Stabilizing network bargaining games by blocking players

Sara Ahmadian[1], Hamideh Hosseinzadeh[2], and Laura Sanità[1]

[1] Dept. of Combinatorics and Optimization, Univ. of Waterloo, Waterloo, Canada,
sahmadian@uwaterloo.ca, lsanita@uwaterloo.ca,
[2] Faculty of mathematical science, Alzahra University, Tehran, Iran.,
hamideh.hosseinzadeh@gmail.com.

Abstract. Cooperative matching games (Shapley and Shubik) and Network bargaining games (Kleinberg and Tardos) are games described by an undirected graph, where the vertices represent players. An important role in such games is played by *stable* graphs, that are graphs whose set of inessential vertices (those that are exposed by at least one maximum matching) are pairwise non adjacent. In fact, stable graphs characterize instances of such games that admit the existence of stable outcomes.
In this paper, we focus on stabilizing instances of the above games by *blocking* as few players as possible. Formally, given a graph G we want to find a minimum cardinality set of vertices such that its removal from G yields a stable graph. We give a combinatorial polynomial-time algorithm for this problem, and develop approximation algorithms for some NP-hard *weighted* variants, where each vertex has an associated non-negative weight. Our approximation algorithms are LP-based, and we show that our analysis are almost tight by giving suitable lower bounds on the integrality gap of the used LP relaxations.

1 Introduction

Game theory is an active and important area of research in the field of Theoretical Computer Science, and combinatorial optimization techniques are often crucially employed in solving game theory problems [15]. For several games defined on networks, studying the structure of the underlying graph that describes the network setting is important to identify the existence of good outcomes for the corresponding games. Prominent examples are *cooperative matching games* introduced by Shapley and Shubik [17] and *network bargaining games* studied by Kleinberg and Tardos [9]. These are games described by an undirected graph $G = (V, E)$, where the vertices represent players, and the cardinality of a maximum matching in G, denoted by $\nu(G)$, represents a total *value* that the players could gain by interacting with each other.

In an instance of a cooperative matching game [17], one seeks for an *allocation* of the value $\nu(G)$ among players, described by a vector $y \in \mathbb{R}^V_{\geq 0}$, in which no subset of players S has an incentive to form a coalition to deviate. This is formally described by the constraint $\sum_{v \in S} y_v \geq \nu(G[S])$ for all subsets S, where $G[S]$ denotes the subgraph induced by the vertices in S. Such allocation y is called *stable*. It is well-known (see e.g. [5]) that cooperative matching game instances that admit the existence of a stable allocation are precisely the set of instances described by *stable graphs*: these are graphs whose set of *inessential vertices* are pairwise non adjacent. We recall here that a vertex v of a graph G is called inessential if there exists at least one maximum matching M in G that *exposes* v, that is, v is not an endpoint of M, and it is called essential otherwise (see Fig. 1 in Appendix A for an example).

Network bargaining games described by Kleinberg and Tardos [9] are network extensions of the classical Nash bargaining games [14]. In an instance of a network bargaining game described by a graph G, the edges represent a set of potential *deals* of unit value that the players (vertices) could make. An outcome of the game is given by a matching M of G (representing the set of deals that the players made) together with a value allocation $y \in \mathbb{R}^V_{\geq 0}$ on each vertex (representing how the players decided to split the values of the deals they made, if any). Kleinberg and Tardos [9] introduced the notion of *stable* outcomes for such games, that are outcomes where no player has an incentive to deviate, as well as the notion of *balanced* outcomes, that are stable outcomes in which, in addition, the values are "fairly" split among the players. The authors proved that a balanced outcome exists if and only if a stable outcome exists, and this happens if and only if the graph G describing the instance is *stable*.

Since not all graphs are stable, there are instances of both network bargaining games and cooperative matching games that do not admit stable solutions. This motivated many authors in past years to address the algorithmic problem of *stabilizing* such instances by minimally modifying the underlying graph. Two very natural ways to modify a graph in order to achieve some desired properties are via *edge-removal* or *vertex-removal* operations. The authors in [4] looked at edge-removal operations, that is, stabilizing instances of the above games by blocking potential deals that the players could make. In this paper, we look at the vertex-removal counterpart, that is, stabilizing instances by *blocking players*. Formally, this translates into the following problem:

Vertex-stabilizer problem: *Given a graph $G = (V, E)$, find a minimum cardinality* vertex-stabilizer*, that is a set $S \subseteq V$ whose removal from G yields a stable graph.*

We also generalize and study this problem in the *weighted* setting (a formal definition is in next subsection).

In addition to the connection with game theory, the vertex-stabilizer problem is also of interest from a combinatorial optimization perspective. In fact, an alternative and equivalent characterization of stable graphs can be given using linear programming and the notion of *fractional* matchings and vertex covers, as we are now going to explain. For a graph $G = (V, E)$, a fractional matching is a feasible solution to the LP:

$$\nu_f(G) := \max\Big\{\sum_{e\in E} x_e : \sum_{e\in\delta(v)} x_e \leq 1\ \forall v \in V,\ x \geq 0\Big\},$$

where $\delta(v)$ denotes the set of edges incident into v. Note that, if we add binary constraints to the above LP we obtain a formulation to find a matching of G of maximum cardinality $\nu(G)$. The dual of the above LP is:

$$\tau_f(G) := \min\Big\{\sum_{v\in V} y_v : y_u + y_v \geq 1\ \forall \{u, v\} \in E,\ y \geq 0\Big\}.$$

Once again, note that if we add binary constraints to this dual LP we obtain the canonical formulation for finding a *vertex cover* of G of minimum cardinality $\tau(G)$, that is, a min-cardinality subset of vertices covering all edges of the graph. For this reason, fractional feasible solutions to the above dual LP are called *fractional* vertex covers.

By duality theory, we know that the following holds: $\nu(G) \leq \nu_f(G) = \tau_f(G) \leq \tau(G)$. In general, there are graphs for which all the above inequalities are strict (e.g. a triangle). However, for certain classes of graphs some of the above inequalities hold tight. In particular, the class of *König-Egerváry* graphs ([18,11,12]) is formed by all graphs G for which $\nu(G) = \tau(G)$, that is, all the above inequalities hold tight. Note that the class of König-Egerváry graphs is a proper superset of the class of bipartite graphs. It is known (see e.g. [9]) that stable graphs are exactly the class of graphs for which $\nu(G) = \nu_f(G) = \tau_f(G) \leq \tau(G)$, that is, graphs for which the cardinality of a maximum matching ($\nu(G)$) is equal to the minimum size of a *fractional* vertex cover ($\tau_f(G)$). We have therefore the following relation:

$$\text{(Bipartite graphs)} \subsetneq \text{(König-Egerváry)} \subsetneq \text{(Stable graphs)} \subsetneq \text{(General graphs)}.$$

The algorithmic problems of turning a general graph into a bipartite one by removing either a set of edges or a set of vertices of minimum weight/cardinality, have been studied in the literature (see e.g. [1,8]). Similarly, the algorithmic problems of turning a given graph into a König-Egerváry one by removing a min-cardinality subset of edges or of vertices have been studied (see e.g [13]). Differently, as mentioned before, for stable graphs only the edge-removal question has been investigated so far, and this yields an additional motivation to study the vertex-removal question in this paper, both in the unweighted and in the weighted setting.

Our results and techniques. We study the vertex-stabilizer problem in Section 2. We first show a structural property of any minimal vertex-stabilizer. Namely, we prove that removing any minimal vertex-stabilizer *does not decrease* the size of a maximum matching in the resulting graph (Theorem 1). This theorem has an interesting interpretation in network bargaining and cooperative matching games: it states that it is always possible to stabilize instances by blocking a minimum number of players *without* decreasing the total value that the players could get. An analogue of Theorem 1 has been proven by Bock et al. [4] for minimal *edge*-stabilizers[3], however, their proof does not hold for the vertex-removal setting, and therefore our proof is different. Interestingly, despite this analogy, algorithmically the two problems appear to have a different complexity: while finding a min-cardinality edge-stabilizer is at least as hard as finding a minimum vertex cover [4], we here prove (Theorem 2) that finding a min-cardinality vertex-stabilizer is a polynomial-time solvable problem. In addition, we can prove (Theorem 6 in Appendix D) that the problem of blocking as few players as possible in order to make a *given* set of deals realizable as a stable outcome is also polynomial-time solvable, once again in contrast with the edge-removal setting, where the analogous question has been studied by [4] and shown to be vertex cover-hard. These three theorems are proved using combinatorial techniques. Theorem 1 exploits the structure of maximum matchings in graphs, that follows from the seminal works in [7,2]. Using Theorem 1, one can compute a lower bound on the size of a minimum vertex-stabilizer (as is done in [4]) exploiting properties of the so-called *Edmonds-Gallai Decomposition* (EGD) of a graph (definition is in Section 2). By further exploiting the relation that interplays between matchings and EGD, we get algorithms that prove Theorems 2 and 6.

[3] These are subsets of *edges* such that their removal from G yields a stable graph.

We study in Section 3 the *weighted* setting. In the vertex-stabilizer problem described before, players are all equally considered, that is, from an objective function perspective, we are assuming that blocking a player u is as costly as blocking a player v, independently on how u and v are connected to the rest of the players in the network. However, from a bargaining perspective, players might not all be equally powerful: as an example, players corresponding to essential vertices have more bargaining power than inessential ones. Moreover, blocking a player that is highly connected in the graph and therefore have the potential to enter in many deals might be more costly than blocking a less connected player. For this reason, blocking different players might have different costs. We can model this by assigning a *weight* $w_v \geq 0$ to each vertex v. In this setting, we could be interested in either *minimizing* the weight of the *blocked* players, or in *maximizing* the weight of the *remaining* players. Two optimization problems then arise:

Min-weight vertex-stabilizer: *Given a graph* $G = (V, E)$*, and vertex weights* $w_v \geq 0\ \forall v \in V$*, find a vertex-stabilizer* S *that minimizes* $w(S) = \sum_{v \in S} w_v$.

Max-weight vertex-stabilizer: *Given a graph* $G = (V, E)$*, and vertex weights* $w_v \geq 0\ \forall v \in V$*, find a vertex-stabilizer* S *that maximizes* $w(V \setminus S) = \sum_{v \notin S} w_v$.

This weighted setting poses more algorithmic challenges, and this is technically the most interesting part of the paper. We prove that both the above problems become NP-hard already if 2 different weights are involved (Theorem 7 in Appendix E). For this reason, we focus on *approximation algorithms*. We give a 2-approximation algorithm for the max-weight vertex-stabilizer problem (Theorem 3), and a $O(\gamma)$-approximation algorithm for the min-weight vertex-stabilizer problem, (Theorem 4), where γ is the size of the so-called *Tutte-set* of the graph G (a formal definition is in Section 2). Both our algorithms are LP-based and rely on the following strategy. As a first step, we identify a suitable LP-relaxation to use for our problems. To this extent, we show that we can reduce our problems to vertex-deletion problems in a *bipartite* graph, in which the goal is to remove a subset of vertices in order to turn some special nodes into *essential* vertices in the remaining graph. This reinterpretation of the problem allows us to write a formulation that uses a set of *flow-type* valid constraints, and exploiting the properties of this flow will be crucial to round fractional solutions into integral ones.
In addition, we show lower bounds on the integrality gap of the LP relaxations we use, that show that our analysis are almost tight. We give a $\frac{3}{2}$ lower bound on the integrality gap in the max-weight case (in Appendix G), and a $\Omega(\gamma)$ lower bound in the min-weight case (in Appendix H), that asymptotically matches the developed approximation factor. The lower bound for the min-weight case holds even on graph with *bounded* (constant) degree, and to construct it we rely on suitable *unbalanced bipartite expander* graphs.
We conclude by showing that we can give an algorithm for the min-weight vertex-stabilizer problem whose approximation factor *is* bounded by the maximum degree of a vertex in G, *if* we have an additional information: namely, if we know which is the set of essential vertices in the final graph (Theorem 5). From a network bargaining perspective, this corresponds to stabilize instances *enforcing* that some specific players will always be able to enter in a deal in any stable outcome. Also for this latter case we show a matching lower bound on the integrality gap of the LP relaxation we use (in Appendix F). Our lower bounds show that to improve significantly our approximation factors a different strategy or at least different formulations have to be used.

Related works. Removing vertices or edges from a graph as to satisfy certain properties has been widely studied in the literature in many variants. The paper that is most closely related to our work is [4] that studied the edge-stabilizer problem in the unweighted setting, and in addition to the results previously mentioned, they give efficient approximation algorithms for sparse graphs and for regular graphs. Biró et al. [3] also studied the edge-stabilizer problem, but considering maximum-weight matchings instead of maximum-cardinality matchings, and showed NP-hardness for this case. Könemann et al. [10] studied a related problem of computing a minimum-cardinality *blocking set*, that is a set of edges F such that $G \setminus F$ has a fractional vertex cover of size at most $\nu(G)$ (but note that $G \setminus F$ might not be stable). They give approximation algorithms for sparse graphs. Mishra et al. [13] studied vertex-removal and edge-removal problems to turn a graph into a König-Egerváry graph. Among other results, they give a $O(\log n \log\log n)$ approximation for the vertex-removal case in the unweighted setting, and show that assuming Unique Game Conjecture, both the minimum vertex-removal and edge-removal problems do not admit a constant factor approximation algorithm. We note that their hardness results do not seem to be helpful for our setting, since the graphs used in their reductions are in fact stable.

2 Minimum cardinality vertex-stabilizers

We first prove that the removal of any minimal vertex-stabilizer does not decrease the cardinality of a maximum matching in the resulting graph. Here $G \setminus S$ denotes the graph obtained by removing from $G = (V, E)$ the subset of vertices $S \subseteq V$.

Theorem 1. *For any minimal vertex-stabilizer $S \subseteq V$ of a graph $G = (V, E)$, we have $\nu(G \setminus S) = \nu(G)$.*

Before giving a proof, we need a proposition (see [9]) that follows from standard results in matching theory, and uses the notion of *M-flower* for a maximum matching M of G. An M-flower is a subgraph of G formed by a u, v-path of even length that alternates edges in $E \setminus M$ and edges in M, plus a cycle containing v of $2k + 1$ edges, for some integer $k \geq 1$, in which exactly k edges are in M (see Fig. 2 in Appendix A).

Proposition 1. [9] *Given graph G, the following are equivalent characterizations of a stable graph: (i) The set of inessential vertices of G are pairwise non adjacent, (ii) $\nu(G) = \tau_f(G)$, (iii) There is no M-flower in G for any maximum matching M. Moreover, if G is not stable, then for every maximum matching M there is an M-flower.*

Proof of Theorem 1. Let S be a minimal vertex-stabilizer of $G = (V, E)$, and M be a maximum matching of $G \setminus S$. Suppose by contradiction that $|M| < \nu(G)$. By classical results on matching theory [2], since M is not a maximum matching in G there exists an *M-augmenting path* P in G, that is, a path P that alternates edges from $E \setminus M$ and edges from M with endpoints s and t which are exposed by M. Clearly, we must have $|S \cap \{s, t\}| \geq 1$, otherwise P would be an augmenting path in $G \setminus S$, contradicting maximality of M. We distinguish two cases.
Case 1: $|S \cap \{s, t\}| = 1$. Without loss of generality, assume $s \in S$. In this case, we will show that $S' = S \setminus \{s\}$ is a vertex-stabilizer of G, which is a contradiction to the minimality of S. Consider the matching $M' = M \Delta P$, where Δ denotes the symmetric

difference operator. M' is a matching of $G \setminus S'$ and $|M'| = |M| + 1$. Since adding one vertex to an arbitrary graph can increase the size of maximum matching by at most one, we deduce that M' is a maximum matching of $G \setminus S'$, hence $\nu(G \setminus S') = |M'|$. We now prove that $G \setminus S'$ is stable by showing that $\nu(G \setminus S') = \tau_f(G \setminus S')$. Let $y \in \mathbb{R}^{V \setminus S}_{\geq 0}$ be a minimum size fractional vertex cover of $G \setminus S$. By stability of $G \setminus S$, $\nu(G \setminus S) = \mathbf{1}^T y$. Define vector $y' \in \mathbb{R}^{V \setminus S'}_{\geq 0}$ as $y'_v = y_v$ for all $v \in V \setminus S$, and $y'_s = 1$. Obviously y' is a fractional vertex cover of $G \setminus S'$. So we have $\tau_f(G \setminus S') \leq \mathbf{1}^T y' = \mathbf{1}^T y + 1 = \nu(G \setminus S) + 1 = \nu(G \setminus S')$, i.e., $G \setminus S'$ is stable.
Case 2: $|S \cap \{s, t\}| = 2$. We first observe that $(G \setminus S) \cup \{s\}$ does not contain any M-augmenting path. Otherwise, by the same arguments as in Case 1, we can deduce that $S \setminus \{s\}$ is a vertex-stabilizer, and obtain a contradiction. Similarly, $(G \setminus S) \cup \{t\}$ does not contain any M-augmenting path. Let $S' = S \setminus \{s, t\}$, and $M' = M \Delta P$. We first show that M' is a maximum matching in $G \setminus S'$. If not, then $\nu(G \setminus S') \geq \nu(G \setminus S) + 2$. Let M'' be maximum matching in $G \setminus S'$. If we remove s from $G \setminus S'$, we delete at most one edge of M''. Therefore, $\nu((G \setminus S) \cup \{s\}) \geq \nu(G \setminus S) + 1$. However, this implies that M is not a maximum matching in $(G \setminus S) \cup \{s\}$, and therefore $(G \setminus S) \cup \{s\}$ contains an M-augmenting path contradicting our first observation. Since M' is a maximum matching in $G \setminus S'$, and $G \setminus S'$ is not stable, by Proposition 1 there exists an M'-flower F, with vertex set $u_1, \ldots, u_p$, with u_1 being the M'-exposed vertex on the even-length path (see Fig. 3 in Appendix A). Note that F cannot be vertex disjoint from P: otherwise, F would be an M-flower as well in $G \setminus S$, contradicting stability of $G \setminus S$. It follows that $F \cup P$ is a connected subgraph of G. Let u_i be the node with the smallest index i that belongs to both F and P. Note that $i \neq 1$, since u_1 is M'-exposed and every node in P is instead M'-covered. Moreover, i is necessarily an even number: if odd, then the edge $\{u_{i-1}, u_i\}$ is in both P and F, contradicting our choice of i. Furthermore, note that the edge $\{u_i, u_{i+1}\}$ belongs to both P and F. Consider the path Q_1 that is the subpath of P connecting u_i to either s or t in $P \setminus \{u_i, u_{i+1}\}$, and the path Q_2 that is the subpath in F with vertex set $u_1, \ldots, u_i$. Their union $Q_1 \cup Q_2$ forms a path from u_1 to either s or t, say s (the other case is similar). In this case, $Q_1 \cup Q_2$ is an M-augmenting path in $(G \setminus S) \cup \{s\}$ (see Fig. 3 in Appendix A), contradicting our first observation. □

We now state our algorithm to find a minimum cardinality vertex-stabilizer, that relies on the notion of *Edmonds-Gallai Decomposition* (EGD) of a graph. The EGD of a graph $G = (V, E)$ is a partition of the set of vertices V into 3 sets (B, C, D) where B is the set of inessential vertices of G, C is the set of essential vertices of G that have at least one adjacent vertex in B, and D is the remaining essential vertices of G (see Fig. 4 in Appendix A). The set C is called the *Tutte-set* of G.

Algorithm 1

1. Compute the EGD (B, C, D) of G, and a maximum matching M^* of G that covers the maximum possible number of isolated vertices in the graph $G[B]$.
2. Let $G_1, \ldots, G_k$ be the non-singleton components of $G[B]$ with one vertex exposed by M^*. Set $S := \bigcup_{i=1}^{k} \{v_i\}$ where v_i is the M^*-exposed vertex of G_i.

Theorem 2. *Algorithm 1 is a polynomial-time algorithm to compute a minimum cardinality vertex-stabilizer S of a given graph G.*

We here sketch the main ideas of the proof. Let k be as in Algorithm 1. First, we note that k is a lower bound on the size of any minimum vertex-stabilizer. This has been proved by Bock et al. [4] for edge-stabilizers, but their proof in fact extends to vertex-stabilizers if one can assume Theorem 1 (see Lemma 8 in Appendix B). Then, we show that $G \setminus S$ is stable, by constructing a fractional vertex cover of $G \setminus S$ of size equal to $|M^*|$. This uses structural properties of maximum matchings and EGD of graphs. Since the techniques we use here are similar to [4], we defer the full proof to Appendix C.

3 The weighted case

We here deal with the vertex-stabilizer problem in the weighted setting, that is much more challenging than the unweighted one. To develop our approximation results, we first find a reformulation of our problems in *bipartite graphs*. Next lemma follows easily from Theorem 1 (proof in Appendix C).

Lemma 1. *Let (B, C, D) be the EGD of a graph G. Let $G_1, G_2, \cdots, G_p$ be the components of $G[B]$ where $G_i = (V_i, E_i)$. Let S be an optimal solution to a min-weight vertex-stabilizer (resp. max-weight vertex-stabilizer) instance defined on G. Then, (i) S is a subset of B, (ii) $|S \cap V_i| \leq 1$, (iii) if $|S \cap V_i| = 1$, then the vertex $g_i \in S$ of G_i is a minimum weight vertex in G_i.*

We can use Lemma 1 to simplify our input. If S contains a vertex from a component G_i, then it must be one of the vertices with minimum weight in G_i. Therefore, we shrink each non-singleton component G_i to a vertex g_i with minimum weight among the vertices in the component, and we call it a *pseudonode* (we remove multiple copies of the same edge created with this operation, if any). Additionally, we know that $S \cap D = \emptyset$, so we can safely ignore these vertices and temporarily remove them from G. Lastly, we remark that it is well-known (see e.g. [16]) that every maximum matching of G matches all vertices in C to vertices in different components of $G[B]$; therefore, we ignore and remove edges between vertices in C from G. In this way we construct from G a weighted bipartite graph $G_b = (\tilde{B} \cup C, \tilde{E})$, where $\tilde{E} \subseteq E$, and $\tilde{B}$ consists of two sets of vertices: the set of pseudonodes, call this set B_1, and vertices corresponding to singletons in $G[B]$, call this set B_2. By construction and our previous remark, $\nu(G_b) = |C|$ and S naturally corresponds to a subset of $\tilde{B}$ of the same weight.

Definition 1. *Let $H = (U \cup W, F)$ be a bipartite graph and $U_1 \subseteq U$. We call $S \subseteq U$ a U_1-*essentializer *if all vertices in $U_1 \setminus S$ are essential in the graph $H \setminus S$.*

Next lemma (proof in Appendix C) basically shows that there is an approximation preserving reduction between the min-weight (resp. max-weight) vertex-stabilizer problem defined on G, and the problem of finding a suitable B_1-essentializer S that minimizes $\sum_{v \in S} w_v$ (resp. maximizes $\sum_{v \notin S} w_v$) in the weighted bipartite graph G_b.

Lemma 2. *Let $\tilde{S} \subseteq \tilde{B}$ be a B_1-essentializer of G_b that satisfies $\nu(G_b \setminus \tilde{S}) = \nu(G_b)$. Then $\tilde{S}$ corresponds to a vertex-stabilizer of G (of the same weight). Let $S \subseteq V$ be an optimal solution to a min-weight vertex-stabilizer (resp. max-weight vertex-stabilizer) instance defined on G. Then S corresponds to a B_1-essentializer in G_b (of the same weight) that satisfies $\nu(G_b \setminus S) = \nu(G_b)$.*

Next, we give an integer programming description of the set of B_1-essentializers, whose relaxation will be at the heart of our algorithms.

Integer programming description. Given $G_b = (\tilde{B} \cup C, \tilde{E})$, with $\tilde{B} = B_1 \cup B_2$, we introduce a binary variable z_v for $v \in \tilde{B}$ to denote if v is in a B_1-essentializer S (i.e. $z_v = 1$ if $v \in S$). We also introduce a binary variable y for $v \in \tilde{B} \cup C$ with the following meaning: for $v \in \tilde{B}$, we let $y_v = 1$ denote if v is an essential node in $G_b \setminus S$; for $v \in C$ instead, we let $y_v = 1$ denote if v is always matched to an inessential node in any maximum matching of $G_b \setminus S$. For a set of vertices T, we let $y(T) = \sum_{v \in T} y_v$, and $N(T)$ denote the set of neighbours (i.e. adjacent vertices) of T. We let

$$
\begin{aligned}
P_I := \Big\{(z, y) : \quad & y_v + z_v \geq 1, && \text{for } v \in B_1 && (1)\\
& y_v + y_u + z_v \geq 1, && \text{for } \{u, v\} \in \tilde{E}, v \in B_2, u \in C && (2)\\
& y(N(A)) \geq |A| - y(A), && \text{for } A \subseteq C && (3)\\
& y(V) = |C|, && && (4)\\
& z \in \{0,1\}^{\tilde{B}},\ y \in \{0,1\}^{\tilde{B} \cup C} \quad \Big\}.
\end{aligned}
$$

Let us give an intuition of the meaning of the linear constraints. Inequality (1) states that a vertex in B_1 is either essential in $G_b \setminus S$ or it is removed (as required by Definition 1). Inequality (2) states that if a vertex v in B_2 is not removed then either v is essential in $G_b \setminus S$ or all of its neighbours have to be matched to inessential vertices in $G_b \setminus S$. The reason is that, if v is inessential in $G_b \setminus S$ but some neighbour u of v is matched to an essential vertex v' in some maximum matching M of $G_b \setminus S$, then it is possible to construct an even length M-alternating path between some M-exposed vertex to v', contradicting the fact that v' is essential. Inequality (3) is a translation of Hall's theorem, and states that there exists a matching between vertices in C with y-value 0 and their neighbours with y-value 1, that covers all vertices in C with y-value 0. The reason is that such vertices will always be matched to essential vertices in $G_b \setminus S$ by any maximum matching. Inequality (4) basically ensures that there is a partition of vertices in C into those that will always be matched to inessential vertices and those that will always be matched to essential vertices of $G_b \setminus S$ by any maximum matching. We would like to emphasize that inequalities (3) are crucial to have a meaningful formulation for our problem. Next lemma (proof in Appendix C) makes this intuition rigourous.

Lemma 3. *P_I describes the set of B_1-essentializers of the graph G_b.*

We denote by P_f the polytope obtained by relaxing the binary constraints of P_I, i.e. replacing them with $0 \leq z \leq 1$ and $0 \leq y \leq 1$. When dealing with fractional points, Inequality (3) does not correspond to Hall's theorem anymore, but it naturally ensures the existence of a *flow* of value $|C| - y(C)$ from vertices in C to vertices in $\tilde{B}$. Among other things, this also implies that although this set contains exponentially (in the size of G_b) many inequalities, we can separate over them in polynomial time. Proof of next lemma is in Appendix C.

Lemma 4. *Construct a directed network $\mathcal{N} = (V_{\mathcal{N}}, A_{\mathcal{N}})$ from graph $G_b = (\tilde{B} \cup C, \tilde{E})$ with $V_N = \tilde{B} \cup C \cup \{s, t\}$ and $A_{\mathcal{N}} = \{(s, u) : u \in C\} \cup \{(v, t) : v \in \tilde{B}\} \cup \tilde{E}$ where* *the edges in $\tilde{E}$ are oriented from C to $\tilde{B}$. Let $(z, y) \in P_f$. Assign y_v amount of capacity to each arc (v, t), $(1 - y_u)$ amount of capacity to each arc (s, u), and ∞ capacity to arcs in $\tilde{E}$. Then, there exists a maximum $s - t$ flow in $\mathcal{N}$ of value $y(\tilde{B}) = |C| - y(C)$.*

Exploiting the structure of this flow, we can derive useful properties on the extreme points of P_f. In particular, we have the following lemma (proof in Appendix C):

Lemma 5. *Let (z, y) be an extreme point of P_f. There exists a maximum matching in G_b between the set of vertices $\{v \in \tilde{B} : y_v > 0\}$ and the set of vertices $\{u \in C : y_u < 1\}$ of cardinality $|\{v \in \tilde{B} : y_v > 0\}|$.*

Finally, we note that the problem of finding a B_1-essentializer S that maximizes $\sum_{v \notin S} w_v$, or minimizes $\sum_{v \in S} w_v$, can be formulated respectively as

$$\max\Big\{\sum_{v \in \tilde{B}} w_v(1 - z_v) : (z, y) \in P_I\Big\}, \quad \text{and} \quad \min\Big\{\sum_{v \in \tilde{B}} w_v z_v : (z, y) \in P_I\Big\}. \quad (5)$$

Algorithm for max-weight vertex-stabilizer. Given a graph $G = (V, E)$ with weights $w_v \geq 0\ \forall v \in V$, we construct from G a weighted bipartite graph $G_b = (\tilde{B} \cup C, \tilde{E})$, with $\tilde{B} = B_1 \cup B_2$, as described in the beginning of Section 3. We then apply Algorithm 2 that relies on solving the LP relaxation of the maximization IP in (5).

Algorithm 2

1. Let $(z^*, y^*) \leftarrow$ optimal extreme point of $\max\{\sum_{v \in \tilde{B}}(1 - z_v) : (z, y) \in P_f\}$.
2. Set $B_+ := \{v \in \tilde{B} : 0 < y^*_v\}$; $B^1_0 := \{v \in \tilde{B} : y^*_v = 0, z^*_v = 1\}$; $B^f_0 := \{v \in \tilde{B} : y^*_v = 0, 0 < z^*_v < 1\}$.
3. **if** $w(B_+) \leq w(B^f_0)$ **then** set $S = (B_+ \cup B^1_0)$, **else** set $S = (B^f_0 \cup B^1_0)$.
4. **While** $\nu(G_b \setminus S) < |C|$ **do**: find $v \in S$ such that $\nu(G_b \setminus (S \setminus \{v\})) > \nu(G_b \setminus S)$, and set $S := S \setminus \{v\}$.

Theorem 3. *There is a polynomial-time LP-based 2-approximation algorithm for the max-weight vertex-stabilizer problem.*

Proof. We consider the set S output by Algorithm 2. Note that if S is a B_1-essentializer *and* $\nu(G_b \setminus S) = \nu(G_b)$, then it corresponds to a vertex-stabilizer in G by Lemma 2. Still, Lemma 2 implies that to prove the claimed approximation guarantee, it is enough to prove that S is a 2-approximated solution for the problem of finding a B_1-essentializer for G_b that maximizes the weight of the non selected vertices.
First, we show that (a) $\nu(G_b \setminus S) = \nu(G_b)$ and (b) every vertex in $B_1 \setminus S$ is essential in $G_b \setminus S$, i.e. S is a B_1-essentializer. Note that (a) holds by construction after step 4 (recall that $\nu(G_b) = |C|$ and $S \cap C = \emptyset$, therefore it is always possible to perform step 4 until the while condition is not satisfied anymore). Moreover, all vertices added in step 4 are essential vertices. We are left with (b). Define $C_f = \{u \in C : y^*_u < 1\}$. Furthermore, partition the set of vertices in $\tilde{B}$ in 4 sets: B_+, B^1_0, B^f_0 and $B^0_0 := \{v \in \tilde{B} : z^*_v = 0\ \&\ y^*_v = 0\}$ (the definition of the first 3 sets is given in Algorithm 2). Note that the vertices in B_1 are either in B^1_0 or B_+, so if $S = B_+ \cup B^1_0$, then $G_b \setminus S$ does not contain any B_1 vertex, and we have nothing to show. Suppose instead $S = B^f_0 \cup B^1_0$. Note

that there does not exist any edge between $v \in B_0^0$ and $u \in C_f$, because $y_v^* + z_v^* = 0$ holds for v and $y_u^* < 1$ holds for u, and therefore Inequality (2) will be violated for the edge $\{v, u\}$, contradicting feasibility of (z^*, y^*). Therefore, the neighbours of vertices C_f in $G_b \setminus S$ are vertices in B_+ and by Lemma 5, we know that there is a matching between C_f and B_+ covering all vertices in B_+. Since every maximum matching in $G_b \setminus S$ covers all the vertices in C, it must cover all vertices in C_f, therefore it must be the case that $|C_f| = |B_+|$ and every maximum matching in $G_b \setminus S$ covers all the vertices in B_+, i.e. all the vertices in B_+ are essential. Since $(B_1 \setminus S) \subseteq B_+$, the result follows.

To conclude the proof, we argue that the weight of the vertices in $G_b \setminus S$ is at least $\frac{1}{2}$ the optimal value of the LP. Let $w_0 = w(B_0^0)$, $w_1 = w(B_+)$, $w_2 = w(B_0^f)$. Note that the weight of the vertices in the graph $G_b \setminus S$ is at least $w_0 + \max(w_2, w_1)$ which is at least half of $w_0 + w_1 + w_2 = \sum_{v \in \tilde{B}} w_v - \sum_{v : z_v^* = 1} w_v$, which is clearly an upper bound on the optimal value of the LP. □

Algorithm for min-weight vertex-stabilizer. Given a graph $G = (V, E)$ with weights $w_v \geq 0 \ \forall v \in V$, we construct a weighted bipartite graph $G_b = (\tilde{B} \cup C, \tilde{E})$, with $\tilde{B} = B_1 \cup B_2$ obtained from G as described in the beginning of Section 3. We then apply Algorithm 3 that relies on solving the LP relaxation of the minimization IP in (5).

Algorithm 3

1. Solve the LP: $\min \left\{ \sum_{v \in \tilde{B}} w_v z_v \ : \ (z, y) \in P_f \right\}$ to get an extreme point optimal solution (z, y), and set $S := \{v : z_v \geq \frac{1}{|C|+1}\}$.
2. **While** $\nu(G_b \setminus S) < |C|$ **do**: find $v \in S$ such that $\nu(G_b \setminus (S \setminus \{v\})) > \nu(G_b \setminus S)$, and set $S := S \setminus \{v\}$.

Theorem 4. *There is a polynomial-time LP-based $(\gamma + 1)$-approximation algorithm for the min-weight vertex-stabilizer problem, where γ is the size of the Tutte-set of G.*

Proof. We consider the set S output by Algorithm 3. As for the max-weight case, due to Lemma 2 and step 2 of the algorithm, to prove the theorem it is enough to show that S is a $(|C| + 1)$-approximated solution for finding a B_1-essentializer for G_b that minimizes the weight of the selected vertices. Trivially, $w(S) \leq (|C|+1) \sum_{v \in \tilde{B}} w_v z_v$, therefore the approximation factor guarantee holds. It remains to show that S is in fact a B_1-essentializer for G_b.

Let $\tilde{S}$ be the set S *before* executing step 2 of the algorithm. We will prove that each $v \in B_1 \setminus \tilde{S}$ is essential in $G_b \setminus \tilde{S}$. This is enough, since every vertex added back in step 2 will be essential by construction, and this addition cannot make any vertex in B_1 inessential. Let us assume by contradiction that $v_0 \in B_1 \setminus \tilde{S}$ is inessential in $G_b \setminus \tilde{S}$. In this case, if we apply Edmonds' Blossom Algorithm [7] in $G_b \setminus \tilde{S}$, we can find a maximum matching M that exposes v_0 and a so-called *frustrated tree* $T = (V_T, E_T)$ containing v_0 with the following properties: (i) $|E_T \cap M| = |V_T \cap C|$, and all vertices in $V_T \setminus \{v_0\}$ are covered by M, and (ii) the neighbours of the set of vertices $V_T \cap \tilde{B}$ in $G_b \setminus \tilde{S}$ are all in the tree T (we refer to [7,6] for details). Note that the neighbours of $V_T \cap \tilde{B}$ in $G_b \setminus \tilde{S}$ are the same as the neighbours of $V_T \cap \tilde{B}$ in G_b, i.e. $N(V_T \cap \tilde{B}) = V_T \cap C$ as $\tilde{S} \subseteq \tilde{B}$. Feasibility of (z, y) implies that for each matching edge $\{u, v\} \in M$, we have

$y_u + y_v + z_v \geq 1$. Since $\tilde{S}$ removed all vertices with z-value $\geq \frac{1}{|C|+1}$, for each edge $\{u, v\} \in M$, $y_u + y_v > 1 - \frac{1}{|C|+1}$. Let $M_T := M \cap E_T$. We have

$$y(V_T) = y_{v_0} + \sum_{\{u,v\} \in M_T} (y_u + y_v) > (1 - \frac{1}{|C|+1}) + |M_T|(1 - \frac{1}{|C|+1})$$

$$= |M_T| + 1 - \frac{|M_T|+1}{|C|+1} \geq |M_T|,$$

where the first inequality follows from the Inequality (1) associated to v_0, and the last inequality follows from the fact that $|M_T| \leq |C|$. Furthermore, for set $A = V_T \cap C$, since $|M_T| = |A|$ by (i), we have

$$y(A) + y(N(A) \cap V_T) = y(V_T) > |M_T| = |A|. \tag{6}$$

If we consider the directed network $\mathcal{N}$and the $s - t$ flow as in Lemma 4, (6) says that the capacity $y(N(A) \cap V_T)$ of the arcs between t and $N(A) \cap V_T$ is strictly larger than the flow sent on the arcs from s to A (that can be at most $|A| - y(A)$). Since a maximum flow necessarily saturates *all* the edges from $N(A) \cap V_T$ to t, there is a neighbour of $(N(A) \cap V_T)$ which is not in A who sends positive flow to some vertex in $N(A) \cap V_T$, but this contradicts property (ii) of T, as $N(N(A) \cap V_T) = N(\tilde{B} \cap V_T) = A$. □

We remark here that we can show a tight lower bound of $\Omega(\gamma)$ on the integrality gap of the minimization IP in (5) that holds even on graphs with *constant* degree. However, we now describe an algorithm whose approximation ratio is bounded by the maximum degree (δ) of a vertex in G, which needs to know the set of essential vertices in the final stable graph (our reduction in Theorem 7 shows that also this problem is NP-hard).

Theorem 5. *There is a δ-approximation algorithm for the min-weight vertex-stabilizer problem, if we know the set of essential vertices in the final stable graph.*

This translates into knowing which subset $B' \subseteq \tilde{B}$ of vertices of G_b must have y-value 1 in our formulation (5). In this setting, we therefore add to P_f the following constraints: $y_v = 1$ for $v \in B'$, and $y_v = 0$ for $v \in \tilde{B} \setminus B'$. We call the resulting LP problem ($\mathcal{P}_1$). To find good integral solution to ($\mathcal{P}_1$), we introduce a new optimization problem ($\mathcal{P}_2$) which in fact corresponds to a minimum cost flow problem, as follows. We define a new weight vector $\overline{w}$ for vertices in C as $\overline{w}_u = \sum_{v \in N(u) \cap (B_2 \setminus B')} \frac{w_v}{\delta}$. Then we remove all vertices in $\tilde{B}$ but B', obtaining graph G'_b. For a set of vertices T, let $N'(T)$ denote the set of neighbours of T in G'_b (while $N(T)$ denotes the neighbours in G_b). We introduce a variable f_u $\forall u \in C$, and let ($\mathcal{P}_2$) be the following problem:

$$\min \left\{ w(B_1 \backslash B') + \sum_{u \in C} \overline{w}_u f_u : |N'(A)| \geq \sum_{u \in A} f_u \;\; \forall A \subseteq C, \; \sum_{u \in C} f_u = |B'|, 0 \leq f \leq 1 \right\}.$$

A part from the constant term ($w(B_1 \setminus B')$) in the objective function, this problem corresponds to a minimum cost flow problem in a directed network $\overline{\mathcal{N}}$ constructed from G'_b via a similar process as in Lemma 4. Formally: (i) add a vertex s and for each vertex $u \in C$ add an arc (s, u) with capacity 1 and cost $\overline{w}_u$, (ii) orient all the edges from C to B' in G'_b and set their capacity to infinity and their cost zero, (iii) add a vertex t and for each $v \in B'$ add an arc (v, t) with capacity 1 and cost 0. An optimal solution to ($\mathcal{P}_2$) can be mapped to a minimum cost $s-t$ flow of value $|B'|$ in $\overline{\mathcal{N}}$, and vice versa. Therefore, since all the capacities are integral, we know that ($\mathcal{P}_2$) has an optimal *integral* solution. Next lemma (proof in Appendix C) maps solutions of these two optimization problems.

Lemma 6. *An optimal solution of ($\mathcal{P}_1$) can be mapped into a solution of ($\mathcal{P}_2$) with no greater weight. An integral solution of ($\mathcal{P}_2$) can be mapped into an integral solution of ($\mathcal{P}_1$) whose weight is at most a δ-factor larger.*

We can now replace step 1 of Algorithm 3 with solving ($\mathcal{P}_2$) and mapping the solution into an integral solution of ($\mathcal{P}_1$) (we keep step 2). Combining Lemma 6 with Lemma 2, one can easily see that this new algorithm yields a proof of Theorem 5.

References

1. A. Agarwal, M. Charikar, K. Makarychev, and Y. Makarychev. O($\sqrt{\log n}$) approximation algorithms for min UnCut, min 2CNF deletion, and directed cut problems. In *Proceedings of STOC 2005*, pages 573–581, 2005.
2. C. Berge. Two theorems in graph theory. *Proceedings of the National Academy of Sciences of the United States of America*, 43(9):842, 1957.
3. P. Biró, M. Bomhoff, P. A. Golovach, W. Kern, and D. Paulusma. Solutions for the stable roommates problem with payments. In *Graph Theoretic Concepts in Computer Science, LNCS*, volume 7551, pages 69–80, 2012.
4. A. Bock, K. Chandrasekaran, J. Könemann, B. Peis, and L. Sanità. Finding small stabilizers for unstable graphs. *Proceedings of IPCO 2014, LNCS*, 8494:150–161, 2014.
5. G. Chalkiadakis, E. Elkind, and M. Wooldridge. Computational aspects of cooperative game theory. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 2011.
6. W. Cook, W. Cunningham, W. Pulleyblank, and A. Schrijver. *Combinatorial Optimization*. John Wiley & Sons, Inc., New York, NY, USA, 1998.
7. J. Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.
8. N. Garg, V. V. Vazirani, and M. Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. *SIAM Journal on Computing*, 25:698–707, 1993.
9. J. Kleinberg and É. Tardos. Balanced outcomes in social exchange networks. In *Proceedings of STOC 2008*, pages 295–304, 2008.
10. J. Könemann, K. Larson, and D. Steiner. Network bargaining: Using approximate blocking sets to stabilize unstable instances. In *Theory of Computing Systems*, pages 655–672, 2015.
11. E. Korach. Flowers and Trees in a Ballet of K_4, or a Finite Basis Characterization of Non-König-Egerváry Graphs. Technical Report 115, IBM Israel Scientific Center, 1982.
12. E. Korach, T. Nguyen, and B. Peis. Subgraph characterization of Red/Blue-Split Graph and König Egerváry Graphs. In *Proceedings of SODA 2006*, pages 842–850, 2006.
13. S. Mishra, V. Raman, S. Saurabh, S. Sikdar, and C. Subramanian. The complexity of König subgraph problems and above-guarantee vertex cover. *Algorithmica*, 61(4):857–881, 2011.
14. J. Nash. The bargaining problem. *Econometrica*, 18:155–162, 1950.
15. N. Nisan, T. Roughgarden, É. Tardos, and V. V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, New York, NY, USA, 2007.
16. A. Schrijver. *Combinatorial optimization*. Springer, New York, 2003.
17. L. S. Shapley and M. Shubik. The assignment game: the core. *International Journal of Game Theory*, 1(1):111–130, 1971.
18. F. Sterboul. A characterization of the graphs in which the transversal number equals the matching number. *Journal of Combinatorial Theory Series B*, 27:228–229, 1979.

A Omitted figures

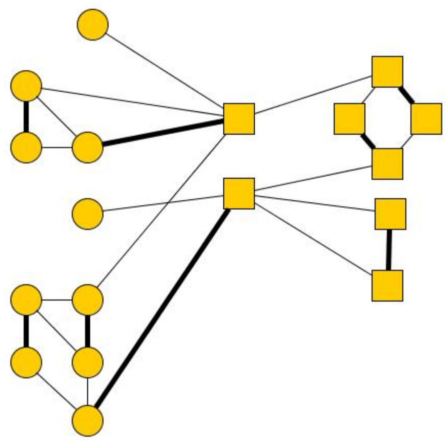

Fig. 1: An example showing inessential vertices, shown by circle vertices, and essential vertices, shown by square vertices, of a graph G along with a maximum matching M shown by bold edges.

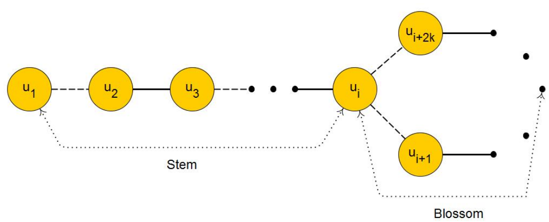

Fig. 2: Depiction of an M-flower, solid edges are M edges and dotted edges are $E \setminus M$ edges.

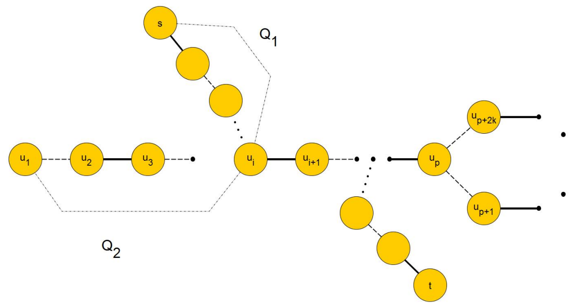

Fig. 3: Construction of an M-augmenting path with one endpoint in $\{s, t\}$. The solid edges are M' edges and dotted edges are $E \setminus M'$ edges. Note that $M' \cap Q_2 = M \cap Q_2$ while $M \cap Q_1 = Q_1 \setminus M'$.

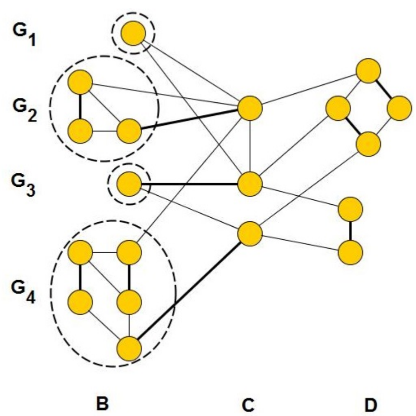

Fig. 4: Example of Edmonds-Gallai decomposition for graph G with maximum matching M showed by bold edges.

B Preliminaries for the omitted proofs

In this section, we present the preliminaries needed for presenting the omitted proofs. For this section, we assume that (B, C, D) is the EGD of the input graph G, and $G_1, G_2, \cdots, G_r$ are the connected components of the graph $G[B]$ induced by B.

The following lemma states some known properties of EGD of graphs. For a proof, see e.g. [16].

Lemma 7. *[16] Let (B, C, D) be Edmonds-Gallai Decomposition of a graph G. Let $G_1, G_2, \cdots, G_r$ be the components of the graph $G[B]$ induced by B. Then, for each non-singleton component $G_i = (V_i, E_i)$ and each $v \in V_i$, the graph $G_i \setminus v$ admits a perfect matching. Furthermore, if M is a maximum matching in G, then: (a) M induces* *a perfect matching between vertices of D; (b) M matches C into distinct components of $G[B]$; (c) M induces a near-perfect matching in each G_i, i.e., $|M \cap E_i| = \frac{|V_i|-1}{2}$.*

The following lemma gives a lower bound on the size of a minimum vertex-stabilizer. It has been proved by Bock et al. [4] for edge-stabilizers, but their proof extends to vertex-stabilizers as well, assuming Theorem 1. We report here a proof for completeness.

Lemma 8. [4] *Let (B, C, D) be the EGD of a graph G, and let M^* be a maximum matching of G that covers the maximum possible number of isolated vertices (i.e. singletons) in the graph $G[B]$. Let k be the number of non-singleton components of $G[B]$ with one vertex exposed by M^*. Then for any minimal vertex-stabilizer S, we have $|S| \geq k$.*

Proof. [4] Let M be a maximum matching in G that also matches the maximum possible number of isolated vertices in $G[B]$. Let $G_1, \ldots, G_k$ denote the non-singleton components in $G[B]$ with at least one vertex exposed by M. Let S be a minimum vertex-stabilizer and $H = G \setminus S$. For each component $G_1, \ldots, G_k$, at least one vertex $v_i \in G_i$ becomes essential in H. If not, either we removed all vertices in G_i (but this is not possible since it would decrease the the size of maximum matching) or it means that inessential vertices do not form a stable set in H, again a contradiction.

Pick a maximum matching N in H. Then, N will cover all these vertices $v_1, \ldots, v_k$ that are essential in H. Since G_i is factor-critical and M matches all but one vertex in G_i using edges in G_i, we may assume without loss of generality, that M exposes all these vertices. The graph $M \Delta N$ is a vertex-disjoint union of even cycles and even paths since $|M| = |N| = \nu(G)$. Consider the k disjoint paths starting at the vertices $v_1, \ldots, v_k$ in $M \Delta N$. We observe that at least one vertex in each of these paths should belong to S, otherwise we can obtain a maximum matching in H that exposes the starting vertex v_i, thus contradicting v_i being an essential vertex in H. Hence $|S| \geq k$. □

The following two lemmas are keystones in most of the proofs presented in Appendix C.

Lemma 9. *Let S be a vertex-stabilizer for a given graph G, and let M be a maximum matching of $G \setminus S$ with $|M| = \nu(G)$ (this matching exists by Theorem 1). Let $G_i = (V_i, E_i)$ be a non-singleton component of $G[B]$, then all vertices of $V_i \setminus S$ are essential in $G \setminus S$. Moreover, every M-exposed vertex v with M-alternating path in G to a vertex in a non-singleton component of $G[B]$ must be in S. Therefore, if G_i contains an M-exposed vertex v, then $v \in S$.*

Proof. To prove this lemma, we exploit a key property of non-singleton components of $G[B]$ in EGD: each non-singleton component $G_i = (V_i, E_i)$ of $G[B]$ is factor-critical which means that for every vertex $v \in V_i$, $G_i - v$ has a perfect matching denoted by M_v. Since M is a maximum matching of G, so by Lemma 7, $|M \cap E_i| = \frac{|V_i|-1}{2}$. Now if $V_i \cap S = \emptyset$, every vertex of $v' \in V_i$ must be essential as otherwise if a vertex $v' \in V_i$ is inessential in $G \setminus S$, then every vertex $v'' \in V_i$ is inessential but since inessential vertices in a stable graph must form an independent set, this cannot happen. If $V_i \cap S \neq \emptyset$ then since M does not cover at most one vertex v in V_i, then $V_i \cap S = \{v\}$ and by Lemma 7 and Theorem 1, every vertex of $V_i \setminus S$ is essential in $G \setminus S$. So every vertex of $V_i \setminus S$ for non-singleton component $G_i = (V_i, E_i)$ is essential which means that any M-exposed vertex v with an alternating path to G_i must be in S as otherwise this vertex makes the vertices of G_i inessential. Since there is a trivial path (with length zero) from an M-exposed vertex v in a non-singleton component G_i to itself, $v \in S$. □

Lemma 10. *Given a set $S \subseteq V$ and a maximum matching M of a graph $G \setminus S$ with size $\nu(G)$, a set S is a vertex-stabilizer if it includes any M-exposed vertex v with M-alternating path to a vertex in a non-singleton component of $G[B]$. We call M a vertex-stabilizer-certificate of S.*

Proof. Let S be a set satisfying the condition in the lemma. We prove that $G \setminus S$ is stable by showing $\tau_f(G \setminus S) = \nu(G \setminus S) = |M|$. Partition B into two sets: B_1 and B_2, where B_2 contains the singleton vertices in $G[B]$ and $B_1 = B \setminus B_2$. Before describing a fractional cover y, we prove any vertex $v \in B_1 \setminus S$ is essential in $G \setminus S$ which is needed in proving that y is a fractional cover. Let set $B' \subseteq B \setminus S$ denote the set of vertices with M-alternating path to a vertex in $B_1 \setminus S$ (so $B_1 \setminus S \subseteq B'$). Since we removed all M-exposed vertices with alternating path to $B_1 \setminus S$, B' only includes M-covered vertices. A vertex in B' is either matched to a vertex in B' or to a vertex in C. Let C' denote the set of vertices matched to some vertex in B'. Note that C' vertices do not have a neighbour in $B \setminus (S \cup B')$ as any vertex v in $B \setminus (S \cup B')$ with a neighbour in C' has an M-alternating path to a vertex v' in B' and since v' has an M-alternating path to a vertex in $B_1 \setminus S$, so v has an M-alternating path to $B_1 \setminus S$ which is a contradiction as v must have been in S. Moreover, B' vertices do not have a neighbour in $C \setminus C'$ as otherwise since such a vertex u is matched to a vertex $v \in B$ (C vertices are essential), v has an M-alternating path to B', and consequently to a vertex in $B_1 \setminus S$ which is a contradiction as v must be in B' in this case. So M induces a perfect matching on $B' \cup C'$ and these vertices do not have a neighbour in $V \setminus (S \cup B' \cup C' \cup D)$, therefore any maximum matching of $G \setminus S$ must induce a perfect matching on this set (note D vertices are always matched to each other by Lemma 7). Hence all vertices in B' including $B_1 \setminus S$ are essential.

Now we describe a fractional vertex cover $y \in \mathbb{R}_{\geq 0}$ with $\mathbf{1}^T y = |M|$: For each essential vertex $v \in V \setminus (S \cup C)$, let $y_v = 0.5$, for a vertex $u \in C$ if u is matched to an essential vertex by M in $G \setminus S$, let $y_u = 0.5$, otherwise let $y_u = 1$, and for the rest of vertices, i.e., inessential vertices define $y_v = 0$.

First, we show that $\mathbf{1}^T y = |M|$. For an edge $e = uv \in M$, we claim $y_u + y_v = 1$. By Lemma 7, either $u, v \in D$, $u, v \in B$, or $u \in C, v \in B$. In the first case, the claim holds trivially. Since in the second case, by reasoning above, both u and v are essential in $G \setminus S$, $y_u + y_v = 0.5 + 0.5 = 1$. For the last case, depending on whether v is essential or not, we get $y_u + y_v = 0.5 + 0.5 = 1$ or $y_u + y_v = 1 + 0 = 1$, respectively. Since any v exposed by M is inessential, i.e., $y_v = 0$, $\mathbf{1}^T y = |M|$.

It remains to show that y is a vertex cover. In order to do so, we just need to show that a vertex v with $y_v = 0$ is not adjacent to a vertex u with $y_u = 0.5$. This follows from the above argument that all C' vertices are only adjacent to B' vertices and they do not have a neighbour in $B \setminus (B' \cup S)$. □

Next, we provide some useful lemmas on the structure of extreme points of P_f which are helpful in proving Lemma 5. Since we are working with fractional solution now, the Inequality (3) does not correspond to Hall's theorem any more, but it corresponds to a flow problem instead. Define a network $\mathcal{N} = (V_\mathcal{N}, A_\mathcal{N})$ as in Lemma 4, and let F be a corresponding maximum flow of $y(\tilde{B}) = |C| - y(C)$ from s to t (proved existence in Lemma 4). Let F_e denote the flow on edge $e = (u, v) \in A_\mathcal{N}$. We can assume that F is acyclic on $\tilde{E}$ edges as otherwise we can do the usual $\{+, -\}$ operations on the edges of a cycle in F, and obtain a new acyclic flow. So F gives us a collection of trees $T_1, T_2, \cdots, T_r$ on $\tilde{B} \cup C$. Now we can prove the following lemmas for each tree T_i. We would like to note that $\tilde{B}$-vertices with $y_v = 0$ form singleton components (or trivial components) and all the following lemmas hold for them trivially. So we can assume the trees in the claims are non-trivial components.

Lemma 11. *Let (y, z) be an extreme point of P_f, and let T_i be any tree obtained by flow F in network $\mathcal{N}$. Then, there is at most one vertex v with $0 < y_v, z_v < 1$ in tree T_i.*

Proof. Let us assume the contrary, i.e, there are two vertices $v_1, v_2 \in B$ with $0 < y_{v_1}, z_{v_1}, y_{v_2}, z_{v_2} < 1$ in T_i. Using F, we construct feasible solution $(y^+, z^+), (y^-, z^-) \in P_f$ such that $(y, z) = \frac{(y^+,z^+)+(y^-,z^-)}{2}$ which is a contradiction. Let $P = \{e_1, e_2, \cdots, e_{2l}\}$ be unique path in T_i from v_1 to v_2. All the edges in P have non-zero flow in F. Let $\epsilon = \min_{e \in P} F_e$. Define F^+, z^+, y^+ and F^-, z^-, y^- as follows

$$F^+_e = \begin{cases} F_e + \epsilon & \text{if } e = e_{2i}, \\ F_e - \epsilon & \text{if } e = e_{2i-1}, \\ F_e & \text{otherwise.} \end{cases} \quad z^+_v = \begin{cases} z_v + \epsilon & \text{if } v = v_1, \\ z_v - \epsilon & \text{if } v = v_2, \\ z_v & \text{otherwise.} \end{cases} \quad y^+_v = \begin{cases} y_v - \epsilon & \text{if } v = v_1, \\ y_v + \epsilon & \text{if } v = v_2, \\ y_v & \text{otherwise.} \end{cases}$$

$$F^-_e = \begin{cases} F_e - \epsilon & \text{if } e = e_{2i}, \\ F_e + \epsilon & \text{if } e = e_{2i-1}, \\ F_e & \text{otherwise.} \end{cases} \quad z^-_v = \begin{cases} z_v - \epsilon & \text{if } v = v_1, \\ z_v + \epsilon & \text{if } v = v_2, \\ z_v & \text{otherwise.} \end{cases} \quad y^-_v = \begin{cases} y_v + \epsilon & \text{if } v = v_1, \\ y_v - \epsilon & \text{if } v = v_2, \\ y_v & \text{otherwise.} \end{cases}$$

We claim that (y^+, z^+) and (y^-, z^-) are in P_f. Note that for each $v \in \tilde{B}$, $z^+ + y^+_v = z^-_v + y^-_v = z_v + y_v$, so Inequalities (1) and (2) are satisfied. Moreover $y^+(V) = y^-(V) = y(V) = |C|$, so Inequality (4) is satisfied as well. Inequality (3) is satisfied as we can send flow of $1 - y_u$ on edge su for each $u \in C$, then send flow of F^+_e (F^-_e) on each edge $e \in \tilde{E}$, and then flow of y^+_v (y^-_v) on edge vt for each $v \in \tilde{B}$ which show that Inequality (3) is satisfied as $F^+(s) = F^-(s) = F(s) = |C| - y(C)$. Note that $(y, z) = \frac{(y^+,z^+)+(y^-,z^-)}{2}$ which is a contradiction, so the claim follows. □

Corollary 1. *Let (y, z) be an extreme point of P_f, and let T_i be any tree obtained by flow F in network $\mathcal{N}$. Then, there is at most one $\tilde{B}$-leaf in T_i.*

Proof. Let v be a leaf of T_i in $\tilde{B}$ and let u be a neighbour of v in T_i which must be in C. Note that $y_v + z_v \geq 1 - y_u$ by Inequality (1) or (2). If $z_v = 0$ then the capacity of the arc (v, t) which is y_v, is enough for routing all the flow coming to u from s which is $1 - y_u$. Therefore, if u sends some flow to some other vertex $v' \in \tilde{B}$, it means that z_v must be non-zero (note that all the edges from $\tilde{B}$ to C are saturated by F). Using this argument, we can see that any leaf in $\tilde{B}$ with a neighbour with degree greater than one has non-zero z value, Therefore there cannot exist two leaves in $\tilde{B}$ by Lemma 11 (Note that any $\tilde{B}$ vertex in a non-trivial tree has positive y value). □

C Omitted proofs

In this section, we present the omitted proofs due to space limitation.

Proof. (Full Proof of Theorem 2). We consider Algorithm 1. First, note that the algorithm is a polynomial-time algorithm, since the EGD can be computed using Edmonds' Blossom algorithm (see [16]), and the matching M^* can be computed by reducing the problem to maximum weighted matching on a bipartite graph, as follows: consider the bipartite graph obtained by first taking $G[B \cup C]$, then removing the edges between C-vertices, and finally shrinking each non-singleton component of $G[B]$ into one pseudonode (ignore multiple copies of edges created by this last operation, if any). Assign zero weight to the edges incident into pseudonodes, and unit weight to all other edges. Then compute a C-perfect matching $\bar{M}$ of maximum weight on this bipartite weighted graph. It is not difficult to show that $\bar{M}$ can be extended to a maximum matching M^* of G with the desired property, by adding a perfect matching between vertices in D, and a near-perfect matching in each non-singleton component of $G[B]$ as to fulfill properties (a),(b),(c) of Lemma 7.

Second, note that $|S| = k$, matching the lower bound given in Lemma 8. Therefore, to finish the proof, we only need to show that $G \setminus S$ is stable. We do this by constructing a fractional vertex-cover y of $G \setminus S$ with $\mathbf{1}^T y = |M^*|$. Since M^* is clearly a maximum matching in $G \setminus S$, this proves stability. Partition the set B into two sets: B_1 and B_2, where B_2 contains the singleton vertices in $G[B]$ and B_1 all the rest. Note that $S \subseteq B_1$. Let $C_1 \subseteq C$ be the set of vertices that are matched to vertices in B_1 by M^*. We assign $y_v = \frac{1}{2}$ for all $v \in D \cup (B_1 \setminus S) \cup C_1$. Then we repeat the following process: for each $v \in B_2$ that is adjacent to some node in $u \in C$ with current assigned value $y_u = \frac{1}{2}$, set $y_v = \frac{1}{2}$ and $y_w = \frac{1}{2}$, where $w \in C$ is the vertex matched to v by M^*. Note that such w must exists, i.e., v cannot be a vertex in B_2 exposed by M^*: if this is the case, then it is easy to realize that there is an M^*-alternating path P of even length between $v \in B_2$ and some vertex $\bar{v} \in B_1$: taking $P \Delta M^*$ we would obtain another maximum matching in G that exposes one less singleton vertex in $G[B]$, namely v, contradicting our choice of M^*. We repeat this process until each vertex $v \in B_2$ that does not have an assigned y-value is adjacent only to vertices in C that also do not have an assigned y-value. At this point, we set $y_v = 0$ for all remaining vertices in B_2 and $y_u = 1$ for all remaining vertices in C. Note that all vertices with y-value 1 in C are matched to (a subset of) vertices of y-value 0 by M^*, and M^* induces a perfect matching on the set of vertices with y-value $\frac{1}{2}$. Therefore, by construction, $\mathbf{1}^T y = |M^*|$. Furthermore, y is a fractional vertex cover of $G \setminus S$, since vertices with y-value 0 are only adjacent to vertices with y-value 1. The result follows. □

Proof. (*Proof of Lemma 1*) Since any minimum weight stabilizer is a minimal stabilizer, by Theorem 1, $\nu(G \setminus S) = \nu(G)$, so $S \subseteq B$ as otherwise $\nu(G \setminus S) < \nu(G)$. Note that if G_i is a singleton, (ii) and (iii) hold trivially. So let us assume G_i is a non-singleton component of $G[B]$, and let M be maximum matching of $G \setminus S$. Suppose M exposes a vertex v of G_i, so by Lemma 9, $V_i \cap S = \{v\}$. Suppose v is not a vertex with minimum weight in G_i, and let g_i be a vertex in G_i with minimum weight. We claim $S' = S \setminus \{v\} \cup \{g_i\}$ is a vertex-stabilizer. Let P be an M-alternating path in G_i from v to g_i which exists as G_i is factor-critical and both M and M' induce a near perfect matching in G_i. Let $M' = M \Delta P$. We show that M' is a vertex-stabilizer-certificate of S' (see Lemma 10). We need to show that there does not exist an M'-alternating path to a vertex in G_i. This follows from the fact that we removed some vertex from G_i and by Lemma 7, all vertices in a non-singleton component after a removal of one vertex, are essential. So M' is a vertex-stabilizer-certificate of S' and by Lemma 10, S' is a vertex-stabilizer but this gives a contradiction as S' has a smaller weight than S. Therefore, if $S \cap V_i \neq \emptyset$ then S includes a minimum weight vertex of V_i. □

Proof. (*Proof of Lemma 2*) Suppose $\tilde{S}$ is a B_1-essentializer. We construct a vertex-stabilizer S of G from $\tilde{S}$ as follows: for each $v \in \tilde{S}$, if vertex $v = g_i \in B_1$, then include a minimum weight vertex g_i of G_i in S, otherwise $v \in B_2$, so include the corresponding singleton v in S. By construction of G_b, S has the same weight as $\tilde{S}$. Now we construct a maximum matching M which is a vertex-stabilizer-certificate of S. Let $\tilde{M}$ be a maximum matching of G_b. The matching $\tilde{M}$ translates to a matching between C vertices and B vertices (covering all C vertices). Add edges corresponding to $\tilde{M}$ to M. Also add a perfect matching between D vertices to M. Now for each non-singleton component G_i, if $g_i \in S$ add a near-perfect matching of $G_i - g_i$ to M, otherwise since all vertices in $B_1 \setminus S$ are essential (by Definition 1), a vertex g_i in G_b is matched by $\tilde{M}$, equivalently some vertex v in G_i is matched by M, so add a near-perfect matching in $G_i - v$ to M. Note that M is a matching in $G \setminus S$ and it satisfies all the conditions in Lemma 7, so M is a maximum matching of G (and $G \setminus S$). Now, we show that there cannot exist an M-alternating path from an M-exposed vertex v in G to a vertex in a non-singleton component G_i, so by Lemma 10 S is a vertex-stabilizer. Assume the contrary. Let P be a subpath that contains exactly one vertex v' in G_i. Note that this path does not use an edge from C to D as if we start at v' and continue along P whenever P reaches a C vertex (from B side), in the next step it should use a matching edge which is between B and C (by Lemma 7), and at last we reach vertex v' which is matched by M (to a C vertex). In fact, this path translates to an $\tilde{M}$-alternating path in G_b as the only vertex in a non-singleton component is the last vertex, i.e., intermediate vertices are either in C or B_2. This gives us a contradiction as this path makes g_i inessential, so $\tilde{S}$ is not a B_1-essentializer.

Now suppose $S \subseteq V$ is a minimum weight vertex-stabilizer. Define $\tilde{S}$, by including a vertex g_i if $S \cap G_i \neq \emptyset$ or including $v \in B_2$ if $v \in S$. Since by Lemma 1, if $S \cap G_i \neq \emptyset$, S includes a minimum weight vertex of G_i, S and $\tilde{S}$ have the same weight. By Theorem 1, there exists a maximum matching M of G which lies in $G \setminus S$. By Lemma 7, M matches all C vertices to a subset of B which cannot contain two vertices from the same non-singleton component of $G[B]$, so M gives a matching $\tilde{M}$ covering all C vertices of $G_b \setminus \tilde{S}$ ($|\tilde{M}| = |C|$). Therefore, $\nu(G_b \setminus \tilde{S}) = \nu(G_b) = |C|$. Note that if a B_1-vertex g_i is not essential in $G_b \setminus \tilde{S}$, then either g_i is not matched by $\tilde{M}$ which means that there exists a M-exposed vertex in G_i which is a contradiction by Lemma 9. Now if g_i is matched by $\tilde{M}$ but it is inessential, it means that there exists a $\tilde{M}$-alternating path from an $\tilde{M}$-exposed vertex in $G_b \setminus \tilde{S}$ to g_i. Let P denote the subpath containing exactly one B_1-vertex, say $g_{i'}$ (might be different from g_i). Note that P only uses edge between B and C and contains exactly one B_1-vertex so P translate to an M-alternating path from an M-exposed vertex in G to a vertex in $G_{i'}$ which is contradiction by Lemma 9. □

Proof. (*Proof of Lemma 3*) Let $\tilde{S}$ be a B_1-essentializer of G_b. Let (B_b, C_b, D_b) be EGD of $G_b \setminus \tilde{S}$. Note that B_b is a subset of B_2 by definition of B_1-essentializer, so $C_b \subseteq C$. Set D_b includes $C \setminus C_b$, call these vertices $D_b(C)$, and any vertex in $\tilde{B} \setminus (\tilde{S} \cup B_b)$, call these vertices $D_b(\tilde{B})$. Moreover, by Lemma 7, there exists a perfect matching between vertices of D_b, and since G_b is bipartite this means that there exists a perfect matching between $D_b(C)$ and $D_b(\tilde{B})$, i.e., $\tilde{M}$ defines a bijection σ between $D_b(C)$ and $D_b(\tilde{B})$. Now we define a feasible (y, z) corresponding to $\tilde{S}$. Define variable y as follows: for $v \in \tilde{B}$, $y_v = 1$ if $v \in D_b(\tilde{B})$ and $y_v = 0$ otherwise, and for $u \in C$ let $y_u = 1$ if $u \in C_b$ and $y_u = 0$ otherwise. For $v \in \tilde{B}$, define $z_v = 1$ if $v \in \tilde{S}$, and $z_v = 0$ otherwise. For each $u \in B_1$, Inequality (1) is satisfied since $\tilde{S}$ is a B_1-essentializer, either $v \in \tilde{S}$, i.e., $z_v = 1$, or v is essential which means that it belongs to $D_b(\tilde{B})$, i.e., $y_v = 1$. For each $v \in B_2$, Inequality (2) is satisfied as either $v \in D_b(\tilde{B})$, so $y_v = 1$, or $v \in \tilde{S}$, so $z_v = 1$, or $v \in B_b$, so v is inessential and every neighbour u of v is in C_b, so $y_u = 1$. y satisfies Inequality (3), since using bijection σ, for any $u \in A$ with $y_u = 0$, we have $y_{\sigma(u)} = 1$ and $\sigma(u) \in N(u)$. y also satisfies Inequality (4), since for $u \in C$ either C has $y_u = 1$ or $y_{\sigma(u)} = 1$. Clearly, the binary constraints are satisfied for y and z.

Now let (y, z) be a point of P_I. Define set $\tilde{S} = \{v : z_v = 1\}$. Define the set $D_b(C) = \{u \in C : y_u = 0\}$ and $D_b(\tilde{B}) = \{v \in \tilde{B} : y_v = 1\}$. Note that for every $v \in B_1 \setminus \tilde{S}$, we have $y_v = 1$ by Inequality (1), so $v \in D_b(\tilde{S})$. Moreover, for any set $A \subseteq D_b(C)$, we have $y(N(A)) \geq |A|$. Therefore, by Hall's condition, there exists a matching M which matches each vertex of $D_b(C)$ to $D_b(\tilde{B})$. Since for any $u \in C \setminus D_b(C)$, $y_u = 1$, by Inequality (4), first $|D_b(C)| = |D_b(\tilde{B})|$, and second for any $v \in \tilde{B} \setminus D_b(\tilde{B})$, we must have $y_v = 0$ as $\sum_{v \in C \cup D_b(\tilde{B})} y_v = |C|$. Since (y, z) satisfies Inequality (2), each vertex v in $\tilde{B} \setminus D_b(\tilde{B})$ which is not in $\tilde{S}$, i.e., $z_v = 0$, cannot be a neighbour of $u \in D_b(C)$. Since there exists a perfect matching inside $G_b(1)$ any maximum matching of $G_b \setminus \tilde{S}$ induces a perfect matching in $G_b(1)$ and so every $v \in B_1 \setminus \tilde{S}$ is essential. □

Proof. (*Proof of Lemma 4*) We show an existence of a max $s - t$ flow in $\mathcal{N}$ of value $y(\tilde{B}) = |C| - y(C)$ by showing that the minimum-cut capacity in $\mathcal{N}$ is $|C| - y(C)$. Let $\mathcal{S}$ be a min-cut in $\mathcal{N}$. Since $\{s\}$ is a $s - t$ cut with capacity $|C| - y(C)$, capacity of $\mathcal{S}$ is at most $|C| - y(C)$. Let $A = \mathcal{S} \cap C$. Since $\mathcal{S}$ cannot contain an edge in $\tilde{E}$ (they have capacity ∞), $N(A) \subseteq \mathcal{S}$, so the capacity of $\mathcal{S}$ is at least

$$\begin{aligned}\sum_{u \in C \setminus A} 1 - y_u + \sum_{v \in N(A)} y_v &= |C \setminus A| - y(C \setminus A) + y(N(A)) \\ &\geq |C \setminus A| - y(C \setminus A) + |A| - y(A) = |C| - y(C).\end{aligned}$$

where the inequality follows from the fact that (y, z) is a feasible solution and satisfies Inequality (3). Hence a min-cut has capacity $|C| - y(C)$ and by Max-Flow Min-Cut Theorem there exists a desired flow. $\square$

Proof. (*Proof of Lemma 5*) Let $B' = \{v \in \tilde{B} : y_v > 0\}$. Each $v \in B'$ belongs to a non-trivial tree T_i (obtained from a network $\mathcal{N}$ constructed in Appendix B). By Corollary 1, there is at most one leaf B'-vertex in each tree. Root each of tree T_i at a leaf B'-vertex if there exists any or at a C-vertex otherwise, then match each B' vertex to one of its children. Note that this is always possible since in each of these rooted trees, there does not exists a B'-leaf which is not a root. So there exists a matching from $\tilde{B}$ to C-vertices lying in some tree, i.e., vertices with $1 - y_u > 0$ with size $|B'|$. $\square$

Proof. (*Proof of Lemma 6*) Let us restate ($\mathcal{P}_2$) labelling its inequalities as follows:

$$\begin{aligned} &\min w(B_1 \setminus B') + \textstyle\sum_{u \in C} \overline{w}_u f_u && (\mathcal{P}_2)\\ &\text{s.t.} \quad |N'(A)| \geq \textstyle\sum_{u \in A} f_u, \qquad \forall A \subseteq C && (7)\\ &\qquad \textstyle\sum_{u \in C} f_u = |B'|, && (8)\\ &\qquad 0 \leq f_u \leq 1, \qquad\qquad \forall u \in C. \end{aligned}$$

We first prove that an optimal solution of ($\mathcal{P}_1$) can be mapped into a solution of ($\mathcal{P}_2$) with no greater weight.

Let (z, y) be an optimal solution of ($\mathcal{P}_1$). Define $f_u = 1 - y_u$ for $u \in C$. For $A \subseteq C$, $y(N(A)) \geq |A| - y(A) = \sum_{v \in A}(1 - y_v) = f(A)$, and since $y(N(A)) = |N(A) \cap B'| = |N'(A)|$, we have that f satisfies Inequality (7). Inequality (8) is also satisfied since $f(C) = |C| - y(C)$ which is equal to $y(\tilde{B}) = |B'|$ by Inequality (4). Note that optimality of (z, y) implies $z_v = 0$ for $v \in B'$, $z_v = 1$ for $v \in B_1 \setminus B'$, and $z_v = \max_{u \in N(v)} (1 - y_u)$ for $v \in B_2 \setminus B'$. We get

$$\sum_{v \in \tilde{B}} w_v z_v = w(B_1 \backslash B') + \sum_{v \in B_2 \backslash B'} w_v \max_{u \in N(v)} (1 - y_u) \geq w(B_1 \backslash B') + \sum_{v \in B_2 \backslash B'} w_v \sum_{u \in N(v)} \frac{1 - y_u}{\delta} =$$

$$= w(B_1 \setminus B') + \sum_{u \in C} \sum_{v \in N(u) \cap (B_2 \setminus B')} \frac{w_v}{\delta}(1 - y_u) = w(B_1 \setminus B') + \sum_{u \in C} \overline{w}_u f_u.$$

We now prove that an integral solution of ($\mathcal{P}_2$) can be mapped into an integral solution of ($\mathcal{P}_1$) whose weight is at most a δ-factor larger.

Given an integral f, construct an integral solution (z, y) of ($\mathcal{P}_1$) as follows. (i) Set $y_u = 1 - f_u$ for $u \in C$, $y_v = 1$ for $v \in B'$, and $y_v = 0$ for $v \in \tilde{B} \setminus B'$. (ii) Set $z_v = \max_{u \in N(v)}(1 - y_u)$ for $v \in B_2 \setminus B'$. (iii) Set $z_v = 1 - y_v$ for $v \in B_1 \setminus B'$. (iv) Set $z_v = 0$ for $v \in B'$. Clearly this solution is integer. We now show feasibility. Inequality (1) and (2) hold trivially. Inequality (3) is satisfied, since for each $A \subseteq C$, $y(N(A)) = N'(A)$ and therefore this inequality is equivalent to inequality (7), satisfied by f. Inequality (4) holds as $y(C) = |C| - f(C) = |C| - |B'|$ and $y(\tilde{B}) = |B'|$. Finally, we get:

$$\sum_{v \in \tilde{B}} w_v z_v = \sum_{v \in B_1 \backslash B'} w_v + \sum_{v \in B_2 \backslash B'} w_v (\max_{u \in N(v)} f_u) \leq w(B_1 \backslash B') + \sum_{v \in B_2 \backslash B'} w_v \sum_{u \in N(v)} f_u$$

$$\leq \delta\Big(w(B_1 \setminus B') + \sum_{u \in C} \sum_{v \in N(u) \cap (B_2 \setminus B')} (\frac{w_v}{\delta}) f_u\Big) = \delta\Big(w(B_1 \setminus B') + \sum_{u \in C} \overline{w}_u f_u\Big). \square$$

D Polynomial-time algorithm for the M-stabilizer problem

We consider in this section the optimization problem of blocking as few players as possible in order to make a *given* set of deals realizable as a stable outcome. This translates into finding a min-cardinality vertex-stabilizer S with the additional constraint that S must avoid (i.e. must be element-disjoint from) a given maximum matching M. We call such S an M*-vertex-stabilizer*.

We prove that this problem is polynomial-time solvable. This is in contrast with the edge-removal setting, where the analogous question has been studied by [4] and shown to be as hard as finding a minimum vertex-cover.

Theorem 6. *There is a polynomial-time algorithm to compute a minimum M-vertex-stabilizer in a given graph, both in the unweighted and in the weighted setting.*

Proof. Let us first consider the unweighted setting. Let S contain all M-exposed vertices with an alternating path to a vertex in a non-singleton component. Note that by Lemma 9, any M-vertex-stabilizer S', which is also a vertex-stabilizer, along with matching M has to include S. By Lemma 10, S is a vertex-stabilizer, so S is a minimum cardinality M-vertex-stabilizer. Note that S can be computed in polytime by running a modified *breadth first search* (BFS) algorithm for each M-exposed vertex which is only allowed to use M edges at even levels.

At last, we point out that in the weighted case, the set S obtained as above is in fact a min-weight M-vertex-stabilizer. This is because any vertex included in S by the algorithm, must actually be in any minimum vertex-stabilizer by reasoning in Lemma 9 and Lemma 10, and we showed that removal of such vertices yields a stable graph. $\square$

E Finding Minimal Vertex-Stabilizer is *NP*-hard

Our goal is to prove the following theorem:

Theorem 7. *The min-weight vertex-stabilizer problem and the max-weight vertex-stabilizer problem are NP-hard, even if there are only 2 distinct weights.*

We will show that the problem is *NP*-hard by using a reduction from *minimum weight satisfiability* (MIN SAT) problem. A MIN SAT instance in the Boolean variables $x_1, x_2, \cdots, x_n$ is composed of a collection of *clauses* $C_1, C_2, \cdots, C_m$ and a non-negative weights $w_1, w_2, \cdots, w_m$ associated with each clause. Each clause $C_i = z_1 \vee z_2 \cdots \vee z_{k_i}$ for some $k_i \geq 1$ where each z_j is called a *literal* and is either a variable x_l or its negation $\overline{x}_l$. The goal is to assign each $x_1, x_2, \cdots, x_n$ a value $true$ or $false$ so that the total weight of satisfied clauses are minimized. A clause is said to be satisfied if one of its literals is assigned value of $true$.

MIN SAT was introduced by Kohli et al.[4] and they showed that the problem is *NP*-hard even for the unweighted version where each clause contains exactly two literals. They also showed that the problem remains *NP*-hard even if all clauses are Horn clauses (i.e., each clause has at most one complemented variable).

We start with proving the following.

Theorem 8. *Finding minimum weight B_1-essentializer is* NP*-hard.*

Proof. We show that there is a correspondence between an optimal solution of MIN SAT instance in the unweighted case and the minimum weight B_1-essentializer. Consider a MIN SAT instance defined as above with unit weight for each clause. Define graph $G_b = (\tilde{B} \cup C, \tilde{E})$ with $\tilde{B} = B_1 \cup B_2$ as follows (see Fig. 5):

- Let $C = \cup_{j=1}^{n} \{T_j, F_j\}$, i.e., one vertex for each possible assignment for variable x_j.
- Let $B_1 = \cup_{j=1}^{n} x_j$, i.e., one pseudonode associated with each variable.
- Let $B_2 = \cup_{i=1}^{m} C_i \cup_{j=1}^{n} \{T_j^1, T_j^2, F_j^1, F_j^2\}$, i.e., B_2 consists of a set of vertices associated with clauses and four extra vertices for each variable x_j.
- The edge set $\tilde{E}$ is defined as follows:
  - For each T_j put edges $(T_j, x_j), (T_j, T_j^1), (T_j, T_j^2)$ and one edge for each clause C_i containing x_j.
  - For each F_j put edges $(F_j, x_j), (F_j, F_j^1), (F_j, F_j^2)$ and one edge for each clause C_i containing $\overline{x}_j$.
- Define $w_v = 4n + m + 1$ for each $v \in B_1$, i.e., variables $x_1, x_2, \cdots, x_n$, and $w_v = 1$ for each $v \in B_2$.

First we show that how to obtain a B_1-essentializer $\tilde{S}$ from an assignment of variables. For each variable x_j, if it is assigned to $true$ then define $M(j) = T_j$ and otherwise $M(j) = F_j$. For each x_j, include all neighbours of $M(j)$ except x_j in $\tilde{S}$. We claim $\nu(G_b \setminus \tilde{S}) = |C| = 2n$. This is because for each $j \in [n]$, $M(j)$ can be matched to x_j and $\{T_j, F_j\} \setminus \{M(j)\}$ can be matched to one of T_j^1, T_j^2 or F_j^1, F_j^2 depending on whether $M(j) = T_j$ or $M(j) = F_j$. Since for each $j \in [n]$, x_j is the only neighbour of $M(j)$, so any maximum matching in $G_b \setminus S$ covers x_j, therefore all x_j's are essential. Hence, the set S is a B_1-essentializer. Now let us calculate the weight of S. For each $M(j)$, we have to remove two vertices $\{T_j^1, T_j^2\}$ or $\{F_j^1, F_j^2\}$ depending on whether $M(j) = T_j$ or $M(j) = F_j$ and any clause connected to $M(j)$, these clauses are exactly the clauses that are satisfied by the assignment. So the weight of S is $2 \times n$ plus the number of satisfied clauses by the assignment.

Now let S^* be a minimum weight B_1-essentializer. We claim S^* does not contain any pseudonode. If we remove all B_2 vertices, we get a graph that all B_1 vertices are essential. Since the weight of all B_2 vertices is strictly less than the weight of a single B_1 vertex, S^* cannot contain a B_1 vertex. Now, consider a maximum matching M of $G_b \setminus S^*$. Let $M(j)$ be the vertex in C that x_j is assigned to by M (x_j is essential so $M(j)$ is well-defined). For each x_j, without loss of generality, suppose $M(j) = T_j$ then T_j^1 and

[4] R. Kohli, R. Krishnamurti, and P. Mirchandani. *The minimum satisfiability problem*. SIAM Journal on Discrete Mathematics, 7(2):275283, 1994.

T_j^2 must be in S^* as these two vertices are exposed by M and they have M-alternating path to x_j. We claim any clause C_i which is a neighbour of T_j is in S^*. Suppose not, then C_i has to be essential, so it will be matched to some $u = T_{j'}$ or $u = F_{j'}$ for some $j' \neq j$, but with similar argument $\{T_{j'}^1, T_{j'}^2\}$ or $\{F_{j'}^1, F_{j'}^2\}$ must be in S^* which is contradiction to S^* being minimum weight since we can remove C_i instead of these two vertices and obtain another B_1-essentializer with a smaller weight. This argument shows that any minimum weight B_1-essentializer yields a matching which matches each x_j to $M(j) \in \{T_j, F_j\}$, and it includes all vertices in $N(M(j)) \setminus \{x_j\}$. Note that removing these vertices yields a graph in which all B_1 vertices are essential as they are the only neighbours of $\cup_{j\in[n]} M(j)$. Since S^* has a minimum weight, it only includes these vertices and so it has a weight $2n$ plus the number of clauses satisfied by assigning x_j to true if $M(j) = T_j$ and false otherwise. So there is a one-to-one correspondence between minimum weight B_1-essentializer and MIN SAT instance constructed above. So if we could solve the problem of finding a minimum weight B_1-essentializer in polytime, then we can solve MIN SAT in polytime which is not possible unless $P = NP$. □

Note that our proof shows a reduction from MIN SAT to finding a minimum weight B_1-essentializer that also preserves the cardinality of a maximum matching. It follows that finding a min-weight vertex-stabilizer is NP-hard as well. Since minimizing $\sum_{v\in S} w_v$ is equivalent to maximizing $\sum_{v\notin S} w_s$, the hardness holds as well for max-weight vertex-stabilizer. That is, we proved Theorem 7.

F Integrality Gap of Linear Programming Formulation ($\mathcal{P}_1$)

We show that the integrality gap of linear programming ($\mathcal{P}_1$) is $\Omega(\delta)$ even if we know $y_v \in \{0,1\}$ for $v \in \tilde{B}$ where δ is the maximum degree of vertices in B_2. The gap instance is a graph $G_b = (\tilde{B} \cup C, \tilde{E})$ where $\tilde{B} = B_1 \cup B_2$ for $B_1 = v_0$ and $B_2 = \{v_1, v_2, \cdots, v_{n^2}\}$, $C = \{u_1, u_2, \cdots, u_n\}$, and $E = \{(u,v) : u \in C, v \in B\}$ for integer $n > 1$. Define the weight function w to assign weight n^2 to v_0 and assign 1 to all the vertices in B_2. Assume $B' = \{v_0\}$. A feasible solution (x, y) of linear programming ($\mathcal{P}_1$) is shown in Fig. 6.

Claim. The solution (x, y) is a feasible fractional solution and has objective value n.

Proof. Inequality (1) is satisfied as for $v = v_0$, $y_v + z_v = 1 \geq 1$. Inequality (2) is satisfied as for any edge (u, v) for $v \in B_2$ and $u \in C$, $y_v + z_v + y_u = 0 + \frac{1}{n} + \frac{n-1}{n} \geq 1$. In order to show Inequality (3) is satisfied, we show that there exists a maximum flow that saturates all the edges in network $\mathcal{N} = (V_{\mathcal{N}}, A_{\mathcal{N}})$ constructed in Lemma 4: send flow on $\frac{1}{n}$ for each $(s, u) \in A_{\mathcal{N}}$, forward the flow of $\frac{1}{n}$ from each $u \in C$ to v_0, and finally send flow of 1 from v_0 to t. Therefore Inequality (3) is satisfied as otherwise the maximum flow should be strictly less than 1 in network $\mathcal{N}$. Inequality (4) is trivially satisfied as $\sum_{v\in\tilde{B}\cup C} y_v = 1 + n\frac{n-1}{n} = n = |C|$. Finally, since each $v \in B_2$ has weight 1 and $z_v = \frac{1}{n}$, the objective value of this solution is n. □

In order to exhibit the integrality gap, it remains to show that an integral solution has weight $\Omega(n^2)$.

Claim. Any integral solution (x, y) has weight $\Omega(n^2)$.

Proof. Since $y(V) = n$ and $y_{v_0} = 1$, there exists at least one vertex $u \in C$ with $y_u = 0$. Since every vertex $v \in B_2$ has y-value zero, we must have $z_v = 1$ for all $v \in B_2$ in order to satisfy Inequality (2). The objective function value for this solution is then $\Omega(n^2)$. □

G Integrality gap of the maximization formulation in (5)

In this section we prove a $\frac{3}{2}$ lower bound on the integrality gap of the maximization formulation in (5). Consider $G_b = (\tilde{B} \cup C, \tilde{E})$ which is a complete bipartite graph with $\tilde{B} = \{v_0, v_1, \ldots, v_p\}$ and $C = \{u_1, u_2\}$. We let $w_{v_0} = \sum_{i=1}^{p} w_i = W$, and $B_1 = \{v_0\}$. An optimal integral solution removes all but one vertices in $\tilde{B} \setminus \{v_0\}$, that is $y_{v_0} = y_{v_1} = 1$ and all other y-values are set to 0, $z_{v_0} = z_{v_1} = 0$ and all other z-values are set to 1, and has objective function value $W(1 + \frac{1}{p})$. However, a fractional solution can set: $y_{v_0} = 1$, $y_{u_1} = y_{u_2} = \frac{1}{2}$, $z_{v_1} = z_{v_2} = \cdots = z_{v_p} = \frac{1}{2}$, and set all other variables to 0, and has objective function value $W(1 + \frac{1}{2})$. For $p \to \infty$, the ratio $\to \frac{3}{2}$.

H Integrality gap of the minimization formulation in (5)

In this section, we will show that the integrality gap of linear programming

$$\min\Big\{\sum_{v \in \tilde{B}} w_v z_v : (z, y) \in P_f\Big\},$$

is $\Omega(n)$ where $|C| = 6n - 1$. This shows that our approximation ratio matches the integrality gap up to a constant and our result is tight. We would like to point out that in our example the maximum degree is constant. Since we have δ approximation algorithm in case of integral y value, this shows that in our example we must have fractional y's. In order to present the example achieving large integrality gap, we first need to address some results on bipartite expander graphs.

Definition 2. *A bipartite multigraph G with bipartition U and V is called a (K, A)* vertex expander *if for any set $S \subseteq U$ with size at most K, the neighbourhood $N(S)$ is of size at least $A \cdot |S|$*

Let $Bip^r_{N,D}$ be the set of D-regular bipartite graphs with N vertices on each side. Bassalygo[5] proved the following theorem[6]

[5] L. A. Bassalygo. Asymptotically optimal switching circuits. *Problemy Peredachi Informatsii*, 17(3):8188, 1981.

[6] for proof see Chapter 4 of "S. P. Vadhan. *Pseudorandomness.* Now Publishers Inc., Hanover, MA, USA, 2012."

Theorem 9. *For every constant D, there exists a constant $\alpha > 0$ such that for all N, a uniformly random graph from $Bip^r_{N,D}$ is an $(\alpha N, D-2)$ vertex expander with probability at least $1/2$.*

Using Theorem 9, we obtain the following result.

Corollary 2. *There exists a bipartite graph G with bipartitions $(U \cup U')$ and V ($|U| = |U'| = |V| = N$) and maximum degree $2D$ such that for any set $S \subseteq U \cup U'$ of size at most αN, $|N(S)| \geq |S|$ where α is a positive constant.*

Proof. By Theorem 9 choosing $D \geq 4$, there exists a D-regular bipartite graph G' with bipartition U' and V' ($|U'| = |V'| = N$) which is $(\alpha N, D-2)$ vertex expander for constant $\alpha > 0$. Let graph G be obtained from G' by adding a copy u of each vertex u' in U' to G', and connecting u to the same set of vertices as u'. Let U denote the set containing copy of vertices in U' and $V = V'$ denote the other partition of G. So the degree of each vertex in $U \cup U'$ is D and the degree of each vertex in V is $2D$.

We claim that G is $(\alpha N, 1)$ vertex expander. Let $S \subseteq U \cup U'$ with size at most αN. Define set S' to be obtained from S by including u' in S' if at least one of u or u' is in S. Note that $\frac{|S|}{2} \leq |S'| \leq |S| \leq \alpha N$ and $N(S) = N(S')$ by construction of G. Since G' is $(\alpha N, D-2)$ and $D \geq 4$, we have

$$|N(S)| = |N(S')| \geq |S'|(D-2) \geq 2|S'| \geq |S|.$$

□

Now, we are ready to present the integrality gap example. Let bipartite G be a graph with bipartition $U \cup U'$ and V' and edge set E' for $N = 2n$ and constant D in Corollary 2. Define graph G' with bipartition $\tilde{B} = B_1 \cup B_2$ and C where $B_1 = V$ and $B_2 = V' \cup W \cup W'$ and $C = U \cup U' \cup U''$ defined as follows

$$U = \{u_i\}_{i \in [2n]}, U' = \{u'_i\}_{i \in [2n]}, U'' = \{u''_i\}_{i \in [2n-1]},$$

$$V = \{v_i\}_{i \in [2n]}, V' = \{v'_i\}_{i \in [2n]}, W = \{w_i\}_{i \in [2n]}, W' = \{w'_i\}_{i \in [2n]}.$$

The edge set $\tilde{E}'$ of G' includes all the edges in G and the solid edges depicted in Fig. 7. Define weight of $v \in W \cup W'$ to be 1 and $v \in V \cup V'$ to be ∞. Define solution (y, z) for $\epsilon = \frac{1}{2n}$ as follows:

- For $v \in V$, $y_v = 1$ and $z_v = 0$.
- For $v' \in V'$, $y_{v'} = \epsilon$ and $z_{v'} = 0$.
- For $w \in W \cup W'$, $y_w = 0$ and $z_w = \epsilon$.
- For $u \in U \cup U'$, $y_u = 1 - \epsilon$.
- For $u \in U''$, $y_{u''} = 0$.

Claim. The solution (x, y) is a feasible solution and has objective value 2.

Proof. Inequality (1) is satisfied as for each $v \in B_1$, as $y_v = 1$ and $z_v = 0$. Inequality (2) is satisfied as for any edge (u, v) for $v \in B_2$ and $u \in C$, $y_u = 1 - \epsilon$ and $z_v + y_u = \epsilon$. In order to show Inequality (3) is satisfied, we show that there exists a feasible flow that sends $1 - y_u$ flow from each $u \in C$ and each $v \in B$ receives y_v flows. This flow is actually depicted in Fig. 7. Inequality (4) is trivially satisfied as $\sum_{v \in B} y_v = 2n \times 1 + 2n \times \epsilon$ and $\sum_{u \in C} y_u = 4n(1 - \epsilon)$ which sum up to $6n - 1$ for $\epsilon = \frac{1}{2n}$. Finally, since each $w \in W \cup W'$ has weight 1 and $z_v = \epsilon = \frac{1}{2n}$, the objective value of this solution is 2. □

In order to exhibit the integrality gap, it remains to show that the minimum B_1-essentializer in this graph has weight $\Omega(n)$. In fact, equivalently we will show that the minimum B_1-essentializer has weight at least αN where $\alpha > 0$ is a constant from Corollary 2 and $N = 2n$.

Claim. The minimum B_1-essentializer for G has weight at least αN.

Proof. Any B_1-essentializer S with finite cost does not remove any of vertices in $V \cup V'$. Since $B_1 = V$, every vertex in V is essential in $G' \setminus S$, so they have to be matched by a maximum matching M^* in $G \setminus S$. Note that $|U''| = 2n - 1$, so at least one of $v \in V$ must be matched to a vertex $u \in U$ (see Fig. 8). Since vertices in V' have ∞ cost as well, all the neighbours $N(u) \cap V'$ must be essential as well. Let U_1 denote the set of vertices in $U \cup U'$ that vertices in $N(u) \cap V'$ are matched to by M^* (see Fig. 8). Note that similar argument holds for $N(U_1) \cap V'$, and they must be matched by M^* (see Fig. 8). We can generalize this argument. More precisely, let V_e denote the set of essential nodes in V', and let U_e denote the set of vertices that V_e is matched to by M^* (U_e does not include u as u is matched to v by M^*). All vertices in $N(U_e \cup \{u\}) \cap V'$ must be essential, so $N(U_e \cup \{u\}) \cap V' \subseteq V_e$. Since graph G is $(\alpha N, 1)$ vertex expander, if $|U_e \cup \{u\}| \leq \alpha N$, then $|N(U_e \cup \{u\}) \cap V'| \geq |U_e \cup \{u\}| > |U_e|$. On the other hand, V_e which includes $N(U_e \cup \{u\}) \cap V'$ is matched to U_e which is not possible. So size of U_e is at least αN which means that the size of V_e is at least αN. For each $u_i \in U_e$, B_1-essentializer must include w_i or w'_i depending on whether $u_i \in U$ or $u_i \in U'$, respectively. Hence B_1-essentializer has size at least αN or weight αN equivalently.
□

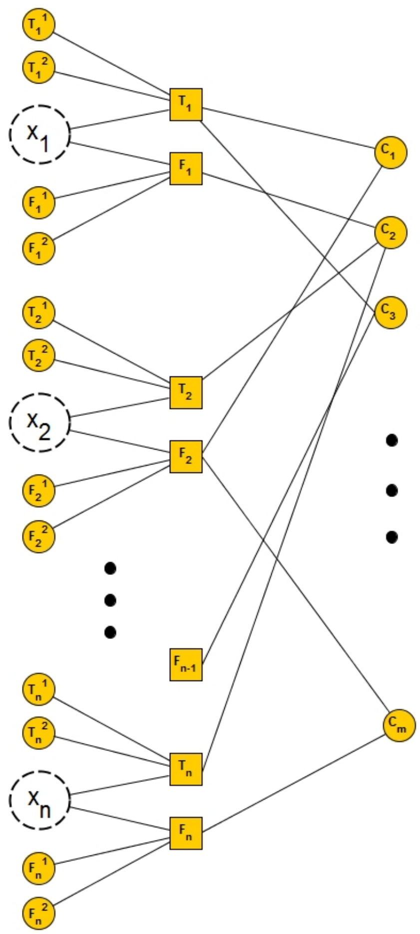

Fig. 5: Graph G_b corresponding to a MIN SAT instance. In this instance, $C_1 = x_1 \vee \overline{x}_2$, $C_2 = \overline{x}_1 \vee x_2 \vee x_n$, $C_3 = x_1 \vee \overline{x}_{n-1}$, and $C_m = \overline{x}_2 \vee \overline{x}_n$. The set $B_1 = \{x_1, x_2, \cdots, x_n\}$ (dotted circles), set $C = \cup_{i=1}^{n} \{T_i, F_i\}$ (square vertices), and set B_2 consists of the remaining vertices (solid circles). The weight vector is defined to be $4n + m + 1$ for B_1 vertices and 1 for B_2 vertices.

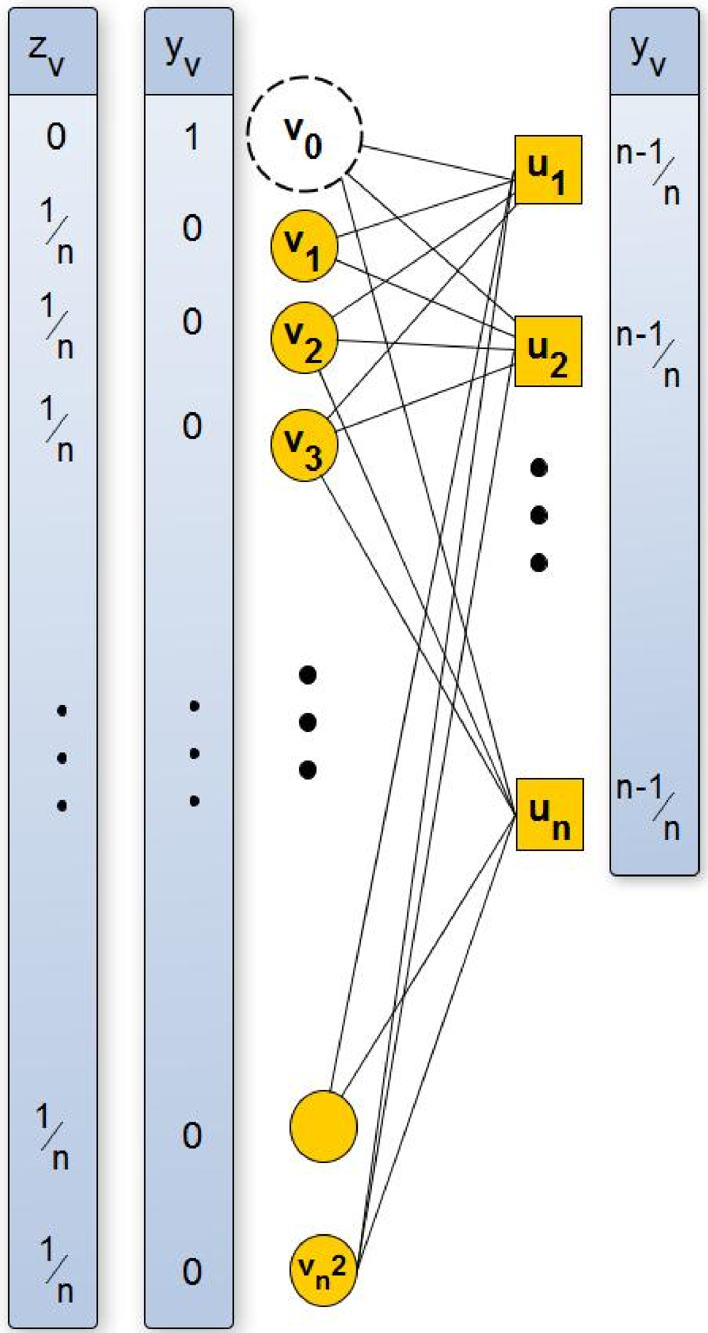

Fig. 6: An Instance showing large integrality gap. $\tilde{B}$ vertices are shown by circles: B_1 vertices are dotted circles, and B_2 vertices are solid circles. C vertices are shown by square.

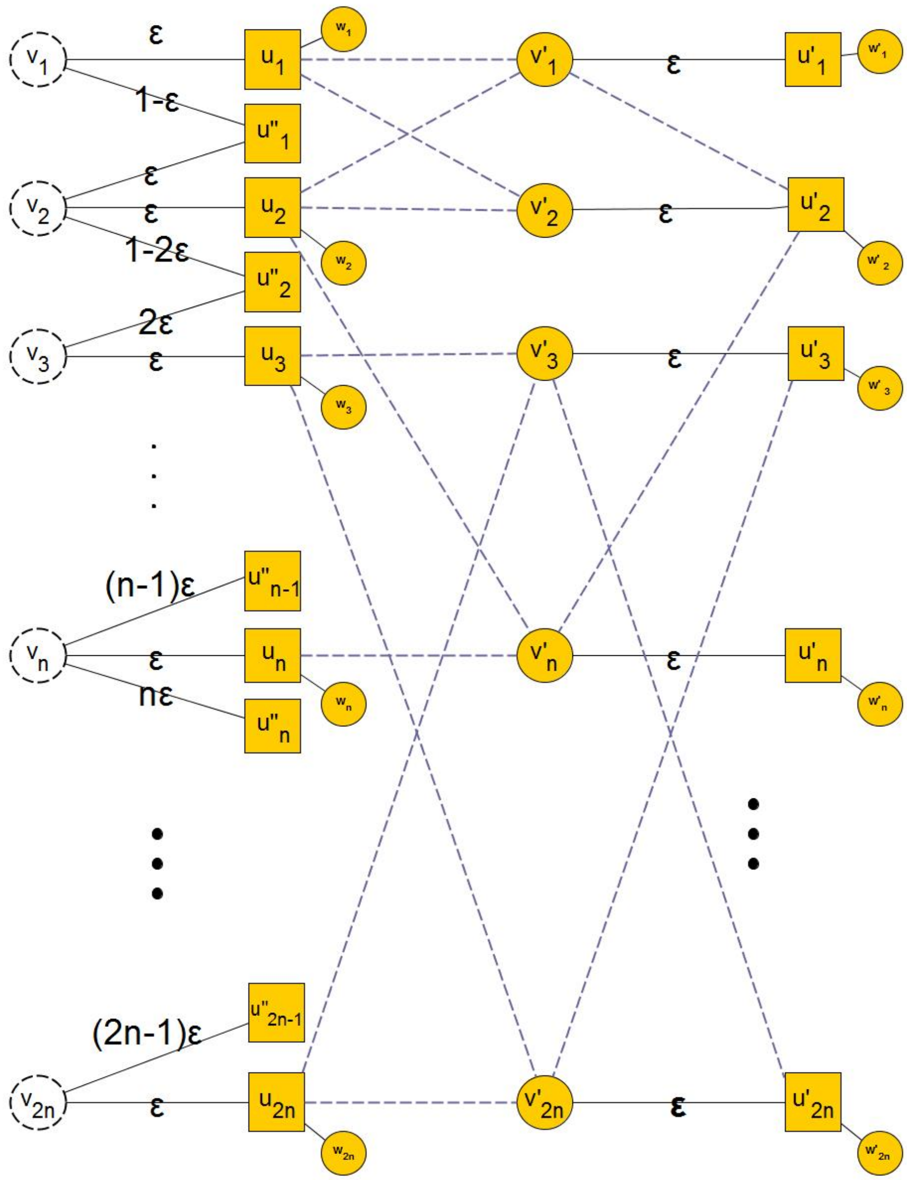

Fig. 7: Graph G' with $\tilde{B}$ vertices shown by circles: B_1 vertices are dotted circles; B_2 vertices are solid circles, and C vertices are shown by squares. The edge set of graph includes solid edges and the edge set of graph G used in the construction of G'.

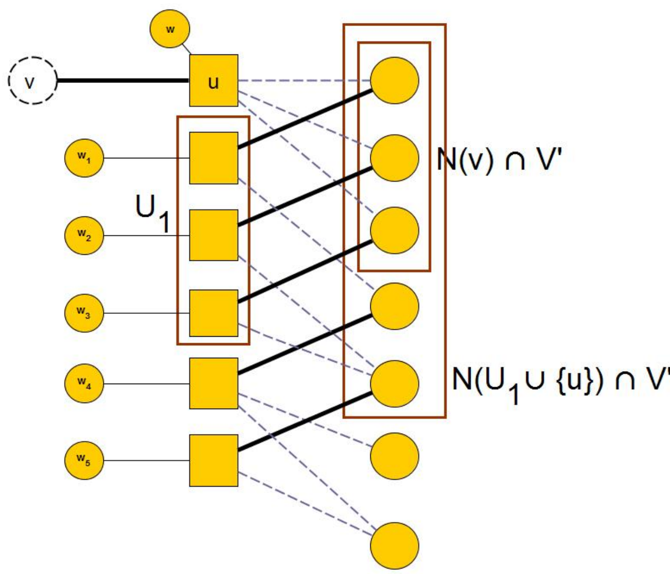

Fig. 8: Depiction of the process used in the proof of claim in Appendix H