y.: Using approximation hardness to achieve dependable computation. In: Luby, M., Rolim, J., Serna, M. (eds.) Randomization and Approximation Techniques in

Computer Science, vol. 1518, pp. 172–186. Springer, Berlin (1998). doi:10.1007/3-540-49543-6_15. Lecture Notes in Computer Science
8. Chen, J., Safro, I.: Algebraic distance on graphs. SIAM J. Sci. Comput. **33**, 3468–3490 (2011)
9. Davis, T.A.: Direct Methods for Sparse Linear Systems. SIAM, Philadelphia (2006)
10. Davis, T.A.: University of Florida sparse matrix collection. ACM Trans. Math. Softw. **38**, 1–25 (2011)
11. Davis, T.A., Hager, W.W., Hungerford, J.T.: The separable convex quadratic knapsack problem. ACM Trans. Math. Softw. **42**, 1–25 (2016)
12. Evrendilek, C.: Vertex separators for partitioning a graph. Sensors **8**, 635–657 (2008)
13. Feige, U., Hajiaghayi, M., Lee, J.: Improved approximation algorithms for vertex separators. SIAM J. Comput. **38**, 629–657 (2008)
14. Fiduccia, C.M., Mattheyses, R.M.: A linear-time heuristic for improving network partitions. In: Proceedings of 19th Design Automation Conference, pp. 175–181. Las Vegas, NV (1982)
15. Fukuyama, J.: NP-completeness of the planar separator problems. J. Graph Algorithms Appl. **10**(2), 317–328 (2006)
16. George, A., Liu, J.W.H.: Computer Solution of Large Sparse Positive Definite Systems. Prentice-Hall, Englewood Cliffs (1981)
17. Gilbert, J.R., Zmijewski, E.: A parallel graph partitioning algorithm for a message-passing multiprocessor. Intl. J. Parallel Program. **16**, 498–513 (1987)
18. Hager, W.W., Hungerford, J.T.: Optimality conditions for maximizing a function over a polyhedron. Math. Program. **145**, 179–198 (2014)
19. Hager, W.W., Hungerford, J.T.: Continuous quadratic programming formulations of optimization problems on graphs. Eur. J. Oper. Res. **240**, 328–337 (2015)
20. Hendrickson, B., Leland, R.: A multilevel algorithm for partitioning graphs. In: Proceedings of Supercomputing '95. IEEE (1995)
21. Hendrickson, B., Rothberg, E.: Effective sparse matrix ordering: just around the bend. In: Proceedings of 8th SIAM Conference on Parallel Processing for Scientific Computing (1997)
22. http://snap.stanford.edu/data/
23. Karypis, G., Kumar, V.: METIS: unstructured graph partitioning and sparse matrix ordering system. Technical Report, Department of Computer Science, University of Minnesota (1995)
24. Karypis, G., Kumar, V.: A fast and high quality multilevel scheme for partitioning irregular graphs. SIAM J. Sci. Comput. **20**, 359–392 (1998)
25. Kayaaslan, E., Pinar, A., Catalyürek, U., Aykanat, C.: Partitioning hypergraphs in scientific computing applications through vertex separators. SIAM J. Sci. Comput. **34**(2), A970–A992 (2012)
26. Kernighan, B.W., Lin, S.: An efficient heuristic procedure for partitioning graphs. Bell Syst. Tech. J. **49**, 291–307 (1970)
27. Kim, D., Frye, G.R., Kwon, S.S., Chang, H.J., Tokuta, A.O.: On combinatoric approach to circumvent internet censorship using decoy routers. In: Military Communications Conference, 2013. MILCOM 2013., pp. 1–6. IEEE (2013)
28. Konno, H.: A cutting plane algorithm for solving bilinear programs. Math. Program. **11**, 14–27 (1976)
29. Leiserson, C., Lewis, J.: Orderings for parallel sparse symmetric factorization. In: Third SIAM Conference on Parallel Processing for Scientific Computing, pp. 27–31. SIAM Publications (1987)
30. Leiserson, C.: Area-efficient graph layout (for VLSI). In: Proceedings of 21st Annual Symposium on the Foundations of Computer Science, pp. 270–281. IEEE (1980)
31. Litsas, C., Pagourtzis, A., Panagiotakos, G., Sakavalas, D.: On the resilience and uniqueness of CPA for secure broadcast. In: IACR Cryptology ePrint Archive (2013)
32. Miller, G., Teng, S.H., Thurston, W., Vavasis, S.: Geometric separators for finite element meshes. SIAM J. Sci. Comput. **19**, 364–386 (1998)
33. Nocedal, J., Wright, S.J.: Numerical Optimization, 2nd edn. Springer, New York (2006)
34. Pothen, A., Simon, H.D., Liou, K.: Partitioning sparse matrices with eigenvectors of graphs. SIAM J. Matrix Anal. Appl. **11**(3), 430–452 (1990)
35. Rendl, F., Sotirov, R.: The min-cut and vertex separator problem. Comput. Optim. Appl. (2017). doi:10.1007/s10589-017-9943-4
36. Ron, D., Safro, I., Brandt, A.: Relaxation-based coarsening and multiscale graph organization. Multiscale Model. Simul. **9**(1), 407–423 (2011)
37. Safro, I., Sanders, P., Schulz, C.: Advanced coarsening schemes for graph partitioning. ACM J. Exp. Algorithm. **19**(2), Article 2.2 (2014)

38. Safro, I., Ron, D., Brandt, A.: Graph minimum linear arrangement by multilevel weighted edge contractions. J. Algorithms **60**, 24–41 (2006)
39. Ullman, J.: Computational Aspects of VLSI. Computer Science Press, Rockville (1984)
40. Vanderbei, R.J.: Linear Programming: Foundations and Extensions, 4th edn. Springer, Berlin (2014)